Object-Based Goal Recognition Using Real-World Data

Roger Granada[0000−0001−5908−9247], Juarez Monteiro[0000−0002−8831−5343], Nathan Gavenski[0000−0003−0578−3086], and Felipe Meneguzzi[0000−0003−3549−6168]

School of Technology,
Pontifícia Universidade Católica do Rio Grande do Sul,
Porto Alegre, RS, Brazil
{roger.granada,juarez.santos}@acad.pucrs.br
nathan.gavenski@edu.pucrs.br, felipe.meneguzzi@pucrs.br

Abstract. Goal and plan recognition of daily living activities has attracted much interest due to its applicability to ambient assisted living. Such applications require the automatic recognition of high-level activities based on multiple steps performed by human beings in an environment. In this work, we address the problem of plan and goal recognition of human activities in an indoor environment. Unlike existing approaches that use only actions to identify the goal, we use objects and their relations to identify the plan and goal towards which the subject in the video is pursuing. Our approach combines state-of-the-art object and relationship detection to analyze raw video data with a goal recognition algorithm to identify the subject's ultimate goal in the video. Experiments show that our approach identifies cooking activities in a kitchen scenario.

Keywords: Goal Recognition · Relationship Detection · Object Detection.

1 Introduction

Goal recognition is the task of recognizing agents' goals, given a model of the environment dynamics in which the agent operates and a sample of observations about its behavior [25]. These observations can be events provided by sensors or actions performed by an agent. Goal recognition has several real-world applications, such as human-robot interaction [28], recognizing navigation goals [14], and recipe identification [10, 12]. Most approaches of goal recognition rely on plan libraries [3] to represent the agent behavior. However, recent advances use classical planning instead of plan libraries, showing that automated planning techniques can efficiently recognize goals and plans [18]. Although much effort has been focused on improving the recognition algorithms themselves [18], recent research has focused on the quality of the domain models used to drive such algorithms [2, 1, 17]. Unlike most approaches that assume that a human domain engineer can provide an accurate and complete domain model for the plan

recognition algorithm, recent work on goal recognition use the latent space [2, 1] to overcome this limitation. These approaches build planning domain knowledge from raw data using a latent representation of the input data. However, building such domain knowledge requires training an autoencoder with states where a transition of two subsequent images can represent the action.

In this paper, we explore the quality of automatically generated domain models that require minimal or no-interference of the human domain engineer. We evaluate our approach empirically using an existing kitchen-centered dataset. Since there is limited data for goal recognition based on videos, we manually annotated the dataset with bounding boxes and relationships between objects. Using this dataset, we perform experiments to compare both domain models, showing that a domain model built with minimal human interference achieves higher accuracy for most of the goals.

2 Background

2.1 Object and Relationship Detection

Object detection aims to determine whether there are any instances of objects from given categories in an image and return their spatial location and extent dimensions [15]. It is widely used in computer vision from simple tasks, such as surface inspection [27], to complex tasks, as autonomous robots operating in unstructured real-world environments [29]. Recently, approaches that learn raw data features, such as the ones based on deep learning [21, 32] have advanced state-of-the-art results. Deep learning is present in many architectures developed over the past few years, such as Faster R-CNN [21] and FSAF [32].

Visual relationship detection (VRD) aims to accurately localize a pair of objects and determine the predicate between them [13]. Recently, VRD has attracted more and more research attention in artificial intelligence since it is a step further on the understanding of images [33]. Nevertheless, recognizing individual objects is generally not sufficient to understand the relation of multiple items in a real-world scenario. VRD plays a crucial role in image understanding since it reflects the relationship between two objects, including relative positions (*e.g.*, *on*, *above*, *etc.*), actions (*e.g.*, *holding*, *moving*, *etc.*), as well as the human-object interactions such as *person holding ball*.

The task of recognizing relations between objects is represented by the widely adopted convention [16] that characterizes each relationship as a triplet in the form (s, r, o), where s and o are the subject and object categories of the relationship predicate r, respectively, *e.g.*, (*person*, *move*, *pan*), (*bowl*, *on*, *table*). Recent methods focus on identifying triplets containing relationships by using deep neural networks [13, 33]. Other approaches describe the creation of a scene graph that contains the relationship between pairs of objects [31].

2.2 Goal Recognition

Plan recognition is the task of recognizing how agents achieve their goals based on a set of observed interactions in an environment. Goal recognition is a particular

case of plan recognition in which only the goal is recognized [25]. Interactions in goal recognition can be some possible observed events performed by an agent in an environment, as well as actions/activities (*e.g.*, *cook*, *drive*), and changing properties in an environment (*e.g.*, *at home*, *at work*, *resting*). Recognizing agent goals and plans is vital to monitor and anticipate the agent behavior, such as in stories and life understanding [4], and educational environments [26].

Unlike relationship detection that identifies the predicate of a pair of objects to understand the scene better, goal recognition concentrates on identifying high-level, complex goals by exploiting relationships between predicates that are observations of the plan. Relationships between predicates are often encoded as a STRIPS-style [8] domain, which is used for *plan recognition as planning* (PRAP), since PRAP uses planning domains to generate hypotheses of possible plans consistent with observations [19].

Formally, we model planning domains of the agents being observed following a STRIPS [8] domain model $\mathcal{D} = \langle \mathcal{R}, \mathcal{O} \rangle$, where: $\mathcal{R}$ is a set of predicates with typed variables. Such predicates are associated with relations between objects in a grounded problem representing binary facts. Grounded predicates represent logical values according to some interpretation as facts, which are divided into two types: positive and negated facts, as well as constants for truth ($\top$) and falsehood ($\bot$). The set $\mathcal{F}$ of positive facts induces the state-space of a planning problem, which consists of the power set $\mathbb{P}(\mathcal{F})$ of such facts, and the representation of individual states $S \in \mathbb{P}(\mathcal{F})$. $\mathcal{O}$ is a set of operators $op = \langle pre(op), \mathit{eff}(op) \rangle$, where $\mathit{eff}(op)$ can be divided into positive effects $\mathit{eff}^+(op)$ (the add list) and negative effects $\mathit{eff}^-(op)$ (the delete list). An operator op with all variables bound is called action and the collection of all actions instantiated for a specific problem induces a state transition function $\gamma(S, a) \mapsto \mathbb{P}(\mathcal{F})$ that generates a new state from the application of an action to the current state. An action a instantiated from an operator op is applicable to a state S iff $S \models pre(a)$ and results in a new state S' such that $S' \leftarrow (S \cup \mathit{eff}^+(a))/\mathit{eff}^-(a)$.

A planning problem within $\mathcal{D}$ and a set of typed objects Z is defined as $\mathcal{P} = \langle \mathcal{F}, \mathcal{A}, \mathcal{I}, G \rangle$, where: $\mathcal{F}$ is a set of facts (instantiated predicates from $\mathcal{R}$ and Z); $\mathcal{A}$ is a set of instantiated actions from $\mathcal{O}$ and Z; $\mathcal{I}$ is the initial state ($\mathcal{I} \subseteq \mathcal{F}$); and G is a partially specified goal state, which represents a desired state to be achieved. A plan π for a planning problem $\mathcal{P}$ is a sequence of actions $\langle a_1, a_2, ..., a_n \rangle$ that modifies the initial state $\mathcal{I}$ into a state $S \models G$ in which the goal state G holds by the successive execution of actions in a plan π. Modern planners use the *Planning Domain Definition Language* (PDDL) as a standardized domain and problem representation medium [9], which encodes the formalism described here.

Bringing this all together, a goal recognition problem is a tuple $\mathcal{P}_{GR} = \langle \mathcal{D}, \mathcal{F}, \mathcal{I}, \mathcal{G}, O \rangle$, where $\mathcal{D}$ is a planning domain; $\mathcal{F}$ is the set of facts; $\mathcal{I} \subseteq \mathcal{F}$ is an initial state; $\mathcal{G}$ is the set of possible goals, which include a correct hidden goal G^* (*i.e.*, $G^* \in \mathcal{G}$); and $O = \langle o_1, o_2, ..., o_n \rangle$ is an observation sequence of executed actions, with each observation $o_i \in \mathcal{A}$, and the corresponding action being part of a valid plan π that sequentially transforms $\mathcal{I}$ into G^*. The solution for a goal recognition problem is the correct hidden goal $G \in \mathcal{G}$ that the

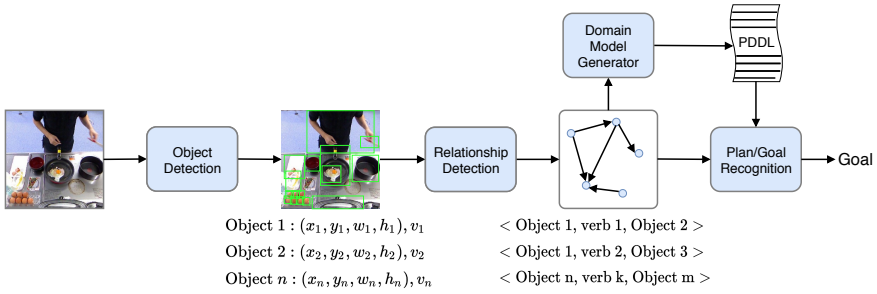

Fig. 1. Pipeline for goal recognition containing object detection, relationship recognition, planning domain generation and goal recognition modules.

observation sequence O of a plan execution achieves. An observation sequence O contains actions representing an optimal or sub-optimal plan that achieves a correct hidden goal, and this observation sequence can be full or partial. A full observation sequence represents the whole plan that achieves the hidden goal, *i.e.*, 100% of the observed actions. A partial observation sequence represents a sub-sequence of the plan for the hidden goal, such that a certain percentage of the actions actually executed to achieve G^* could not be executed.

3 Goal Recognition Using Relationship Information

In order to generate a domain representation of the environment from raw data, we perform the following processes (as illustrated in Figure 1): (a) Object detection that identifies and extracts a set of bounding boxes from the raw data that contain objects. (b) Relationship detection that extracts important relations between objects. (c) Domain model generation that converts relationships extracted from a module (b) into a STRIPS [8] domain model. Although (a) and (b) processes are nontrivial, there are significant research on both subjects recently [21, 32, 31]. In this paper, we address these problems by training off-the-shelf architectures with our dataset and consider that the resulting relationships contain a particular noise given each model's accuracy. The output of the relationship recognition module contains triplets with the relationship of the two objects, which can be represented as a scene graph [31].

Our contribution in this paper relies on generating domain models for goal recognition using minimal or no-interference from domain engineers. Upon using minimal interference, the human domain engineer has to define verbs and predicates in the extraction of complex relationships. Considering a no-interference domain generation, the STRIPS domain model is entirely generated from data.

3.1 Planning Domain Generation With Minimal Interference

To generate the planning domain model, we extract all relationships annotated in the ground truth of the training set and build a scene graph [31], as illustrated

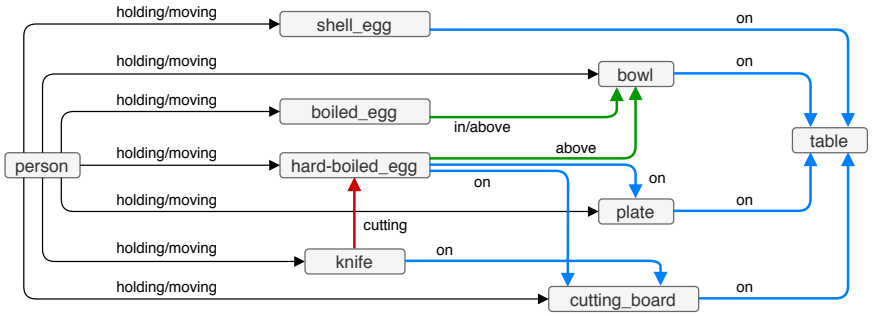

Fig. 2. Direct multigraph generated using an excerpt of the relationships from *boiled-egg* recipe. For simplicity, multiple edges are grouped into a single edge with relationships represented with a dash division.

in the excerpt in Figure 2. A scene graph is a directed multigraph $G = (V, E)$ containing relationships between objects, where V is a set of vertices containing the subjects and objects of a list of relationships, and E is a set of directed edges containing the relationship between objects. The direction of the arrow in the graph comes from the relationship of the *subject* to the *object*. For example, the relationship (*knife*, *cutting*, *hard-boiled_egg*) generates the red arrow in Figure 2.

From the scene graph, we can extract complex relationships between vertices. Here, we denominate a complex relationship as a relationship with more than two vertices. For example, from the excerpt scene graph presented in Figure 2, we can extract the relationship (*person*, *holding*, *boiled-egg*, *above*, *bowl*). We use the complex relationships to create the domain model.

This process considers a minimal interference of the human domain engineer since he only has to decide what verbs generate complex relationships. In our experiments, we identify four sets of actions. The four types of actions are: *handling actions*, *moving actions*, *cutting actions*, and *metamorphic actions*.

Handling actions are the actions associated with persons manipulating objects. The multigraph denotes this action by vertices connected to the *holding* edge. From these vertices and edge, we generate three actions: *hold*, *take*, and *put*. The *hold* action considers the vertices connected to the edge *holding*. For example, in Figure 2, we extract the action (*person*, *hold*, *shell_egg*). In order to generate *take* and *put* actions, the *relationship* in the triplet (*subject*, *relationship*, *object*) must be *holding* and the *object* must have at least one edge with another vertex in the multigraph where this edge is a preposition. Using the Figure 2, we extract the complex relationships (*person*, *take*, *shell-egg*, *on table*) and (*person*, *put*, *shell-egg*, *on*, *table*).

Moving actions occur when objects change their position in the scene. The multigraph denotes this action by vertices connected to the *moving* edge. To generate the action *moving*, the *relationship* in the triplet must be *moving* and the *object* must have at least two edges in the multigraph identified by prepositions. For example, we extract the complex relationship (*person*, *moving*, *hard-boiled-*

egg, *on*, *plate*) from Figure 2. This complex relationship describes that the *person* moves the *hard-boiled-egg* to place it *on* the *plate*.

Cutting actions occur when an object is divided into pieces using another object. The multigraph denotes this action by vertices connected to the *cutting* edge. To generate the *cutting* action, the *relationship* in the triplet (*subject*, *relationship*, *object*) must be *cutting*, and the *object* must have at least one edge containing a preposition. For example, in Figure 2, we extract this action from the edges connecting *knife*, *hard-boiled-egg* and *cutting-board* as (*knife*, *cutting*, *hard-boiled-egg*, *on*, *cutting-board*).

Finally, *metamorphic actions* occur when an object changes its form. Different forms can be seen in Figure 2 as the many names to describe an egg (*shell-egg*, *boiled-egg*, and *hard-boiled-egg*). In order to create this action, we consider as pre-conditions all the relationships that appear in the frame before the first appearance of the new form of the object. For example, in order to generate the *boiled-egg*, the frame before its first appearance contains the relationships where the *shell-egg* must be in the *pan* (*shell-egg*, *in*, *pan*), the *person* must be *holding* the *pan* (*person*, *holding*, *pan*) and the *pan* must not be *on* the *stove* (*pan*, *on*, *stove*).

3.2 Planning Domain Generation With No-Interference

To generate the planning domain model without human interference, we consider all relationships annotated in the ground truth of the training set. Using the annotated relationships, we create a binary vector containing all possible relations of the domain. An action is generated any time a transition occurs, *i.e.*, when some relationship changes between two frames. We encode these transitions in a binary vector containing all the possible relationships that appear in training data. This approach is similar to the one performed by Amado *et al.* [1]. However, instead of deriving binary vectors from an autoencoder, we consider a binary vector composed by all the relationships containing in the training set. Having generated binary vectors considering all transitions between frames, we derive a set of actions by performing a bit-wise comparison using each state before a transition s and the state after a transition s'. This process is summarized in Algorithm 1, where a modified XOR operation (XOR_E) on both states generates the effect of the transition, and a modified version of the XNOR operation ($XNOR_P$) generates the preconditions of candidate actions.

As described by Amado *et al.* [1], the *Effect XOR* (XOR_E) operation (Line 5) computes the *effect* of the action when applying a state s. This operation outputs 1 for *positive effects*, *i.e.*, when a bit changes from 0 in s to 1 in s', and -1 for *negative effects*, *i.e.*, when a bit changes from 1 in s to 0 in s'. When multiples transitions result in the same effect, we apply the *Precondition XNOR* ($XNOR_P$) operation (Line 10) to identify which bits do not change in all the set of states s of the transitions. The idea is that a bit that appears in all transitions with the same effect must be a necessary predicate to execute this action. The $XNOR_P$ operation outputs 1 for *positive preconditions*, *i.e.*, when the bit is 1 in all states that generate the effect, and -1 for *negative preconditions*, *i.e.*, when

Algorithm 1 Learn actions of a planing domain [1]

Require: Set of transitions T
function ACTION-LEARNER(T)
  $E \leftarrow \langle\rangle$ ▷ Map of actions
  $A \leftarrow \langle\rangle$ ▷ Set of generated actions
  for all $(s, s') \in T$ **do**
    $\mathit{eff} \leftarrow XOR_E(s, s')$
    $E(\mathit{eff}) \leftarrow E(\mathit{eff}) \cup s$
  for all $\mathit{eff} \in E$ **do**
    $pre \leftarrow \emptyset$ ▷ Derived pre-condition
    for all $s \in E(\mathit{eff})$ **do**
      $pre \leftarrow XNOR_P(pre, s)$
    $A \leftarrow A \cup \langle pre, \mathit{eff} \rangle$
  return A

the bit is 0 in all states that generate the effect. Using the algorithm 1, we can generate the PDDL domain model with a compressed number of actions.

3.3 Goal Recognition Problem

Before setting up the goal recognition problem, we must have a planning problem. We generate the planning problem by extracting the relationships existent in the first frame of the videos and the goal as the relationships existent in the last frame. As described in Section 2.2, we represent a goal recognition problem as a tuple $\mathcal{P}_{GR} = \langle \mathcal{D}, \mathcal{F}, \mathcal{I}, \mathcal{G}, O \rangle$, where the domain $\mathcal{D}$ we compute using Algorithm 1, facts $\mathcal{F}$ are represented by a binary vector encoding all possible relationships. We compute the initial state $\mathcal{I}$ as the intersection of the relationships existent in the initial frame of all videos of the training set. This set of relationships are converted to a binary vector representing the initial state. We generate the set of candidate goals $\mathcal{G}$ as the intersection of the relationships presented in the last frame of each recipe, and then encode the resulting relationships into a binary vector. Finally, for each frame of the test set, we derive the observations O by encoding the relationships presented in the frame into a binary vector representing the state.

After building a goal recognition problem, we can apply off-the-shelf goal recognition techniques, such as [19, 18]. The output of such techniques is the goal with the highest probability of being the correct one.

4 Experiments

In this section we describe the dataset we use in the experiments, the annotation process, the training of the detectors and the goal recognition execution. In order to evaluate our models, we generated a dataset by altering the level of observability available to the algorithm. We set five different percentages of observability: 100%, 70%, 50%, 30% and 10%. We also compare both approaches using real-world data extracted using a visual relationship detector [13].

4.1 Dataset

Previous work on goal recognition using a kitchen scene dataset [12] has the drawback of using limited dataset. Koller *et al.* [12] affirm that a thorough evaluation was not possible due to the small number of annotated videos. Their work uses a subset of MPII Cooking 2 dataset [22] annotated with objects that contains only two different performed recipes. Since a large number of images with ground truth object bounding boxes are critical for learning object detectors using a neural network, we decide to manually annotate a dataset with objects and relationships between objects.

The Kitchen Scene Context based Gesture Recognition dataset[1] (KSCGR) [23] is a fine-grained kitchen dataset that contains videos of 7 different subjects creating five menus for cooking eggs in Japan: *ham and eggs*, *omelet*, *scrambled egg*, *boiled egg*, and *kinshi-tamago*. The original ground truth annotation of the dataset aims to recognize cooking actions and comprises of 8 cooking gestures performed by the subject. The dataset is divided into *training*, *validation*, and *test* sets [10], where the *training* set contains 4 subjects, each of them performing 5 recipes, the *validation* set contains 1 subject performing 5 recipes, and the *test* contains 2 subjects, each performing 5 recipes. We select this dataset since it has been used for goal recognition [10].

4.2 Dataset Annotation

The dataset annotation consists of two tasks: the bounding boxes annotation and the relations annotation. For each task, we use two subjects that follow a similar protocol that dictates first to watch the video containing the recipe being performed and then annotate all bounding boxes and relations existent in each video frame. Finally, the subjects must generate a file for each video containing all annotated data. For bounding box annotation, we generate a list containing 30 category labels used at least in one recipe. The category labels contains: *person*, *baked egg*, *boiled egg*, *broken egg*, *hard-boiled egg*, *mixed egg*, *shell egg*, *ham egg*, *kinshi egg*, *scrambled egg*, *omelette*, *ham*, *pan*, *frying pan*, *pan handle*, *pan lid*, *bowl*, *chopstick*, *cutting board*, *dishcloth*, *glass*, *knife*, *milk carton*, *oil bottle*, *plate*, *saltshaker*, *spoon*, *turner*, *table*, and *stove*. The released dataset annotation[2] contains 2,356,829 instance-level annotations for objects. Both subjects annotate a single video, and we measure the quality of inter-annotator agreement using the Cohen's Kappa [5]. We consider a correct agreement if the bounding boxes have an intersection over union (IoU) greater than 0.7. Using this criteria, we achieve a $\kappa = 0.99$, indicating almost perfect agreement between annotators.

The process of relationship annotation aims to identify relations between objects that are interesting for the performed task. We consider 6 types of relations predicates: 3 spatial relations (*in*, *on*, and *above*) and 3 action relations (*cutting*, *holding*, and *moving*). Applying these predicates, we manually created a list containing 164 relations between the 30 category labels that reflects the interesting

[1] `http://www.murase.m.is.nagoya-u.ac.jp/KSCGR/`
[2] `https://github.com/rogergranada/kscgr_annotation`

relations between two objects in the dataset. This list characterizes each relationship as a triplet (s, r, o), where s and o are the subject and object categories respectively, and r is the relationship predicate. For example, a relation for a *ham on* the *cutting board* is identified as (*ham*, *on*, *cutting board*). The complete dataset annotation contains 2,269,151 triples for all 35 videos of the dataset. We compute the inter-annotator agreement using a single video annotated by both subjects, resulting in a $\kappa = 0.87$, indicating an almost perfect agreement.

4.3 Object and Relationship Detection

In order to generate triplets for our experiments, we train an off-the-shelf object and relationship detectors. It is important to note that our approach is independent of both detectors, although their accuracy may influence the final results. We use the Faster R-CNN [21] as object detector and VDR-DSR [13] as a relationship detector. We trained the Faster R-CNN using a VGG-16 [24] as the base network. We loaded weights pre-trained in PASCAL VOC 2007 dataset [7] and freeze the first five convolutional layers. In the other layers, we use a learning rate of 1e-3, which decreased by a factor of 10 after every epoch. VRD-DSR [13] uses the RoI features from Faster R-CNN to use as the visual appearance features and spatial location cues. As we previously extracted features from Faster R-CNN, we can perform only predicate detection using VRD-DSR. In such task, VRD-DSR predicts the correlated predicates given a pair of localized objects. We train the VRD-DSR using the VGG-16 as the backbone network for 20 epochs. We use Adam optimizer [11] for all networks and set the learning rate to 1e-5.

4.4 Goal Recognition

To test our approaches' ability to recognize goals using only the relationships between objects, we perform goal recognition using a landmark-based heuristics [18], which is the current state-of-the-art in goal and plan recognition. Following the current goal recognition research, we evaluate our approaches in terms of accuracy and spread using different observability levels. We set five different percentages of observability: 100%, 70%, 50%, 30% and 10%. Finally, we test our approaches using the real-world data from the VRD-DSR relationship detector.

Table 1. Experimental results on Goal Recognition problems using domain model with minimal human interference.

	Boiled egg			Ham egg			Kinshi egg			Omelette			Scrambled egg		
Obs	$\|O\|$	Acc	$S_{\mathcal{G}}$	$\|O\|$	Acc	$S_{\mathcal{G}}$	$\|O\|$	Acc	$S_{\mathcal{G}}$	$\|O\|$	Acc	$S_{\mathcal{G}}$	$\|O\|$	Acc	$S_{\mathcal{G}}$
10	591	1.00	3.36	575	0.76	1.64	513	0.13	2.17	585	0.67	1.74	521	0.58	2.41
30	1774	1.00	3.31	1727	0.77	1.65	1540	0.12	2.10	1758	0.48	2.16	1564	0.60	2.45
50	2957	1.00	3.41	2879	0.76	1.65	2568	0.11	2.15	2931	0.42	1.99	2608	0.58	2.45
70	4139	1.00	3.38	4031	0.77	1.64	3596	0.13	2.14	4103	0.52	1.94	3651	0.60	2.39
100	5914	1.00	3.39	5759	0.76	1.65	5137	0.12	2.15	5863	0.48	2.19	5217	0.59	2.41
VRD	5914	1.00	3.43	5759	0.65	1.54	5137	0.11	2.07	5863	0.56	2.31	5217	0.44	2.50

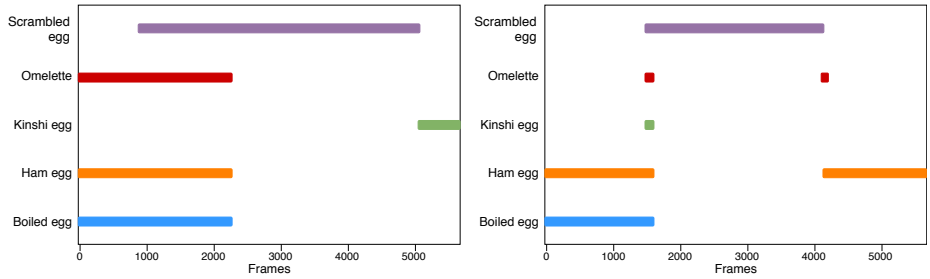

Fig. 3. Examples of possible goals in goal recognition for (a) *Kinshi* and (b) *Ham* eggs.

Table 1 summarizes the goal recognition performance using our approaches for all recipes of the test set, where *Obs* is the percentage of the plan that is actually observed; $|O|$ is the average number of observations; *Acc* (Accuracy) represents the average number of problems in which the correct goal was among the recognized goals; $S_{\mathcal{G}}$ (Spread of goal $\mathcal{G}$) is the average number of returned goals when multiple-goal hypotheses were tied in the recognition algorithm.

As we can see, the accuracy for *Boiled egg* was the highest when compared with the other recipes. It justifies since the *Boiled egg* recipe is the most dissimilar recipe. While the other recipes uses the *frying pan* to prepare the egg, the *Boiled egg* uses the *pan*. The egg is also different in this recipe, since it is not broken into the container. A counter point is illustrated by the *Kinshi egg* recipe, that achieved the lowest accuracy of all recipes. Figure 3 (a) illustrates the output of the goal recognizer to the *Kinshi egg* recipe, where we can see that in most frames the recipe *Scrambled egg* is predicted as the correct recipe. In fact, both recipes are very similar, when the *egg* is being cooked in the *frying pan*. However, in *Kinshi egg*, the *person* cuts the *egg* after taking it out of the *frying pan*, which represents the change in the correct goal by the goal recognizer in the last frames. Figure 3 (b) illustrates the candidate goals for *Ham egg*, where *Scrambled egg* is predicted mainly in parts of the video where the *egg* is in *frying pan*.

Table 2 shows the results achieved when using the domain model without human interference. When dealing with our automatically generated domain model, it seems that the plan focus on *Scrambled egg* actions, since in most plans this action is the candidate goal with the highest number of achieved landmarks. The unique case where it does not infer the *Scrambled egg* goal occurs when we feed the observations from *Boiled egg* recipe.

When using real-world data from the visual relationship detector (VRD), there is not a large difference between the scores achieved by using missing observations. As the goal recognizer is based on landmarks, the most of the noisy data do not interfere the goal prediction. When comparing our results with the results achieved by Granada *et al.* [10], our approach using a minimal inference can achieve a highest number of correct goals. In that work, some goals were never achieved since missing information would lead to wrong plans in the plan

Table 2. Experimental results on Goal Recognition problems using domain model with no human interference (data-driven).

	Boiled egg			Ham egg			Kinshi egg			Omelette			Scrambled egg		
Obs	$\|O\|$	Acc	$S_{\mathcal{G}}$	$\|O\|$	Acc	$S_{\mathcal{G}}$	$\|O\|$	Acc	$S_{\mathcal{G}}$	$\|O\|$	Acc	$S_{\mathcal{G}}$	$\|O\|$	Acc	$S_{\mathcal{G}}$
10	591	1.00	1.97	575	0.12	1.89	513	0.26	2.46	585	0.04	2.47	521	0.97	2.70
30	1774	1.00	1.95	1727	0.22	1.79	1540	0.24	2.42	1758	0.04	2.47	1564	0.98	2.71
50	2957	1.00	1.96	2879	0.23	1.79	2568	0.25	2.43	2931	0.04	2.49	2608	0.98	2.69
70	4139	1.00	1.96	4031	0.21	1.82	3596	0.25	2.42	4103	0.04	2.47	3651	0.97	2.67
100	5914	1.00	1.97	5759	0.25	1.77	5137	0.25	2.43	5863	0.04	2.49	5217	0.97	2.69
VRD	5914	1.00	1.96	5759	0.23	1.68	5137	0.26	2.33	5863	0.04	2.43	5217	0.97	2.73

library. As our work considers more than the sequence of actions to determine the correct goal, we improve the goal recognition process.

5 Related Work

Amado *et al.* [1] combines goal recognition techniques with deep autoencoders to generate domain theories from data streams. A pair of images is encoded into an autoencoder generating a 72 bits vector representation of the transition. These pairs of states are fed into an Action Learner to generate the domain model. Their experiments are evaluated in simple problems, such as Hanoi Tower and MNIST 8-Puzzle datasets. Using a kitchen scenario, Koller *et al.* [12] proposes a goal recognition approach that infers goals from video using spatial object annotation traces. In order to find the goal, their technique matches the properties of detected objects to the preconditions of planning operators, using a knowledge base. Unlike our method, they build a domain model using domain engineering.

Granada *et al.* [10] perform goal recognition using the KSCGR [23] dataset. Their approach modifies a symbolic plan recognition approach called Symbolic Behavior Recognition (SBR) to work with a Convolutional Neural Network (CNN). The CNN identifies individual actions from raw images that are then processed by a goal recognition algorithm that uses a plan library describing possible overarching activities to identify the subject's ultimate goal in the video. Although their approach uses the same KSCGR dataset, plan libraries limit the goal recognition process, since some recipes contain actions that appear only in the test set and not in the training set.

6 Conclusion

This paper describes two approaches for goal recognition using the relationship between objects in a kitchen scenario. The first approach uses minimum interference of a human domain engineer, while the second approach is totally data-driven. In order to perform experiments, we annotate a dataset of actions occurring in a kitchen scenario and make it freely available. Experiments using

both models shows that the model using a minimal human interference achieves better results when comparing with the automatically generated domain model.

As future work, we plan to test different objects and relationship detectors [6, 30] to verify their influence in the outcome results. We intend to test other goal recognizers such as probabilistic plan recognizers [20] and plan recognition algorithms to deal with incomplete domain models to handle inconsistencies and noise generated by the domain model generator.

References

1. Amado, L., Pereira, R., Aires, J.P., Magnaguagno, M., Granada, R., Meneguzzi, F.: Goal recognition in latent space. In: Proceedings of the 2018 International Joint Conference on Neural Networks. pp. 4841–4848 (2018)
2. Asai, M., Fukunaga, A.: Classical planning in deep latent space: Bridging the subsymbolic-symbolic boundary. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence. pp. 6094–6101 (2018)
3. Avrahami-Zilberbrand, D., Kaminka, G.A.: Fast and complete symbolic plan recognition. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence. pp. 653–658 (2005)
4. Charniak, E., Goldman, R.: Plan recognition in stories and in life. Machine Intelligence and Pattern Recognition **10**(1), 343–351 (1990)
5. Cohen, J.: A coefficient of agreement for nominal scales. Educational and Psychological Measurement **20**(1), 37–46 (1960)
6. Dai, B., Zhang, Y., Lin, D.: Detecting visual relationships with deep relational networks. In: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition. pp. 3298–3308 (2017)
7. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International Journal of Computer Vision **88**(2), 303–338 (2010)
8. Fikes, R., Nilsson, N.: Strips: A new approach to the application of theorem proving to problem solving. Artificial Intelligence **2**(3-4), 189–208 (1971)
9. Fox, M., Long, D.: Pddl2.1: An extension to pddl for expressing temporal planning domains. Journal of Artificial Intelligence Research **20**, 61–124 (2003)
10. Granada, R., Pereira, R., Monteiro, J., Barros, R., Ruiz, D., Meneguzzi, F.: Hybrid activity and plan recognition for video streams. In: The AAAI 2017 Workshop on Plan, Activity, and Intent Recognition (2017)
11. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
12. Koller, M., Patten, T., Vincze, M.: Plan recognition from object detection traces (preliminary report). In: The AAAI 2020 Workshop on Plan, Activity, and Intent Recognition (2020)
13. Liang, K., Guo, Y., Chang, H., Chen, X.: Visual relationship detection with deep structural ranking. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence. pp. 7098–7105 (2018)
14. Liao, L., Fox, D., Kautz, H.: Hierarchical conditional random fields for gps-based activity recognition. In: Robotics Research. pp. 487–506 (2007)
15. Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., Pietikäinen, M.: Deep learning for generic object detection: A survey. International Journal of Computer Vision **128**, 261–318 (2020)

16. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: Proceedings of the European Conference on Computer Vision. pp. 21–37 (2016)
17. Pereira, R.F., Meneguzzi, F.: Landmark-based heuristics for goal recognition. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence. pp. 8127–8128 (2018)
18. Pereira, R.F., Oren, N., Meneguzzi, F.: Landmark-based heuristics for goal recognition. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence. pp. 3622–3628 (2017)
19. Ramírez, M., Geffner, H.: Plan recognition as planning. In: Proceedings of the 21st International Jont Conference on Artifical Intelligence. pp. 1778–1783 (2009)
20. Ramírez, M., Geffner, H.: Probabilistic plan recognition using off-the-shelf classical planners. In: Proceedings of AAAI-10. pp. 1121–1126 (2010)
21. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Proceedings of the 28th International Conference on Neural Information Processing Systems. pp. 91–99 (2015)
22. Rohrbach, M., Rohrbach, A., Regneri, M., Amin, S., Andriluka, M., Pinkal, M., Schiele, B.: Recognizing fine-grained and composite activities using hand-centric features and script data. International Journal of Computer Vision **119**(3), 346–373 (2016)
23. Shimada, A., Kondo, K., Deguchi, D., Morin, G., Stern, H.: Kitchen scene context based gesture recognition: A contest in icpr2012. In: Revised Selected and Invited Papers of the International Workshop on Advances in Depth Image Analysis and Applications - Volume 7854 (WDIA'12). pp. 168–185 (2013)
24. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: arXiv preprint arXiv:1409.1556 (2014)
25. Sukthankar, G., Goldman, R.P., Geib, C., Pynadath, D.V., Bui, H.H.: Plan, Activity, and Intent Recognition: Theory and Practice. MKP Inc., 1st edn. (2014)
26. Uzan, O., Dekel, R., Seri, O., Gal, Y.K.: Plan recognition for exploratory learning environments using interleaved temporal search. AI Magazine **36**(2), 10–21 (2015)
27. Wang, Y., Liu, M., Zheng, P., Yang, H., Zou, J.: A smart surface inspection system using faster r-cnn in cloud-edge computing environment. Advanced Engineering Informatics **43** (2020)
28. Wang, Z., Mülling, K., Deisenroth, M.P., Amor, H.B., Vogt, D., Schölkopf, B., Peters, J.: Probabilistic movement modeling for intention inference in human–robot interaction. The International Journal of Robotics Research **32**(7), 841–858 (2013)
29. Yang, S., Wang, J., Wang, G., Hu, X., Zhou, M., Liao, Q.: Robust rgb-d slam in dynamic environment using faster r-cnn. In: Proceedings of the 3rd IEEE International Conference on Computer and Communications. pp. 2398–2402 (2017)
30. Zellers, R., Yatskar, M., Thomson, S., Choi, Y.: Neural motifs: Scene graph parsing with global context. In: Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition. pp. 5831–5840 (2018)
31. Zhang, J., Shih, K.J., Elgammal, A., Tao, A., Catanzaro, B.: Graphical contrastive losses for scene graph parsing. In: Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition. pp. 11535–11541 (2019)
32. Zhu, C., He, Y., Savvides, M.: Feature selective anchor-free module for single-shot object detection. In: Proceedings of the 2019 IEEE Conference on Computer Vision and Pattern Recognition. pp. 840–849 (2019)
33. Zhu, Y., Jiang, S.: Deep structured learning for visual relationship detection. In: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence. pp. 7623–7630 (2018)