Biologically Inspired Cognitive Architectures (2016) xxx, xxx–xxx

Available at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/bica

RESEARCH ARTICLE

Learning task goals interactively with visual demonstrations

James Kirk *, Aaron Mininger, John Laird

University of Michigan, Ann Arbor, MI, United States

Received 17 June 2016; received in revised form 1 August 2016; accepted 1 August 2016

KEYWORDS
Interactive task learning;
Goal learning;
Human-robot interaction

Abstract
Humans are extremely good at quickly teaching and learning new tasks through situated instructions; tasks such as learning a novel game or household chore. From studying such instructional interactions, we have observed that humans excel at communicating information through multiple modalities, including visual, linguistic, and physical ones. Rosie is a tabletop robot implemented in the Soar architecture that learns new tasks from online interactive language instruction. In the past, the features of each task's goal were explicitly described by a human instructor through language. In this work, we develop and study additional techniques for learning representations of goals. For game tasks, the agent can be given visual demonstrations of goal states, refined by human instructions. For procedural tasks, the agent uses information derived from task execution to determine which state features must be included in its goal representations. Using both approaches, Rosie learns correct goal representations from a single goal example or task execution across multiple games, puzzles, and procedural tasks. As expected, in most cases, the number of words required to teach the task is reduced when visual goal demonstrations are used. We also identify shortcomings of our approach and outline future research.

Introduction

As collaborative robots become ever more prevalent, it will be increasingly important for non-expert human users to be able to adapt and extend their behaviors without explicit programming. Interactive Task Learning (ITL; Laird, 2014) is a general approach where robots learn new tasks in real time through natural interactions with humans. ITL extends and unifies techniques such as learning from demonstration (Argall, Chernova, Veloso, & Browning, 2009) and learning from natural language instruction. In prior research (Mohan, Kirk, Mininger, & Laird, 2012), language instruction

* Corresponding author.
E-mail addresses: jrkirk@umich.edu (J. Kirk), mininger@umich.edu (A. Mininger), laird@umich.edu (J. Laird).

http://dx.doi.org/10.1016/j.bica.2016.08.001
2212-683X/

Please cite this article in press as: Kirk, J et al., Learning task goals interactively with visual demonstrations ..., *Biologically Inspired Cognitive Architectures* (2016), http://dx.doi.org/10.1016/j.bica.2016.08.001

proved to be effective for communicating novel tasks, allowing an instructor to describe the fundamental knowledge required to define a task: the available actions, goals, constraints on solutions, hierarchical task structures, and procedures for solving the problem. However, we have observed that humans use a wide range of techniques to teach and learn, and that language is not always the easiest or least cumbersome way to teach certain concepts.

Observing human-human teaching scenarios is informative as to what interactions an agent should support to facilitate natural, efficient, and accessible communication. Although people are great at learning from experience, much of how humans learn (or teach) is through explicit guided instruction with an expert human, particularly in settings when collaboration is necessary. From game learning to job training, many tasks are taught through grounded real-time instructions in situated shared environments, which is the context for our research. We have observed that when teaching a game to another person, humans employ a wide array of strategies and techniques to communicate information. Some examples include giving demonstrations, asking questions, using analogies, and making inferences. Similarly, when Kaochar et al. had people teach a simulated UAV agent how to carry out a mission and gave them multiple modes of instructions, the majority used all three that were available: teaching by example, demonstration, and reinforcement (Kaochar et al., 2011).

In particular, we are interested in the different ways that goals are taught and how an agent can use these strategies to learn goal representations. Even when the agent is taught a procedure, having an internal representation of the goal is useful when verifying that a task was successfully completed. A general goal representation also enables the agent to perform internal search and generalize to new variations of the task. We hypothesize that the instruction can be more efficient in some cases if the agent can learn the goal through non-verbal modalities. This hypothesis is explored and evaluated on Rosie, an Interactive Task Learning tabletop robotic agent implemented in the Soar cognitive architecture (Laird, 2012). Interactive task learning provides a challenge for cognitive architectures because it requires a general, end-to-end, task-independent framework for learning all types of task knowledge, without any task-specific programming by a human. It also requires diverse types of knowledge and reasoning that span topics from computer science and cognitive modeling, and requires integration of many fields: natural language processing, vision, machine learning, logical reasoning, knowledge management, etc.

We demonstrate and evaluate these approaches within the context of learning novel puzzles, games, and procedural tasks. We focus on how our approaches use different characteristics of these tasks to learn goals using a *single* example. Our empirical evaluation focuses on two desiderata of ITL systems: generality and communication efficiency. We evaluate generality by using this approach on three puzzles, one game, and five procedural tasks. We evaluate communication efficiency by measuring the number of words required to specify a task by the instructor and comparing strategies for teaching goals: one where the goal is described and one where the goal is demonstrated.

Background

Rosie is a robotic agent that uses a Microsoft Kinect sensor and robotic arm affixed to a tabletop, and is controlled by procedural, declarative, and episodic knowledge encoded in Soar. The agent can detect and manipulate foam blocks of various shapes, colors, and sizes. The human teacher interacts with the agent through a chat window or using Google's TextToSpeech services for speech production and CMU PocketSphinx for speech recognition. An image of the agent and chat window can be seen in Fig. 1.

The agent exists in a mixed-reality environment, where perceptions are augmented with additional information to make the environment more complex. First, the Kinect produces a RGBD point cloud from which points belonging to the table and arm are removed. The remaining points are partitioned using a union-find algorithm where points are joined if they are close in physical space and similar in color. Partitions with a significant number of points form the set of objects, and features for color, shape, and size are extracted. These features are then used to train a KNN classifier to produce a set of perceptual labels for the object (e.g. *red*, *triangle*). These labels, along with the position and bounding box of the object, are provided as input to the Soar agent. In addition, the agent is also provided with a list of named regions on the table to serve as locations (e.g. the pantry or stove). These locations can have simulated properties and dynamics. For example, the pantry has a simulated door which can be opened and closed. Internally, the agent represents and treats locations

Fig. 1 Picture of the robot arm and tabletop block environment. Also shown is the chat interface with the instructions needed to describe the goal for Eight Puzzle using a description.

Please cite this article in press as: Kirk, J et al., Learning task goals interactively with visual demonstrations ..., *Biologically Inspired Cognitive Architectures* (2016), http://dx.doi.org/10.1016/j.bica.2016.08.001

the same as other perceived objects. This augmented reality adds significant complexity to the environment and allows the agent to do a wider variety of tasks than is possible using just the robot arm.

The agent maintains a separate symbolic representation of the world in its working memory, which it updates as new perceptual information comes in. This world state representation consists of a set of objects $\{o_1, o_2, \ldots, o_n\}$ and predicates over those objects. Unary predicates defined over a single object describe properties of that object. Common ones include *red*(o_i), *square*(o_i), *block*(o_i), *location*(o_i), *grabbed*(o_i), and *closed*(o_i). Binary predicates defined over two objects describe spatial relations between those objects, for example, *on*(o_i, o_j) and *right-of*(o_i, o_j). All relational predicates are represented symbolically, but are grounded in continuous representations maintained in the agent's spatial/visual short-term memory.

There also exist a set of discrete actions the agent can perform. The agent sends each command to its output buffer where it is interpreted and executed by the motor system. Some commands, such as closing a door, are simulated. Commands involving the arm are executed using a simple arm controller which follows a prescribed set of scripted actions. Examples of actions include *open*(o_i), *pick-up*(o_i), and *put-down*(o_i, x, y). For this basic set of actions the agent knows the preconditions for each actions and can model each action during internal simulation (e.g. the result of *pick-up*(o_i) is adding *grabbed*(o_i)).

Natural language is processed using a parser, implemented in Soar, that is integrated with Rosie. The parser takes as input grammatical English sentences and produces a semantic interpretation that includes grounding all references to objects and locations to their referents in Rosie's perceptual system. The agent is initialized with general knowledge about how to interact with an instructor via language, how to learn procedural and declarative knowledge, and how to solve problems through internal search. The agent initially knows only simple primitive actions such as picking up a block or opening a door. In earlier work, we showed that through interactive language-based instruction, it can learn new verbs (such as *store* and *cook*), nouns (such as *rectangle* and *triangle*), adjectives (such as *big* and *red*), prepositions (such as *behind* and *on*) (Mohan et al., 2012), hierarchical tasks (Mohan & Laird, 2014), and games (Kirk & Laird, 2014).

In our original work, the instructor explicitly described all the objects, properties, and relations that were constituent parts of the goal directly via natural language. However, we discovered that this is cumbersome for goals that have many objects and relations. For example, describing the goal of the Eight Puzzle requires 69 words in our system (see Fig. 1). In this paper, we explore methods where the agent learns a general goal representation from a single example for games (both multiplayer games and puzzles) and procedural tasks. The challenge is determining which state features (objects, properties, and relations) define the goal, while ignoring those state features that just happen to be true in that specific instance. There is a crucial distinction between games and procedural tasks, which leads to two different methods for extracting the goal from the environment. Games add additional artificial constraints to the possible actions, which can be circumvented when demonstrating the goal, whereas the constraints for procedural tasks are inherent to the environment itself and thus cannot be circumvented.

For games, often the instructor can provide an explicit example of the goal in the environment without actually having to solve it. For example, in the Tower of Hanoi puzzle, the agent is forbidden from moving a larger disk onto a smaller one or putting a disk on the table (in our version we use blocks). However, it is easy for the instructor to quickly construct an example of the goal by violating the rules of the puzzle. Similarly, in Tic-Tac-Toe, the instructor can show an example of three-in-a-row without playing the game. In our approach, the agent compares this example state to its memory of the initial state to generate an initial representation of the defining characteristics of the goal, which is then refined through interaction with the instructor.

For procedural tasks, the instructor cannot shortcut performing the task – all of the intermediate actions must be performed. Therefore, in order to achieve the goal, either the instructor or the agent must go through the actions. By having the agent perform the actions itself, it can learn both the goal and the policy at the same time. When the goal is reached, not only does the agent have an example of the goal, but it also has additional information about what actions were performed. We use knowledge of these actions to help determine which aspects of the final state should be included in the goal without the need for further instructions.

Learning goals of games

Previous work has demonstrated that Rosie can learn over ten different games and puzzles, from Tic-Tac-Toe to Sokoban, through natural language descriptions of the legal actions and goals of the task. Here we describe how to learn these goals from demonstrations and evaluate our claims on four tasks: Tic-Tac-Toe, Tower of Hanoi, the Eight Puzzle, and the Frog and Toads Puzzle. In each case, the actual task solved in our tabletop block domain is an isomorphism of the classic game. For example, the tile sliding Eight Puzzle isomorphism uses colored blocks and colored locations rather than numbered tiles.

Often setting up the goal can be easier than describing it with explicit language. When given a demonstration of the goal, Rosie gets a snapshot of all the objects and predicates that make up the state. The problem of determining the goal representation is therefore a feature selection problem, where Rosie must determine what objects and predicates are essential to the goal so that a correct general representation can be learned. Including unnecessary features will cause over-specific representations, while excluding relevant features will lead to over general representations. We have developed heuristics to help estimate the relevant features, but in general it is hard to ensure correct feature selection, especially in one example. The final step of the learning process is to directly engage the teacher to make final adjustments to the goal hypothesis and verify correctness.

Please cite this article in press as: Kirk, J et al., Learning task goals interactively with visual demonstrations ..., *Biologically Inspired Cognitive Architectures* (2016), http://dx.doi.org/10.1016/j.bica.2016.08.001

The first heuristic used is that only predicates referenced during the task instruction should be included in the goal. Rosie knows a large number of predicates, including ones for colors, shapes, sizes, and spatial relations. Considering every possible predicate would lead to a large number of extraneous predicates being included in the goal. Predicates mentioned during instruction are more likely to be relevant. During a task, Rosie does not extract every possible feature or relationship it knows. Instead, when a predicate, such as *on*, is mentioned Rosie begins to extract instances of that relationship in the world. This provides a heuristic estimate of which relationships are relevant — those that have been used or taught during the current teaching session — essentially the predicates used to describe the task so far.

The second heuristic is that aspects of the final state which differ from the initial are likely to be important. Thus the differences between the two states are used to create an initial estimation of the goal. This process is formally described in Algorithm 1. This algorithm takes in the initial state and the final state where the goal is demonstrated. The first step of the algorithm, lines 2–3, extracts predicates from the initial and final states. Only the instances of predicates that are in the final state and not in the initial state are added to the goal representation. The next step (line 5) finds all objects O referenced in those predicates.

Algorithm 1. Goal estimation

1: **procedure estimate-goal**(state s_i, state s_f)
2: $P_i \leftarrow$ **predicates**(s_i)
3: $P_f \leftarrow$ **predicates**(s_f)
4: $P \leftarrow P_f - P_i$
5: $O \leftarrow$ **objects**(P)
6: **for** $(p_j \in P_f)$ **do**
7: $A \leftarrow$ **objects**(p_j)
8: **if** $(A \subseteq O)$ **then**
9: $P \leftarrow P + p_j$
return state(P, O)

The last heuristic used to estimate the goal is that the goal should also contain predicates which haven't changed between the two states but involve objects which have. Most unary predicates, like *blue* or *block*, do not change between the initial and final states, but can be necessary to distinguish objects in the goal representation. This is the reason for explicitly collecting the set of all objects: to consider other relationships that exist between objects already in the goal. Specifically, the algorithm (lines 6–9) includes all other predicates from the final state that are defined over those objects, even predicates that have not changed. This step includes information about the goal objects and their associated predicates that existed in both the initial and final states, but could still be relevant to the goal.

This process provides an estimation, but is not always sufficient for learning the correct goal. The assumptions behind the heuristics may be violated, leading to relevant predicates being ignored or irrelevant predicates being included. For example, the first heuristic is wrong when predicates are used elsewhere in the task which should not be included in the goal. The second heuristic is wrong when there are additional changes in the world that weren't relevant to the goal. For example, if the final state demonstration in Tic-Tac-Toe also contains opponent pieces on the board, those will be incorrectly included in the goal representation.

Since this initial estimation will often be wrong, it is important for the instructor to be able to refine the goal representation through further interactions. To do this, the instructor must acquire knowledge of the agent's goal hypothesis. Predicting the mistakes the agent may have made is difficult, especially if the teacher is unaware of the current relationships known to the agent. To facilitate corrections, as soon as Rosie creates a goal hypothesis, it communicates the representation in simple English, listing each predicate relation between the objects. This ability to expose internal knowledge representations is critical to meaningful communication between a teacher and student, where each has a model of what the other knows.

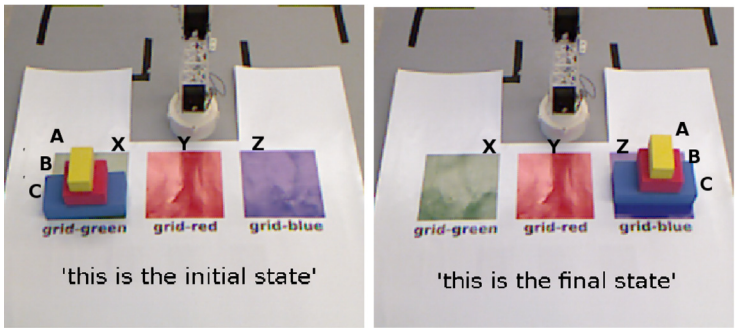
With this knowledge, the teacher can make changes to the goal representation by specifying predicates or objects to either pay attention to (*attend to*) or to *ignore*. When modifying the goal hypothesis, predicates are referenced by their name. Specific instances such as $on(B, C)$ cannot be ignored or added. Sets of objects can be referenced (*attend to the green blocks*) as well as single objects (*ignore this block*). Thus ignoring or attending to predicates is a fairly coarse refinement, while ignoring or attending to specific objects allows for more fine grain modifications. This approach is sufficient to correctly refine the goal representations of all the tasks explored in the work so far. In the future we will consider adding support for ignoring specific instances of predicates (*ignore that the blue block is on a location*), although we expect explicit goal descriptions to be more efficient in these cases. Ignored and attended objects and predicates are added and removed respectively from the objects O and predicates P returned from the `estimate-goal` algorithm. Additionally predicates are removed from P if they refer to objects no longer in O and objects are removed if no predicates refer to them. The final steps of `estimate-goal` that add additional final state predicates are repeated to add potentially relevant relationships between the new attended objects.

Tower of Hanoi example

The learning of the Tower of Hanoi puzzle by Rosie is a good example to illustrate the goal learning process. Fig. 2 shows the initial and final state demonstrations for Tower of Hanoi in our tabletop environment with the extracted predicate relationships below and the learned representations to the right. Fig. 3 displays the accompanying interactions between the teacher and Rosie.

The predicates that change from the initial to final state are highlighted in bold. These form the set P in line 4 of the `estimate-goal` algorithm: $P = \{on(C, Z), below(Z, C)\}$. Then the unchanged predicates $blue(Z)$, $location(Z)$, $large(C)$, $block(C)$ and $larger(Z, C)$ are added because they are defined over objects in the set O (lines 6–9). This forms the initial hypothesis shown in Fig. 2. Note that even though

Please cite this article in press as: Kirk, J et al., Learning task goals interactively with visual demonstrations ..., *Biologically Inspired Cognitive Architectures* (2016), http://dx.doi.org/10.1016/j.bica.2016.08.001



small(A),medium(B),large(C),blue(Z)
on(A,B),on(B,C),on(C,X)
below(B,A),below(C,B),below(X,C)
location:{X,Y,Z} block:{A,B,C}
larger(B,A),larger(Z,C) ...

small(A),medium(B),large(C),blue(Z)
on(A,B),on(B,C),**on(C,Z)**
below(B,A),below(C,B),**below(Z,C)**
location:{X,Y,Z} block:{A,B,C}
larger(B,A),larger(Z,C) ...

Initial Hypothesis:
on(C,Z), below(Z,C)
blue(Z), large(C), block(C)
location(Z), larger(Z,C)
Intermediate Hypothesis:
on(C,Z), block(C), large(C)
location(Z), blue(Z)
Learned Goal:
on(C,Z), on(B,C), on(A,B)
blue(Z), location(Z),
block:{A,B,C}
small(A), medium(B), large(C)

Fig. 2 Initial and final state demonstrations for teaching the goal for Tower of Hanoi. The predicate relations are between the blocks (A,B,C) and locations (X,Y,Z). The locations are labeled as grid-green, grid-red, and grid-blue. The predicate relationships extracted for each pictured state are shown below the images. The initial, intermediate, and final learned goal representations are on the right.

The name of the goal is all-stacked.
Please describe or demonstrate the goal.
This is the initial state.
Okay, please setup the goal.
This is the final state.
I think that the goal is that the large block is on the blue location and the blue location is below the large block and the blue location is larger than the large block.
Ignore below.
Ignore larger-than.
Attend to the blocks.
Done.
Ok, I've learned that the goal is that a small block is on a medium block and the medium block is on a large block and the large block is on a blue location.

Fig. 3 Transcript of the interactions to demonstrate the goal of the Tower of Hanoi. Rosie's responses are italicized.

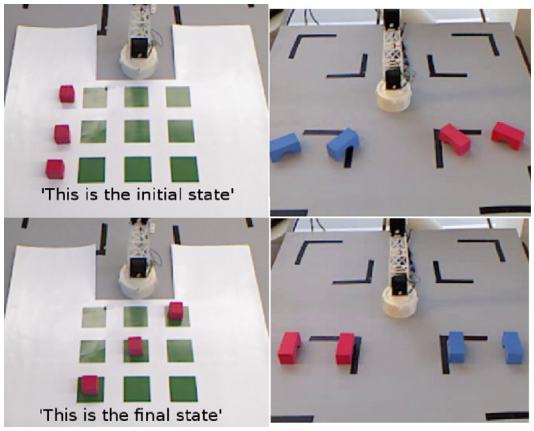
the goal should contain blocks *A* and *B* and predicates *on*(A, B) and *on*(B, C), because they are in both states they are not included. Additionally *on* and *below* are both included in the goal representation even though they are redundant. Once the goal estimate is generated, Rosie then describes this goal hypothesis to the instructor (Fig. 3).

The teacher should notice the two missing relations, the *medium block* is on the *large block* and the *small block* is on the *medium block*, and the two unnecessary relations, *below* and *larger-than*. The teacher first instructs the agent to *ignore below*, which is redundant, and *ignore larger-than*, which is unnecessary. This leads to the intermediate hypothesis, which contains only the relevant features *on*, *large*, *location*, *block*, and *blue*. Finally the teacher gives the instruction *attend to the blocks* in order to correct for the lost relevant objects. Attending to these objects causes the agent to also add the *on* relationships for the two blocks that did not change from the initial state. After these interactions, Rosie reports the modified and now correct goal hypothesis, "*I think the goal is that a small block is on a medium block and the medium block is on a large block and the large block is on a blue location.*" The final representation learned is also displayed in Fig. 2. If the final state demonstration for Tic-Tac-Toe contains blue opponent pieces, the teacher would instruct *ignore the blue blocks* in order to prevent learning an overly-specific goal description.

Evaluation

Our hypothesis is that the ability to substitute goal demonstrations for language descriptions reduces the amount of time and effort required to teach a task. To evaluate this claim, we ran experiments on four games, Tic-Tac-Toe, Tower of Hanoi, the Eight Puzzle, and the Frog and Toad puzzle. For all games, after goal refinements, the agent learns the correct goal representation, which is neither too specific nor too general. Fig. 4 shows the final state representation that Rosie learns for each game. Because the goal representation is relational and symbolic, it generalizes effectively to other goal instances and is easy to verify for correctness. For example, for Tic-Tac-Toe, given only one demonstration of three in a row, Rosie determined that the predicate *linear*(l_1, l_2, l_3) is part of the representation, which generalizes to all instances of the goal. This is also an illustrative case of why the final heuristic is useful; even though the relationships between the locations do not

Please cite this article in press as: Kirk, J et al., Learning task goals interactively with visual demonstrations ..., *Biologically Inspired Cognitive Architectures* (2016), http://dx.doi.org/10.1016/j.bica.2016.08.001



Tic-Tac-Toe:Three-in-a-row
$red : \{o_1, o_2, o_3\}$,
$on(o_1, l_1), on(o_2, l_2), on(o_3, l_3), linear(l_1, l_2, l_3)$
Eight Puzzle: All-Matched
$red : \{o_1, l_1\}, blue : \{o_2, l_2\}, green : \{o_3, l_3\} \ldots$
$on(o_1, l_1), on(o_2, l_2), on(o_3, l_3) \ldots$
Tower of Hanoi: All-Stacked
$small(o_1), medium(o_2), large(o_3), blue(l_3)$
$on(o_1, o_2), on(o_2, o_3), on(o_3, l_3)$
Frog Puzzle: Frogs-Swapped
$red : \{o_1, l_1, o_2, l_2\}, blue : \{o_3, o_4, l_3, l_4\}$,
$on(o_1, l_1), on(o_2, l_2), on(o_3, l_3), on(o_4, l_4)$

Fig. 4 On the right the states demonstrated for Tic-Tac-Toe and Frog puzzle are displayed. The final learned goal representations for each of the four evaluated games is to the left. The unary predicates are listed first, omitting *block*(o_n) and *location*(l_n), which would be included for every game. The relational predicates are listed second, which are defined over the blocks o_n and locations l_n.

change, *linear* is added to the initial goal hypothesis because each location is already in the goal representation.

Our analysis focuses on the number of words needed to teach just the goal of the game with and without the use of demonstrations. This metric is a stand in for efficiency as it does not capture the added mental effort and time required to construct goal descriptions or the physical effort and time required to manipulate the world environment for goal demonstrations. However for these tasks, the mental and physical effort is minimal. In the future, we plan to more directly measure efficiency, using overall teaching time.

There are many different, equivalent ways to describe the goals through natural language. In this evaluation, the goals were specified as efficiently as possible using our language processing system. The number of words includes only those uttered by the teacher. The results of these experiments are shown in Fig. 5. As expected, goal demonstration decreased the number of words required to teach the games. Tower of Hanoi shows the least benefit, because the goal is fairly easy to describe and the demonstration requires many additional interactions to correct the initial goal hypothesis. The Eight puzzle shows the most benefit, as the supported language for teaching the goals is verbose (see Fig. 1). Some goals can be learned without additional interaction, such as in Tic-Tac-Toe, although redundant below predicates will be included unless explicitly ignored.

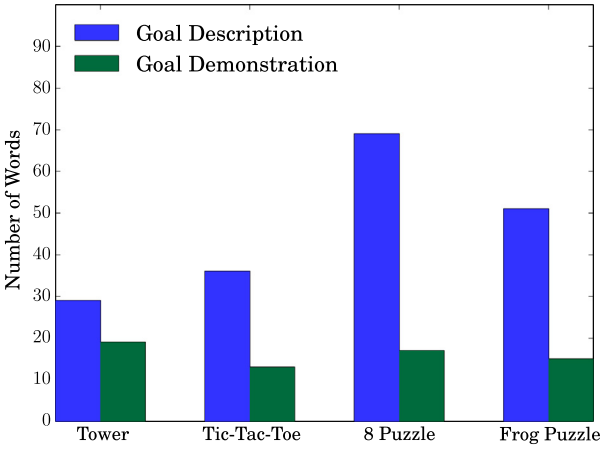


Fig. 5 An analysis of the number of words used to teach goals with descriptions and demonstration.

These results illustrate the potential benefit of goal demonstrations. The benefit of using goal demonstration increases as the size of goal representation increases. For example, in the 15 Puzzle and the 10 block variant of Tower of Hanoi, the interactions to teach using goal demonstration would be the same as their simpler versions, except for the additional blocks that the person would need to move to demonstrate the goal. All the attend and ignore instructions would remain the same. However, the language descriptions would increase linearly with the size of the problems. This is partially a byproduct of the limitations of the language system. For example utterances such as *all blocks are on locations with matching colors* (only 8 words) that contain qualifiers like *all* and task-specific abstract predicates like *matching* cannot currently be processed, although this is a venue of current research. Using abstract concepts and qualifiers allows for extremely efficient descriptions such as *all the blocks are stacked from smallest to largest on the goal* (12 words) or *that the red blocks are swapped with the blue blocks* (10 words). However, for a fair comparison the words necessary to teach the abstract predicates, like *matching* and *stacked*, should also be counted, especially as they are often task and domain specific.

Rather than evaluating the efficiency by the number of words, it would be better to measure the amount of time used to teach. This would help account for the mental effort needed to construct the goal representation when describing it explicitly with language, as well as the effort to move objects and setup goals when giving demonstrations. This would allow us to better analyze the trade offs between demonstrating and describing goals, but will require more extensive user studies. However, it seems clear that humans prefer mixed strategies (using description and demonstration) and that Interactive Task Learning agents should support both kinds of goal learning.

Please cite this article in press as: Kirk, J et al., Learning task goals interactively with visual demonstrations ..., *Biologically Inspired Cognitive Architectures* (2016), http://dx.doi.org/10.1016/j.bica.2016.08.001

Learning goals of procedural tasks

In addition to games, Rosie learns procedural tasks that involve executing a sequence of unrestricted actions to achieve a desired goal. An example of teaching the *store* task is shown in Fig. 6. In our original approach, the instructor provided a linguistic description of the goal and the agent internally searched for a sequence of actions that would lead to the goal. This search was depth limited; if it failed, the human instructor would provide the action that the agent should next perform to make progress to the goal. This process was repeated until the agent achieved the goal. While this is effective, it is often more natural or efficient for the instructor to omit the goal description and instead give just the step-by-step instructions to solve the problem. At that point, the instructor tells the agent the task is finished, and the agent attempts to generate the goal by analyzing the actions it took to carry out the task.

The `generate-procedural-goal`(s, A) in Algorithm 2 describes this process formally. This algorithm requires the initial state s_i when the task started and the final state s_f when the task was completed (represented as a set of objects and predicates), as well as a list of actions A it performed to accomplish the task. In addition, the agent must have models of its primitive actions and be able to internally simulate the effect of an action on a state, by knowing what predicates to add and remove (lines 5–6). To generate the goal, the agent simulates applying each action to the state (line 7), and accumulates the changes as a set of predicates P (lines 5–6). Of those predicates, those that exist in the final state (line 8) form an instance of the goal representation. The agent then tries to generalize this goal representation so that it can apply to different tasks involving different objects. It does this by seeing if any objects or predicates in the goal were also mentioned in the task description. If they were, it turns them into variables which take on the same value as the corresponding item in the task description. For example, the goal of *stack(red-block, blue-block)* is *on(red-block, blue-block)*, but it should not create the same exact goal for the task *stack(green-block, yellow-block)*. Instead, the objects in the goal should match whatever values they are in the task description.

Store the red block
What is the goal?
The goal is that the red block is in the pantry and the pantry is closed
What do I do next?
Open the pantry
(*Rosie opens the pantry*)
What do I do next?
Pick up the red block
(*Rosie picks up the red block*)
(*Rosie puts the red block in the pantry*)
(*Rosie closes the pantry*)

Fig. 6 Instructions given to teach the store task (Rosie's responses are in italics). The goal description is in bold. The last two actions are done without the need for instruction because Rosie found a solution through search.

Algorithm 2. Goal extraction after an example procedure

1: **procedure generate-procedural-goal**(state s_i, state s_f, actions A)
2: $P \leftarrow \{\}$
3: $s \leftarrow s_i$
4: **for** $(a_i \in A)$ **do**
5: $P \leftarrow P \bigcup$ **added-predicates**(s, a_i)
6: $P \leftarrow P -$ **removed-predicates**(s, a_i)
7: $s \leftarrow$ **simulate**(s_i, a_i)
8: $P \leftarrow P \cap$ **predicates**(s_f)
9: $O \leftarrow$ **objects**(P)
10: return $\{P, O\}$

As an example, consider the task "*store the red block*". Below is the sequence of actions given by the instructor and the predicates that are added and removed.

1. **Open the pantry.**
   +open(pantry)
2. **Pick up the red block.**
   +grabbed(red-block)
3. **Put the red block in the pantry.**
   −grabbed(red-block), +in(red-block, pantry)
4. **Close the pantry.**
   −open(pantry), +closed(pantry)

After these predicates are accumulated, the final result is the set {*in(red-block, pantry), closed(pantry)*}. The red block appears as an argument in the task description, so it becomes a variable that matches the task argument. The pantry and the predicates *on* and *closed* are not in the task description, so they are not generalized. Thus the final generalized goal for *store(A)* is {*in(A, pantry), closed(pantry)*}.

This approach makes two main assumptions. The first is that all of the actions taken by the agent are directly relevant to the goal. Thus we rely on the instructor to not give extraneous steps. However, in some cases this cannot be avoided. For example, when teaching "*move the red block to the table*," suppose the red block is in a closed pantry. The first step would be opening the pantry, so the predicate *open*$(pantry)$ would be included in the goal; making the goal over-specific. This would cause it to open the pantry during every move task, even those where the object does not start in the pantry. This is why it is important for the agent to be able to communicate its knowledge so that the instructor can correct it if it is in error. Our solution is to have the agent describe its hypothesis about the goal and allow the instructor to remove predicates, just as with games. In this example, the agent would say "*I think that the goal is that the red block is on the table and the pantry is open*," to which the instructor could reply "*Ignore the pantry.*"

The second assumption is that all aspects of the task's goal are captured by the agent's action models. This assumption breaks if there are dynamics in the environment which change the state in ways the agent does not predict. Thus we rely on the instructor to include all the information relevant to the goal in the action commands. For example, when teaching the *cook* task, the agent will place the steak on the stove, turn the stove on, and after five seconds the

Please cite this article in press as: Kirk, J et al., Learning task goals interactively with visual demonstrations ..., *Biologically Inspired Cognitive Architectures* (2016), http://dx.doi.org/10.1016/j.bica.2016.08.001

cooked(*steak*) predicate will be added. Since this wasn't the direct result of an action, this predicate will not be included in the goal as it is not added by an action model. However, if the instructor gives the additional action of "*Wait until the steak is cooked*," the agent will include the *cooked* predicate as part of the goal. Alternatively, as with games, the instructor could add the predicate after the fact by saying "*Attend to cooked*."

Since the agent has been given a full execution sequence, it may seem unnecessary to create a goal representation. However, our agent does not learn a fixed procedure, but learns a flexible state-based policy. If the agent tries to do the task in a different situation where the same procedure won't work, it can use the goal representation to search for an alternate way to achieve the goal. For example, if we taught the agent how to move a block to a location, then later tell it to move it to the pantry, the agent must first open the pantry. Even though this step was not encountered previously, because it has a goal description the agent can figure it out by searching over its actions to find a way to achieve the goal. Thus with a goal, it can adapt its behavior to new situations without needing additional instruction.

Evaluation

To evaluate our system, we compare the amount of instruction needed to teach new tasks with two types of instructional strategies. In the first strategy, a linguistic description of the goal is given to the agent. The agent can then try and search for a sequence of actions that will achieve the goal, or ask for help if it cannot find one. In the second strategy, no description of the goal is given. Instead the instructor gives the agent a sequence of actions to perform and then tells it when it has achieved the goal. An example of these two strategies being used in the *move* task is shown in Fig. 7.

Our environmental setup consists of a tabletop with four regions designated as locations — a table, garbage, pantry, and stove. These locations have simulated dynamics: the pantry has a door that is either *open* or *closed*, and a stove that is either *on* or *off*. The agent moves around real blocks and perceives their visual properties (color, shape, size), but simulates the dynamics of interacting with locations (e.g. it won't put a block on the pantry if the door is closed). The agent starts with learned knowledge about colors, shapes, sizes, and spatial relations. It also has an initial set of known primitive actions: *pick up, put down, turn on, turn off, open,* and *close*.

We teach the system five different tasks: *discard* an object by putting it in the garbage; *move* an object to a specified destination; *store* an object in the pantry and close the door; *cook* an object by putting it on the stove, turning the stove on, waiting until the steak is cooked (simulated property), and turning the stove off; and *set the table* by putting a blue square on the table, with a red rectangle to its right and a green rectangle to its left. For each test, we count the number of words needed to teach the task for each of the two strategies. For our evaluation, we assume perfect instruction. From these instructions, our agent always learns the correct policy and goal representation. Thus we are not comparing correctness, but efficiency of communication.

The first strategy (Goal Description) has the instructor give a linguistic description of the goal and provide additional instructions when needed. The agent can internally search for a sequence of actions that lead to the goal, so it may not need to ask for additional instructions. However, the search space grows exponentially with the depth, as it involves searching though all possible actions with all objects and spatial relations. So for tasks requiring many actions, the search can be very time consuming. Therefore, we limit the search depth to two. In our results we also show the best possible word count if the agent did an unlimited search (did not need to ask for additional instructions), and the worst possible word count if the agent could not do any search. The second strategy (No Goal Description) has the instructor just give the sequence of actions that lead to the goal, and the agent extracts the goal representation itself. In both strategies, the agent generates the same description of the goal.

The results are shown in Fig. 8. They illustrate a trade off when it comes to teaching goals. The tasks *discard* and *move* have simple goal descriptions (e.g., the goal of *discard(block)* is that the block is in the garbage) and have easy to find solutions. Thus for a depth two search, the agent requires no additional instructions and describing the goal ends up taking fewer words than giving the actions. But the *cook* and *set-table* tasks have complex goal descriptions (for *set-table* the goal is that the blue square is on the table and the red rectangle is right of the blue square and the green rectangle is left of the blue square) and so it takes fewer words for the instructor to tell the agent how to do the task and have the agent extract the goal itself. This trade off is also influenced by the search ability of the agent. If the agent cannot search, then even if it is given a description of the goal, it still needs to be given the full procedure and thus always takes more words (the blue light-shaded portion on the left is always the highest). Whereas if the agent has unlimited search it is always more efficient to give the goal description and have the agent search for the actions that lead to the goal (the blue dark-shaded portion on the left is always the lowest). However, doing an unlimited search is very time consuming so the

Goal Desc w/ No Search	Goal Desc w/ Depth 2 Search	No Goal Description
Move the blue box to the table. The goal is that the blue box is on the table. Pick up the blue box. Put the blue box on the table.	Move the blue box to the table. The goal is that the blue box is on the table.	Move the blue box to the table. Pick up the blue box. Put the blue box on the table. You are done.

Fig. 7 Example of different instruction strategies for the move task. Goal Description is shown when the agent does no internal search, and when it does a search up to two moves.

Please cite this article in press as: Kirk, J et al., Learning task goals interactively with visual demonstrations ..., *Biologically Inspired Cognitive Architectures* (2016), http://dx.doi.org/10.1016/j.bica.2016.08.001

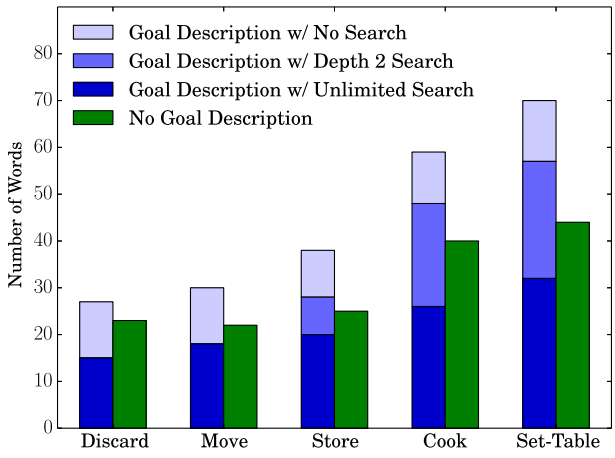


Fig. 8 Number of words needed to teach each task when the agent is given a goal description versus not given a description and having to extract it from the task execution. The word count of the goal description agent depends on its search capabilities. We show results with no search (worst), unlimited search (best), and search with depth limit of two.

instructor might prefer giving a few additional instructions to waiting on the agent to figure it out.

Related work

Although there has been a variety of research on task learning, it invariably depends on learning from *multiple* examples. For example, Kaiser (2012) and Barbu, Narayanaswamy, and Siskind (2010) describe systems that extract the relevant goal-state predicates for games, including Tic-Tac-Toe, by viewing many iterations of game play. Not surprisingly, most learning from demonstration systems (Argall et al., 2009; Chao, Cakmak, & Thomaz, 2011) for robotic tasks teach action sequences and not goal descriptions. Moreover they rely on multiple examples, and rarely have the ability to take in language to aid in learning. However, Nicolescu and Mataric (2003) describe a robotic agent that first learns from an instructive demonstration of the task and then refines its knowledge over multiple trials and feedback from the teacher, but does not learn goal descriptions. One learning from demonstration system that does learn goals is by Akgun and Thomaz (2015), where instructors not only give action keyframes but also goal keyframes during the same demonstration.

Closely related to our work, is the work of Hinrichs and Forbus (2014), which describes a computer agent that learns to play Tic-Tac-Toe embedded in CogSketch from interactions with a human teacher. The teacher can give goal demonstrations through sketching during game play. However the agent does not learn procedural tasks, and it does not do any feature selection reasoning. When distractor objects or relationships are present, the teacher must manually select the subset of objects that define the goal in the sketch window. An alternative approach to learning goals is to learn a reward function from demonstrations, known as Inverse Reinforcement Learning (Abbeel & Ng, 2004). Of course, this too requires many examples.

Discussion

In this paper, we have shown how it is possible to extend task learning from instruction so that a goal description can be learned from a single goal demonstration. Our results show that for some tasks, goal demonstrations provide an efficient alternative to language descriptions.

Interactive Task Learning, grounded in real-time scenarios in a robotic cognitive architecture provides a great platform for studying different strategies of situated interactive learning/teaching as well as many related cognitive tasks. We are interested in studying problems in how to unify knowledge from many sources: natural language processing, vision systems, logical reasoning, search, knowledge bases, etc., that cross diverse fields in computer science and cognitive modeling. This fits well with the short and long-term goals of cognitive architectures like Soar: to jointly develop intelligent agents that can learn and solve complex tasks and to study what the necessary mechanisms might be to support these activities and ultimately general cognition.

There are limitations to our approach, which define our agenda for future work. The agent can only learn goals that are defined by a final state, excluding ongoing tasks that involve continually performing a procedure, such as patrolling a building. Also, the agent cannot currently learn a goal with disjunctive predicates, where there are multiple variations of the goal. Another potential goal refinement strategy is providing many goal exemplars to determine which relationships are relevant. In addition, it cannot learn to include the negation of predicates in a goal description solely from demonstration. For example, it cannot learn that a goal is achieved if a red block is *not* on a blue block. Negated relationships can, however, be described in the descriptions of games. These could potentially be learned from demonstration by observing what predicates in the initial state no longer exist. A further restriction is that once a goal description is learned, the teacher cannot freely switch between the different teaching methods to refine knowledge, a capability we have observed in human-to-human task learning. In addition, the current language comprehension system is restrictive and often limits the efficiency of communication. Research is being conducted in parallel to expand the language capabilities of the agent so that the teacher can more naturally describe desired states and reduce the number of words necessary.

Even with these limitations, this work takes important steps in making Interactive Task Learning agents more efficient and more accessible by exploring a new method for acquiring goals and facilitating interactions to refine those states.

Acknowledgments

The work described here was supported by the National Science Foundation under Grant No. 1419590 and the Office of Naval Research under Grant No. N00014-08-1-0099. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressly or implied, of the NSF, ONR, or the U.S. Government.

References

Abbeel, P., & Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on machine learning* (p. 1). ACM.

Akgun, B., & Thomaz, A. (2015). Simultaneously learning actions and goals from demonstration. *Autonomous Robots*, 1–17.

Argall, B., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*.

Barbu, A., Narayanaswamy, S., & Siskind, J. M. (2010). Learning physically-instantiated game play through visual observation. In *2010 IEEE international conference on robotics and automation (ICRA)* (pp. 1879–1886). IEEE.

Chao, C., Cakmak, M., & Thomaz, A. (2011). Towards grounding concepts for transfer in goal learning from demonstration. In *Proceedings of the international conference on development and learning*.

Hinrichs, T. R., & Forbus, K. D. (2014). X goes first: Teaching simple games through multimodal interaction. *Advances in Cognitive Systems, 3*, 31–46.

Kaiser, L. (2012). Learning games from videos guided by descriptive complexity. In *AAAI*.

Kaochar, T., Peralta, R. T., Morrison, C. T., Fasel, I. R., Walsh, T. J., & Cohen, P. R. (2011). Towards understanding how humans teach robots. In *User modeling, adaption and personalization* (pp. 347–352). Springer.

Kirk, J. R., & Laird, J. (2014). Interactive task learning for simple games. *Advances in Cognitive Systems, 3*, 13–30.

Laird, J. E. (2012). *The soar cognitive architecture*. MIT Press.

Laird, J. E. (2014). *Report of the NSF-funded workshop on task ability revised title: Interactive task learning*.

Mohan, S., Kirk, J. R., Mininger, A., & Laird, J. (2012). Acquiring grounded representations of words with situated interactive instruction. *Advances in Cognitive Systems, 2*, 113–130.

Mohan, S., & Laird, J. E. (2014). Learning goal-oriented hierarchical tasks from situated interactive instruction. In *Proceedings of the twenty eighth AAAI conference on artificial intelligence*. AAAI Press.

Nicolescu, M., & Mataric, M. (2003). Natural methods for robot task learning: Instructive demonstrations, generalization and practice. In *Proceedings of the second international conference on autonomous agents and multi-agent systems* (pp. 241–248).

Please cite this article in press as: Kirk, J et al., Learning task goals interactively with visual demonstrations ..., *Biologically Inspired Cognitive Architectures* (2016), http://dx.doi.org/10.1016/j.bica.2016.08.001