Knowledge

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

3

Functional Knowledge Requirements for Interactive Task Learning

Robert E. Wray III, Niels A. Taatgen, Christian Lebiere,
Katerina Pastra, Peter Pirolli, Paul S. Rosenbloom,
Matthias Scheutz, Terrence C. Stewart, and Janet Wiles

Abstract

What knowledge needs to be learned to acquire a novel task? What background knowledge does an agent need to use newly acquired knowledge effectively? This chapter considers the functional roles of knowledge in task learning. These roles of knowledge span interaction with other entities and the environment and core functional capabilities of the reasoning system itself (i.e., architecture). Perspectives are offered on the definition of "task" and the relationship between task and knowledge. In addition, three specific challenges central to the role of knowledge in interactive task learning (ITL) are examined: the identification of architectural primitives (basic functional and representational building blocks) needed for ITL, requirements for enabling shared understanding ("common ground") between learner and instructor, and conditions that support projection and anticipation of future states. In conclusion, specific research questions are put forth to address these challenges and advance ITL as a field of inquiry.

Introduction

What knowledge needs to be learned to acquire a novel task? What background knowledge does an agent need to use newly acquired knowledge effectively?

Group photos (top left to bottom right) Robert Wray, Niels Taatgen, Katerina Pastra, Peter Pirolli, Janet Wiles, Paul Rosenbloom, Terry Stewart, Matthias Scheutz, Christian Lebiere, Janet Wiles, Robert Wray, Niels Taatgen, Katerina Pastra, Paul Rosenbloom, Terry Stewart, Janet Wiles, Peter Pirolli, Christian Lebiere, Robert Wray, Matthias Scheutz, Niels Taatgen

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

Answering these questions requires some consideration of the functional roles of knowledge in task learning. These roles include interaction with other entities, interaction with the environment, and the core functional capabilities of the reasoning system itself (i.e., the agent architecture).

This chapter offers an introductory consideration of these functional issues. We begin by defining tasks, elaborating on the relationship of task and knowledge, and then suggesting a formulation of the interactive task learning (ITL) challenge that emphasizes the role of knowledge. We focus primarily on a human teaching an artificial agent a new task but also consider other configurations of learner and instructor (for a general treatment of the topic of task instruction, see Shah et al., this volume). Thereafter we consider three goals or challenges which we regard as fundamental for understanding the functional role of knowledge in ITL:

1. *Identify the basic building blocks of ITL functionality, both computational and representational.* Can the field come to understand which building blocks are required? Is it reasonable or feasible to assume that such building blocks can be found?
2. *Enable the development of a common, shared understanding during interaction* (i.e., common ground) (Clark and Brennan 1991). Several chapters in this volume (e.g., Mitchell et al., Levinson, Chai et al., and Thomaz et al.) emphasize the importance of achieving common ground during interaction. Here, we explore the computational implications of this requirement, both in terms of "background knowledge" (general knowledge of the world that learner and instructor bring to the interaction) and dynamic shared understanding within the instructional interaction.
3. *Support rapid, pervasive anticipation and prediction of future states.* Both interaction and learning benefit functionally from the ability to anticipate (or to "predict") future states. What are the expected benefits of prediction capabilities for ITL? What are the requirements for prediction within the context of ITL systems?

ITL represents a scientific challenge that can only be addressed by multiple researchers, from diverse areas of expertise, over many years. How might we better identify common goals and work together to achieve them? First, we must acknowledge and capture some of the diverse perspectives on the goals of ITL, before considering community perspectives on the endeavor and the potential for shared tools. We recommend that common, shared learning tasks would aid improved communication and sharing of results, and we outline the notion of challenge problems for the community, including one specific example. To this end, we summarize our analysis and present a number of high-level research challenges for the community of ITL researchers.

The Nature of Tasks for Interactive Task Learning

The breadth and depth of tasks that one could ask a human or agent to perform are considerable. This raises the question of whether there is a general definition of "task" that can encompass this breadth and depth, or whether "task" is a cluster concept where instances only bear a family resemblance (in deference to Wittgenstein). Cluster concepts have clear (noncontroversial) instances, clear non-instances, and a large set of border cases where even experts will not agree on whether they ought to belong to a given class. For the concept of "task," composing and sending out a set of customized emails about an upcoming event is a clear instance. Eye-blinking is a clear non-instance. Here, we attempt to define "task" in two different ways. First, we outline a conceptual definition of "task," with the caveat that the proposed definition is incomplete and may raise objections; still, we hope that it will help the research community understand which kinds of new learner behaviors are targeted for an ITL system. Second, we provide a list of specific and diverse examples of tasks.

What Is a Task?

Colloquially, modern human life is replete with tasks. We run errands to buy groceries and fuel. We cook dinner, set the table, and wash the dishes. We schedule meetings, take notes, write summaries. We dance, play a piano, and sing. We play football, shoot basketball, run for fitness. We play cards and games. We tend lawns and gardens, monitor for plumbing leaks, and repair those leaks when we find them. We move boxes and cartons, plan recreational activities, and balance a banking account. We call or text a friend. We help our young children dress and eat. We surprise our partners with some token of our affection.

Because so much of human activity can be characterized as tasks, we can ask: What is not a task? Is recognizing a face a task? Is daydreaming a task? Is earning an undergraduate degree a task? Is ocean surfing a task?

Table 3.1 lists the primary characteristics that we feel are central to the notion of "task" in ITL. This characterization is not definitive. It is meant to convey the direction of current focus in research, as well as the ambition to move to tasks that are more complex, more meaningful to the public (outside our research laboratories), and more broadly inclusive of what it may mean to learn a new task or to instruct another agent in learning a task.

The properties of tasks listed in Table 3.1 allow us to consider the questions introduced earlier. Is daydreaming a task? The answer would be "no," because there is no (deliberate) goal for daydreaming; that is, daydreaming may have functional benefits in some cases, but the essence of daydreaming is typically a wandering away from more purposeful behavior or thought. Similarly, facial recognition, by itself, is not a task. Facial recognition in humans can be considered a single-step process at the cognitive level that occurs very quickly

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

Table 3.1 Characteristics of tasks for interactive task learning.

Dimension	Description
Task goal(s)	A state of the world that the agent should achieve, maintain (homeostatic goal), or perform as a result of the task. Task goals can be seemingly simple (pick up a block) or complex (continually monitor a space station for leaks); they can also condition or modify the performance of the task (move with control and grace).
Multiple steps	A task generally requires that the agent perform a series of separable actions under its own control. In some cases, repeated task performance might lead to more succinct representations of behavior (Laird et al. 1986; Mitchell et al. 1986; Taatgen and Lee 2003) and could result in single-step execution, even if the task is learned as a multistep process.
Temporal bounds	Human-scale tasks are generally ones that are performed on a timescale of minutes to hours.
Instructible tasks	A human must be able to articulate/express how the task should be executed, verbally or nonverbally (e.g., instructions might include demonstrations). This dimension does not imply that a human is necessarily able to perform the task.

(Bruce and Young 1986). It would be difficult for a human to articulate how one should recognize faces in comparison to other objects. However, there are very simple tasks that could include facial recognition as a component: finding and choosing smiling faces in an image would be a task according to the characteristics enumerated in Table 3.1. Earning an undergraduate degree is not a task, according to the listed characteristics, because of the lengthy timescale required to achieve this task. However, there are many individual tasks that would be important to the pursuit of a degree, such as taking class notes or performing algebraic manipulations while solving an equation in differential calculus.

Some tasks can be described as a skill that involves, for example, dexterity (e.g., bipedal walking, drumming, or swinging a tennis racquet to hit a ball). Such tasks are largely continuous in nature and hard to analyze in terms of symbolic complexity. Such skills/tasks can play the role of primitives in more complex tasks, as they have an instrumental role in structuring other tasks. However, the distinction between skills and tasks of definable symbolic complexity is fuzzy. In essence, there is a paradox between task and skill: the more familiar or adept a learner becomes with a task, the less prominent the corresponding task structure knowledge becomes in the learner's memory when executing the task: the task then becomes a "skill" for the learner. For example, when a young adult is learning to drive, the many steps associated with successfully controlling a vehicle are explicitly in mind and consciously attended to during driving. A person who has been driving for many years, however, will likely not regard these individual steps at all. Unless the learner needs to teach or describe the task to someone else, the symbolic task structure used during the learning phase is a piece of task knowledge that is not usually activated in routine task performance. Thus, the "complex" skill can now play the role of the primitive in a task of higher complexity. "Driving to the market" can be considered a single step in the execution of the task to buy groceries at the market.

According to the conception of "task" proposed here, surfing is a task (comprised of individual skills): It has a goal (riding a wave). There are multiple steps in the process of surfing (e.g., identifying a "good" wave, moving from prone to standing, guiding the board along the wave, maintaining balance). A single instance of surfing takes place over a few minutes. Surfing instructors can make a living by teaching others to surf.

The characteristics in Table 3.1 not only help us characterize the concept "task," they also call out what may need to be learned when an agent and instructor undertake ITL. A surfing instructor can describe the goals of surfing, help the learner break down wave-riding into a series of largely decomposable steps, and guide the learner in practice. Although the timescale of performance in surfing is only a few minutes, learning to surf may require many days or weeks of practice, some of which is guided by the instructor. The important observation is that the timescale of learning to perform a task at a high level of proficiency (e.g., consider surfing or playing chess) may be several orders of magnitude larger than the time required to perform the task once.

Practitioners within this emergent research domain have diverse and sometimes divergent perspectives on the nature of ITL and the goals and priorities of research within the field. Mitchell et al. (this volume) provide a high-level definition with which others (including some of the authors of this chapter) may not fully agree. For example, some of us assert that a key characteristic of ITL is that the instructor agent has a goal to increase (make more efficient) the speed of learning on the part of the learner agent. This more strongly emphasizes the role of the instructor in ITL than the definition offered by Mitchell et al. This disagreement about edge cases (i.e., whether they should or should not count as instances of some concept) is the hallmark of a "cluster concept" outlined above.

Although disagreement remains about what constitutes ITL at its boundaries, there is also broad agreement about its core. The notion that tasks have goals is a guiding idea to classify activities that may or may not be tasks (e.g., recognizing particular faces). Moreover, in general, we have chosen in this chapter to exclude task-learning scenarios where there is little ongoing interaction between the teacher and the learner (e.g., single instructions or instructional videos): we regard "interaction" as an exchange between two agents, rather than one-way communication (additionally, regardless of who these agents are, simply interacting with the environment is not sufficient). These definitions exclude learning contexts where the learner acquires competence in performing a task from unsupervised interactions with a task environment, which is a feature of some recent advances in machine learning, such as the ability of Deep Mind to learn to play Atari video games (Mnih et al. 2015).

Finally, we acknowledge that a discussion of the functional knowledge requirements for ITL results necessarily in imprecision, as is inherent to all central concepts involved in ITL. “Task,” “learning,” and “interaction” as well as concepts involved in spelling out those concepts (e.g., “knowledge”) are themselves cluster concepts.

Examples of Domains and Tasks

To explore the notion of “task” for ITL further, we outline specific examples of ITL which the field could and should attempt to address. We deliberately include examples of tasks that are “atypical” to those currently at the center of most ITL research. Our intent is to suggest that the span of tasks which ITL should be able to address is broader than the array of tasks the field is currently addressing. In some cases, suggested tasks will introduce new requirements for ITL capabilities. For each learning task, we also introduce a domain of usage in which the specific task to be learned could be introduced, with the goal of highlighting why the learner’s performance of that task could not simply be “programmed” into the original performance system.

Function Composition in User Interfaces

Personal assistants that can automate the tedium of administrative tasks—scheduling meetings (Allen et al. 2007), making travel arrangements (Knoblock 2004), completing “paperwork”—have been a long-time goal of researchers in both artificial intelligence (AI) and human–computer interfaces (Lehman and Carbonell 1989). Such personal assistants may have powerful built-in functionality, but to be maximally useful in everyday contexts, they must be able to learn to perform specific tasks required by the individual user. Today’s existing personal assistants (e.g., Apple Siri, Google Now, Amazon Alexa) are limited to preprogrammed tasks (e.g., enter calendar entry). The devices in which these assistants are embedded, however, have the needed infrastructure already in place (e.g., voice commands, application programming interfaces to apps) on most target platforms to extend their capabilities toward task learning and instruction.

ITL could be the means by which a user instructs a personal electronic device (e.g., computer, tablet, smart phone) to perform automatically increasingly complex sequences of actions to achieve high-level user goals (e.g., fill out an expense report from stack of receipts). Such functionality is already being explored in everyday tasks, such as ordering coffee (Azaria et al. 2016). Such task learning offers tremendous potential benefits, from increased work productivity to enabling access to support services for those with cognitive impairments.

From “Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,”
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

The interaction for function composition could be as follows: Users give a verbal description of a target task as they perform it, associating the various components and arguments with the actions performed and values entered. Alternatively, the system starts by being able to perform basic actions (e.g., typical application menu commands, keyboard entry) from voice commands and is instructed step-by-step how to perform more complex ones.

Call Centers and Field Linguistics: Learning to Interact Socially in New Cultural Contexts

Increasing the efficiency of call centers can provide significant cost savings (Gray et al. 1993). Customer satisfaction often declines when automation replaces human operators in call centers, leading to caller frustration and anger. Rather than canned, automated responses, it would be preferable to have call center agents that can adapt to the customers' needs, to the affect that they have on the customer, and to the social context of the caller.

The rudiments for this type of capability might be able to be programmed in advance. However, since cues and dynamics of interaction differ according to social and cultural context, it would be advantageous to have an ITL system that could conduct basic diagnostics and remediation, adapting when necessary to the specific cultural (and even subcultural) cues so as to customize the response to the caller. This approach would benefit the call center by enabling greater scalability (one core system customizable to many different sociocultural contexts) and mitigate the problem of cultural and gender bias that has been observed in some AI systems (Caliskan et al. 2017).

The social interaction learning task would involve an instructor and a learner (an agent), tasked with learning to respond appropriately to specific cultural cues of future callers. The learner would need to learn such things as turn-taking and interruption strategies, theory of mind of the entity with which it is interacting, and how to respond appropriately to affective states or displays. One of the advantages of ITL over preprogramming is that the agent would receive ongoing feedback and thus improve its ability to recognize and to adapt to specific caller cues (a kind of "on the job" training). It is not intended for the learner to learn how to understand or generate the content of the communications, other than that it may need to learn to extract cues from the content for aid in making social decisions.

Another use case for such social learning could be robots and virtual agents being developed to support field linguists. The task of a field linguist begins with segmentation and discovery of the sound system of a new language. What are the distinctions that make a difference to meaning? These can include perceptual distinctions that are not initially in the linguist's repertoire. The linguist needs to discover how sounds are combined in flexible ways into wordlike units, centered on word stems. The structure is inherently combinatorial, with the task being the discovery of alternates for roles and fillers at multiple levels,

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

including identification of word stems and the possible suffixes, affixes, and infixes. The knowledge representation in a language comprises multiple levels, including phonemes (sounds units), morphemes (units of meaning), words, phrases, sentences, and syntax as well as embodied actions such as pointing and gaze.

Evaluation of competence is revealed through transcription of corpora. Much of the task of the field linguist involves the transcription of recorded language material. For ITL, sets of corpora could form a challenge domain with a range of tasks for human and agent learners, which include construction of learning examples and other tasks that reveal the sources of variability in how information is communicated, multilevel structure, and discovery of slot-and-filler syntax in multiscale structures.

More generally, although there are thousands of human natural languages in use today, only several hundred are supported through current information technology tools. An agent designed to act as a surrogate or aid for language study by a field linguist would also benefit from being able to be quickly customized to specific cultural and social patterns of interaction. While it may be possible to encode an expert system that could support the dynamics and nuance of interaction for a particular language, this approach is infeasible due to the funding and technical expertise that would be required. A mature ITL solution that allowed an expert in the language to teach its aid rapidly and without technical know-how would scale to the diverse needs of the thousands of human languages.

Robotic Assistants: Fast Customization for Emerging Tasks

Robotic systems offer significant capability for performing tasks that are repetitive, dangerous, or beyond human capability. While robotic systems may have much inherent capability "out of the box," their function will likely need to be modified or extended, either to amortize the cost of the robot under changing task performance requirements or because the specific behavior in a performance environment can only be finalized or specialized *in situ*. Consider the following three examples of ITL in the robotic domain:

1. *Industrial manufacturing*: For any assembly task, we would like to instruct the robot how to manipulate parts to assemble the prescribed structure (e.g., a piece of furniture, a car, an airplane wing): how to pick up parts; how to insert them into, connect, or mount them onto other parts; how to place and reorient parts, and so on. An ITL robot could be instructed or receive demonstrations to support changing assembly requirements.
2. *Disaster site monitoring*, including routine tasks such as counting containers (e.g., containers of nuclear waste), recording their ID tags, and performing gamma measurements on some of them. The International

Atomic Energy Agency is currently seeking to identify small robotized rolling platforms capable of assisting the human inspector by performing the following tasks: moving autonomously across a storage area, counting items of a specific geometry, recording their ID tags, and carrying specific instrument payloads. The agency has issued an open challenge for a robot that is partially autonomous (i.e., navigation), with the remaining functions to be teleoperated. This basic robotic platform could be enhanced with ITL to enable the robot to learn specific measures needed at a particular site (e.g., where to go to take measurements, how to navigate the inside of structures) and to provide a more autonomous monitoring capability.

3. *NASA's space robotics challenge* identified three tasks aimed at simulating what a robot may be required to do while assisting a NASA mission to Mars, whether in a preparatory capacity before astronauts arrive, or alongside astronauts: (a) aligning a communications array; (b) repairing a broken solar array; and (c) identifying and repairing a habitat leak. All three tasks could be instructed using natural language, including the details of what an object looks like. Currently, this challenge does not require robots to be instructible.

Studying Human Task Learning to Advance ITL

Here we consider four examples of human task learning that might be apt subjects for advancing ITL: playing card games, tying a knot, dancing the Argentine tango, and playing BrainQuest. These examples of human task learning appear relatively straightforward. They can be taught by humans to other humans without requiring the instructor to have a high level of task proficiency or training in instruction. Currently, these tasks present challenges (to lesser and greater degrees) to existing artificial (nonhuman) agents. Thus, they are useful in identifying directions that future research may wish to take.

Playing card games. An agent needs to play a new card game, which can include a software simulation with high-level sensors and actuators (e.g., moving cards, flipping them, reading them). Some aspects of the new card game may rely on preexisting concepts (e.g., tricks, piles, cards-in-your-hand, discards) and actions (e.g., play a card, draw a card) whereas others may be new. As the agent is introduced to a new game, it may need to construct both novel task operators and new terms that describe or summarize task states (e.g., a "good hand"). Once the agent understands the basic rules, it might also be given other instructions, such as suggested strategies or indications as to what "good" moves are. Thus, learning a new card game may involve the use of existing knowledge to support the acquisition of new task knowledge. These are core requirements in ITL (discussed further below).

Tying a knot. This task can be used to test a number of aspects of ITL due to the range of task complexity. One could begin with an easy knot (an overhand

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

knot) and proceed to increasingly complex knots, such as bowline or highly complex decorative knots. In addition, knot tying involves a hierarchy of task primitives: one type of knot learned previously is part of the tying of a more complex one and actually forms a necessary step of it (i.e., there is dependency). Currently, robots exist that have learned to tie different types of knots based on direct demonstration or manipulation (Schulman et al. 2016). These robots offer direct comparisons of the benefits of interactive instruction and generality of the resulting agent capabilities.

Knot tying presents challenges for verbalization (verbal instruction) and thus provides an opportunity to test different instruction modalities and their effects on learning (e.g., use of analogies in language, use of visual demonstration alone, or combinations of verbalization and demonstration). A basic requirement for instruction is the establishment of common ground (e.g., the "tail of the rope" vs. the "standing edge," the notion of a "loop"); term creation is needed. Knot tying is a highly interactive task: the learner needs immediate feedback while performing the task, and it is crucial to know at which part of the process things went wrong. Knot tying would also stress the temporal constraints and social aspects of the interaction: the learner may easily become frustrated and the teacher may find it challenging to communicate the same information in different, more effective ways.

Knot tying requires relatively little background knowledge. It is an activity that involves learners of any age, including children; thus one can test this task developmentally using age range or level of expertise (expert vs. novices). It is an integral part of many different activities in everyday life, such as tying one's shoelaces, getting dressed in a suit, and connecting two items with rope (e.g., in sailing, climbing, fishing, scouting, rescue activities, knitting/creative decorations, medical practice/surgery). Today's robots may lack the necessary dexterity to support near-term, direct reproduction of human instruction. However, simulations of knot tying could be developed to explore what would be required for a robot to master this task.

Argentine tango. Human adults can learn new tasks in a variety of ways, from taking a class at a community college, to online videos or engaging a teacher or coach directly. Learning to perform a new physical skill is an example of one of these tasks. As a particularly challenging example, consider the task of a physically embodied agent learning to dance the tango with a human (or other agent) as its partner. This task requires being taught the basic moves and sequences, and then performing those moves with a partner. When dancing with a partner, there will likely be a lot of nonverbal instruction (or perhaps reinforcement learning) to get the more subtle aspects of the movements that are dependent on the partner. However, the robot could also be verbally instructed about the sorts of nonverbal cues that the leading partner will present to indicate the desired actions of the following partner. One interesting requirement for performing the tango is the aesthetics of the dance: a combination of grace, fluidity, and power characterize a good tango. Thus, learning this task requires performing the dance with these requirements in addition to the more basic expectations of rhythmic steps and stances.

BrainQuest. Children learn new tasks from teachers, parents, and other children. For ITL research, various developmental stages of children offer the opportunity to focus on tasks that require less sophisticated and rich sources of knowledge, thus providing greater opportunities for a cognitive modeling environment to allow the architecture to "show through." As an example, consider BrainQuest, a game that consists of cards with a question on it, typically accompanied by a picture. There are cards for 3- to 4-year-old children, for 5- to 6-year olds, as well as older children. Each card introduces a little task in itself. Take, for example, a picture of a mother duck with five ducklings and a chicken with six chicks. The question posed on the card is: "Which mother bird has six babies?" Another example is a picture of a lion, a dog, and a cat with the question: "I bark and I like to go for walks. Which one am I?" There is overlap in knowledge with other tasks, but the task is almost never exactly the same.

Examples of Desired Features for ITL Research Tasks

In the course of considering a range of tasks for ITL, we identified a number of task features that an agent should learn. These features are not all in agreement with one another. Thus, in our listing of desiderata for ITL research tasks, we summarize some of the rationales that one might make for including a particular task feature:

- *Interactive tasks*: Tasks that require interaction in performance (as well as in learning) are of special interest to some researchers. Interactive task domains (e.g., conversational personal agents such as the help center, robots that support child learning through games and play, tango dancing) represent domains that are currently very difficult to engineer effectively (for a discussion of language interaction requirements, see Levinson, this volume). Using ITL to teach effective interactions (as in the help center and tango dancing domains) would benefit the larger AI community because of the difficulty of providing effective interactions in today's systems.
- *Generalization and transfer*: Researchers in AI and cognitive science, especially cognitive architecture (Laird et al. 2017), are interested in exploring the generality of an agent system and the ability to perform successfully across a wide range of tasks (Anderson et al. 2004; Newell 1990). Card playing (as highlighted above) provides a good test of generality within a limited sphere, as would the task domains proposed for general game playing (Genesereth and Thielscher 2014).
- "*Complex enough*" *domains*: Toy domains are attractive because the field of ITL is relatively immature and there are many potential barriers to entry. However, toy domains may not offer sufficient complexity,

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

and the lack of complexity may not always be apparent when the domain is chosen. Thus, there is a tension between wanting to work on domains that matter (below) and domains that are not overwhelming. One potential challenge for the field would be to provide systematic characterization of the complexity of the task learning and performance domains, to enable more informed and deliberate selection of research domains.

- *Domains that* "*matter*": Developing an ITL research system will require a significant investment and be conducted over many years. Because of this, it may be beneficial to focus on tasks which "matter"; that is, domains (and tasks within them) that offer direct benefit to the larger world, if the research led to field-testable prototypes and field-capable systems. A good example is the assistant for a field linguist, as summarized above, which could play an important role in helping to preserve endangered languages. Another example is ITL agents and robots for disaster response assistance, which is a domain that requires extraordinary flexibility and *in situ* adaptivity. It is important to note that what "matters" is subjective and that researchers may differ in their assessments. Domains that matter need not necessarily be highly complex domains that require huge investments. Stakeholders within each respective domain's community may help clarify important requirements for ITL research.

Computational, Representational, and Task Primitives for ITL

In AI, an agent's behavior and capability is typically assumed to be a function of its architecture operating on its store of knowledge while interacting in an environment (Russell and Norvig 1995). In this context, "architecture" refers to a fixed set of computational operations and representational elements used to build an intelligent agent (Newell 1990).

Cognitive science and AI have produced many different types of agent architectures, with various representational and processing assumptions. Thus, the same type of functionality (e.g., ITL) can be realized in different ways in different architectures. For instance, while most cognitive architectures may require explicit task representations in declarative or procedural memories that can be assembled through learning processes, robotic architectures might represent the task only as an action policy that implicitly represents its tasks in the form of state-action pairs (i.e., mappings from state descriptions to actions that will allow the agent to reach new states, resulting over time in the agent's performance of the task). Rich (this volume) and Scheutz et al. (2013) discuss these distinctions further.

Agent architectures are blueprints for computers: machines capable of universal computation. Computation must be grounded in primitives that are executable. For standard computer architectures, computational primitives form an instruction set that enables execution of complex programs. The composition of representational primitives (bits and bytes) and computational primitives (move, store, add) provide sufficient mechanisms for the creation of complex and sophisticated computer applications. Some combinations of these primitives are sufficiently useful such that they can be organized into a stable and reusable higher level of abstraction above the most primitive level. An operating system defines a set of primitives (files, typed numbers, strings, system calls such as "open," "load," and "execute") which can then be used to compose software programs that depend only on the operating system level of abstraction (e.g., a word processor, a mobile phone application). For a cognitive or robotic architecture, primitives include at least the mechanisms that enable the representation and processing of knowledge and skills. For an artificial neural architecture, it includes the operations that compute outputs from inputs given information in the form of inputs and weights.

What elements comprise an effective architecture (abstract computational level) for pursuing ITL? Does it even make sense to pursue such a goal? Given the variety of architectures, it is important to take a "least commitment perspective" when discussing architectural requirements for ITL. As yet, there is no agreed upon set of common functional components across architectures for different types of agents (virtual and robot) or task domains (e.g., solving mathematical problems vs. performing assembly tasks) although there is longstanding and ongoing interest in such identification and unification (Laird et al. 2017; Newell 1990; Sloman and Scheutz 2002). In this section, we consider the architectural implications of ITL while not committing to specific architectural assumptions; instead, we focus on the notion of *knowledge and process abstraction* as well as the role of *primitives*. We consider four distinct issues related to computational abstractions for ITL:

- Do cognitive architectures offer a useful level of architectural abstraction to pursue ITL?
- Is a fixed level of abstraction for ITL a reasonable goal?
- How might the mixed modalities of interaction within ITL shape requirements for an architectural abstraction?
- How might diverse conceptions of "task" inform the goals of architectural abstraction?

Cognitive Architectures as Potential ITL Architectures

Cognitive architectures provide computational abstractions that are, to varying degrees, designed to realize a human-mind-like virtual machine. For ITL, we need to know whether these existing computational architectures are defined at a useful level of abstraction to support ITL. How do the primitives defined for some particular architecture map to the functional requirements for ITL?

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

Are existing primitives too low level, too cumbersome, and too tedious to support efficient pursuit of ITL? If so, one direction of research could be to define an ITL capability (a higher-level architecture or virtual machine) that "implements" ITL for some architecture. For example, in cognitive architectures such as ACT-R and Soar, the representation of a task would be distributed across multiple memories (semantic, procedural, and episodic). In an ITL virtual machine, a "task" might be a primitive representation that was then decomposed into the specific and distributed representations of the underlying architecture.

Conversely, it could be that primitives (or at least some primitives) of today's architectures are defined too coarsely to support ITL, thus making it difficult to realize ITL within a given architecture. In standard computer architectures, it may sometimes be necessary to decompose the primitives of an architecture into finer-grained elements (e.g., the primitives of an architecture at a lower level of abstraction). In most cases, an applications developer will work at the operating system level of abstraction, but may need to shift to the assembly-language level in some cases. The instruction set level of architecture may need to be described at the level of electrical current flows. Any computational architecture is grounded in lower-level computational or physical processes. Thus, we need to ask whether ITL requires a reconsideration of many of the "standard" or assumed primitives of computational cognitive architectures.

In Search of Reusable Levels of Abstraction for Task Learning

Within any computational architecture, computation ultimately builds from the execution of its computational primitives, upon which more complex programs can be built. For example, for a cognitive robotic architecture, action primitives would likely specify basic movement behaviors that enable the emergence of more complex ones; such primitives may be explicitly represented (the case in most symbolic representation architectures) or may implicitly emerge from computation (as in neural architectures). Different architectures commit to different computational and representational primitives. As described by Taatgen (this volume), the primitives of a high-level cognitive or robotics architecture can be created from the composition of primitive elements of another architecture defined at a lower level, analogous to the various architectures one finds defined for standard computation, as outlined in the previous section.

This perspective raises questions as to what sufficient and useful primitives are for ITL, and whether these primitives can be mapped to the primitives of existing computational architectures (as outlined above). Should concepts such as "task," "task step or action," or "task state" map directly onto computational primitives for an ITL architecture? The identification and exploitation of the primitives of task representation and task learning is a basic research goal for computational and behavioral disciplines alike. These task primitives are essential in task learning. For example, if task primitives are defined at an

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

appropriate level of abstraction sufficient for representation and computation, task representations will more readily scale to tasks of arbitrary complexity and can be applied creatively when encountering unknown, uncertain, and noisy or ambiguous conditions.

Decomposition of processes, knowledge, and tasks into core nondecomposable units cannot be arbitrary if scalability and effectiveness in learning is of interest. However, the fundamental principles that may govern such decomposition are largely unexplored (Barto et al. 2013).

We assume that task primitives are not necessarily mapped to architectural primitives: the knowledge required to represent and to execute a task may need to be represented at different levels within an ITL system and the level at which task knowledge is represented impacts the composition of the knowledge and potential for reuse (for further discussion, see Taatgen, this volume). ITL increases the challenge because the representation of the task itself changes with task learning. For example, we may teach the learner a sequence of actions (“raise your arm; move your hand a short distance back and forth for a few seconds”) and then give that sequence of actions a particular name (“wave”) so that it can be referred to in the future by the more abstract label. Importantly, however, there may be times when the learner needs to consider how some task action is composed and further decompose it. For instance, when teaching the learner to reach for an object located above the learner, we may want to refer to a movement previously labeled “raising your arm,” but be more specific about which joints to use and their target angles.

Flexible use of various levels of abstraction extends to all parts of the system. The above example focuses on actions, but the same can be said of objects as well. Sometimes we may want to refer to a crowd of people, an individual person, a face, an eye, a circle, or combinations of these. Sometimes it is sufficient to refer to something as “the number 2” whereas at other times we need to be more specific (“a 2 with a loopy bottom”) or less specific (“a number”). Furthermore, the representation of time must be flexible as well, as we discuss further below. Actions may take place over milliseconds, seconds, minutes, hours, or longer periods of time. Actions, objects, and durations can also be combined at various levels of abstraction.

Flexible and adaptive abstraction affects communication between the learner and the tutor (these different levels of abstraction must be able to be mutually identified and understood), as well as the internal cognitive architectures of these systems. For example, the motor control system may be given high-level commands (“wave your right hand”) or low-level commands (“move this joint to this angle at this speed”). Perceptual categorization may provide objects at many hierarchical levels (e.g., “object,” “person,” “Jill,” “head,” “face,” “eye”). There is some lowest level of nondecomposable task atoms, but because we do not know what these lowest level primitives are, we cannot be assured that any particular architecture’s computational primitives are fixed at a sufficiently low-enough level to support the flexibility required by ITL.

From “Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,”
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

Experimental research in verbal categorization has shown that some verbally expressed categories of entities (and potentially actions, features, and processes of any kind) do exemplify properties that render them ideally informative for sensorimotor similarity-based generalization; for example, basic level concepts in prototype theory (Rosch 1973) conform to this property. Thus, language may point to useful levels of generalization from which the identification of task primitives may be founded to support both decomposition and synthesis. This basic level of generalization (which comprises categories that are neither too general nor too specific) could provide a common abstraction level for tasks of any type and complexity, serving as a representational ground across ITL systems. The fact that tutor–learner communication is primarily (though not exclusively) verbal makes the role of natural language in providing a basic, common abstraction representation level even more significant, and a promising direction for research.

The Role of Modality-Specific Abstraction

Abstraction of task knowledge and processes may be expressed through one or more modalities, each modality being more appropriate for different types and levels of abstraction according to its strengths and limitations. Consider the different modalities that may be used in teaching a robot to grasp and manipulate an object. Verbal representation of task knowledge is symbolic and high level: "Please place the fork next to the plate." On the other hand, sensorimotor representation of task knowledge may be more usefully represented at a subsymbolic level, such as a demonstration of the movement involved in placing the fork as described. Language is particularly suitable for expressing the goal of a task (i.e., the local or end goal; the final location of the fork). A sensorimotor representation would be limited to a description of the achieved state. These visuomotor modalities, however, are clearly more suitable for capturing the actual movements and physical relationships needed to achieve the goal of moving the fork to the desired location. Thus, linguistic and visuomotor primitives for task knowledge and execution need to be coordinated and aligned with one another during all phases of ITL, and these coordination requirements may impose additional constraints on the specific architectural abstraction needed for ITL.

Task Representation

Task descriptions in past and current research tend to be rigid and overly specific. Accomplishing a given task is usually precisely specified, with clear criteria for successful completion. Often there are equally straightforward procedures for accomplishing the task. While this may seem appropriate, and indeed it is useful in the short term, it tends to prevent cumulative progress in the medium and long term. The very task specificity that allows an ITL problem

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

to be solved precisely prevents it from being directly reused in other, related situations. Those situations tend to have a slightly different representation or contextualization, leading the original task solution to be unusable unless it is explicitly modified to accommodate the new specification.

While this specificity is similar to that of traditional programming paradigms, where the approach has been quite successful, it is quite unlike what Herbert Simon referred to as ill-defined problems characterizing natural intelligence, where their very nature requires the flexibility and adaptivity of human cognition (Simon 1996). Newell advanced cognitive architectures as the solution to the "20 questions" problem (Newell 1973b), which resulted from models of distinct cognitive tasks being developed in incompatible frameworks, thus preventing increasingly integrated models from being developed. However, while cognitive architectures have enabled the generation of compatible models of various tasks, they have not generally resulted in composition for increasingly complex cognitive capacities. In Soar, for example, natural language understanding and dynamic plan execution were demonstrated in an integrated agent (Lehman et al. 1995). That demonstration, however, did not result in these capabilities being routinely composable in future Soar models. Similar examples abound across cognitive architectures. This observation is not a criticism of these architectures but rather a caution for ITL: demonstration of significant integrated systems capability is not sufficient to define reusable and useful higher-level abstractions for ITL.

Co-Construction and Common Ground: Shared Contexts for ITL

In this section we explore what is required to achieve common ground (Clark and Brennan 1991) during learning interactions and what architectural primitives are required to support it. Our discussion here is not conclusive. Although we have some general agreement that architectures require mechanisms to support the achievement of common ground, conclusive demonstration of the necessity and sufficiency of those primitives requires development and testing by the emerging ITL community.

Common Ground in ITL

Agents, as well as their human instructors, must support functionality for establishing common ground in ITL. Borrowing from the notion of common ground in communication theory (Clark and Brennan 1991), this means that the shared context and mutual knowledge available to human and AI agents in a collaborative instructional situation have to be updated, maintained, and often repaired. This assumes that we reinterpret ITL as a form of joint action carried out in a coordinated manner by both the human and the AI agent.

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

This means coordination of the content (e.g., about the to-be-learned task) as well as the process by which the ITL activity progresses. In the ITL situation, common ground makes it possible for a learner and an instructor to coordinate on what the instructor means and what the learner understands the instructor to mean. As a joint activity, ITL implies bilateral processes: instructors must monitor the actions of the learner in context, and the AI learner must somehow signal their current state of learning.

Levinson (this volume) suggests that the functional requirements for common ground in human communication include:

- *Error checking at every level.* Participants continually monitor their understanding. Monitoring must be fast and accomplished at many levels (e.g., simultaneous monitoring of syntax and semantics). The fast execution of error checking and repair within the tempo of the conversation reflects predictive processes that anticipate errors or potential misunderstandings prior to their occurrence.
- *Continual turn-taking.* Participants engage in dynamic and complex turn-taking interactions that require little explicitly acknowledged cues of negotiation about the turn-taking. Such turn-taking includes shared contextual understanding of when interruption may or may not occur.
- *Mirroring of terminology.* As participants engage in dialogue and "co-construct" a shared understanding, they tend to begin using the same terms. There is reduced use of synonyms under conditions of language/capability mismatch.

Levinson argues that it may be impossible, with the current state of scientific and technical knowledge, to produce ITL systems that can reproduce the richness, subtlety, and timeliness of human–human communication. Instead, we may need to adopt engineering shortcuts that take advantage of the human tendency to anthropomorphize as a means of mitigating this issue.

In the field of human–robot interaction, the theory of common ground has been adopted widely and built upon, for example, in the approach of coactive design (Johnson et al. 2014). The idea is that coordinated tasks involving humans and robots require managing the interdependencies and constraints among the agents' activities. The common ground in such collaborative human–robot activity includes the relevant capacities (i.e., the total set of capabilities, knowledge, and resources) of the interdependent agents needed to perform the joint activity. Capacities are defined completely by the interaction of the agent and its environment—an idea similar to Newell's definition of knowledge-level descriptions (Newell 1982). To support common ground functionally in human–robot activity, coactive design (Johnson et al. 2014) promotes the development of interagent interfaces that support observability, predictability, and directability. These principles can be reinterpreted slightly to the ITL situation: *Observability* means the learner makes pertinent aspects of

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

its own state and knowledge of the instructional situation, target task, and environment observable to the instructor. *Predictability* means that the learner's actions are predictable so that the instructor can act appropriately. *Directability* is the ability of the instructor to direct the behavior of the learner, which in the specific situation of ITL means being able to direct the AI agent to engage in learning actions and processes.

"Global" and "Local" Common Ground

Despite different languages and cultural backgrounds, humans share a high level of commonality, at least in comparison to a human and a machine. One can think of common ground as the set of referents and knowledge shared between the learner and the teacher. These are internal representations that are "grounded" in "common" with one another. Clearly, these may not be perfectly identical; however, successful communication relies on them being sufficiently "aligned." Chai et al. (this volume) discuss the importance of background knowledge and comment that the lack of common ground is both a significant limitation of existing ITL robotics systems and a relatively underexplored research area.

We recommend the introduction of minor terminological distinctions to highlight these differences for future communication within the community. Specifically, we suggest a distinction between "local" and "global" common ground. Global common ground is shared background knowledge (or "common knowledge") which agents may have before the interaction starts. Some of this background knowledge may not be shared, and part of the interaction between teacher and learner may involve detecting and fixing misalignments in background knowledge. However, this sort of knowledge seems to be fairly distinct from the common ground that is local to the instruction of the task itself, for example, identifying which object one agent is referring to with "this cup" and knowledge about the actions and events taking place within the interaction.

Functional Requirements for Reaching Local Common Ground

Computational cognitive architectures have largely focused on identifying the components of thought and mind for individual cognition (Anderson et al. 2004; Kieras and Meyer 1997; Newell 1990). A "standard model" of the components comprising a cognitive architecture has been recently proposed (Laird et al. 2017). It attempts to identify common functional elements of these architectures, such as different kinds of memories (episodic, procedural, semantic) and the computational operations that occur within and between these memories to achieve cognition.

An open question is whether the computational decompositions reflected in these individual architectures (and in the emerging standard model) are

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

sufficient for ITL and for the ability of ITL systems to achieve common ground with human instructors. In other words, what are the architectural implications of ITL? If there are specific architectural capabilities that are required for ITL, can the community identify clear requirements, invariants, or "laws" that should be encapsulated by the architecture? How can these cognitive architectures support the functionality described by Levinson (this volume) or coactive design (Johnson et al. 2014)?

We have not yet reached firm conclusions regarding the sufficiency of existing architectures. However, we have identified the following architectural requirements for ITL, focusing especially on co-construction or achieving of shared understanding during ITL:

- The architecture should directly support the *acquisition of new task knowledge*. This requirement implies the acquisition of new procedural representations (e.g., routines, functions). It also includes many other representations as well, such as the ability to learn new procedures from the composition of previously learned procedures, to articulate (at some level) an internal model of one's procedural understanding, and to assess one's confidence of understanding and ability with task procedures. Rich (this volume) refers to these latter elements as *metaknowledge* of the task procedure. This "knowledge of what I can do" appears critical to co-construct shared understanding effectively as the learner grows in ability and knowledge of the task.
- Architectures should support *concept refinement*. Task concepts are not static in human learning and they should not be assumed to be in ITL, if artificial task-learning systems need to achieve shared understanding with human instructors. Introductory assumptions may need to be modified or extended (e.g., imagine introducing the "castle move" in chess sometime after the normal movements of the king and rook have been explained). Task actions or concepts, which may have been treated as unitary in the introduction of the task, may need to be further decomposed, resulting in the breakdown of apparent task primitives into more fine-grained primitives. Architectures must then support fast and flexible reformulation of these task concepts to support ongoing interaction and learning.
- Architectures should support *recognition and rapid response to realignment*. As discussed above, error checking is a pervasive component of the dynamic, shared understanding that occurs in human interaction. The pervasiveness of such capability strongly suggests that this would be an innate (or architectural) capability in humans. However, many existing architectures do not explicitly embed language processing at the architectural level. This raises an open question of what architectural functions in such architectures could give rise to such fast and pervasive low-level language processing capability.

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

Finding Common Ground with Nonhuman Intelligences

It is easy to fall into the trap of believing that the challenge of ITL is to provide AI agents with humanlike cognitive architectures so that such agents are instructible in a natural humanlike way. If we can just make robots perceive, act, think, learn, and talk like humans then our problems are solved. Although some agents may end up doing tasks in the same physical world as we humans, many will not. Current commercial conversational agents live primarily in a world of application software, with limited communication channels to human users and limited awareness of the physical environment. As roboticists are quick to tell us, humanoid robots do not "see" the world or execute their motor actions in the same way as humans. Agents in virtual environments typically rely on representations of space (and negative space) based on implementation representations rather than the spatial representation a human viewer might perceive.

To take an even more extreme example, imagine a long-range, long-lived, continuously flying, solar-powered drone that seeks to discover huge collections of plastics in the ocean. It may perceive the world through some combination of LIDAR (light detection and ranging), hyperspectral imaging, GPS, etc. and move about using rotors, wing foils, etc., utilizing flight controls and perceptual processing outside of human expertise and experience. It may be continuously learning using deep learning, reinforcement learning, or some hybrid combination of machine learning techniques that are difficult for humans to interpret. So the AI drone's underlying architecture to address its task environment may consist of fundamentally different perceptions, actions, and representations than humans. However, if we include ITL with humans as part of the drone's task environment, then that creates the need to interface the drone's internal representations with those of humans.

It may also be possible to jettison entirely the idea of having a dedicated task learning architecture that maps and executes onto the underlying cognitive architecture of the agent. The agent could support some general functionality for establishing common ground in the ITL environment. This would require the agent to be able to interact with instructors in ways that align common elements of task capabilities and knowledge to communicate about the task environment, those capabilities, and relevant knowledge.

Mental Models and Simulation for ITL

An ITL system will need to know how its environment will change as a result of its actions as well as through the actions of others. Such understanding is necessary to reason about the potential effects or outcomes of various choices, or to anticipate future states based on the actions of others. This sort of prediction has a long history in AI and cognitive science, but ITL systems may place unique demands on such a system. In particular, following novel instructions will, by necessity, require the agent to predict the future *under conditions it has never previously experienced*. A successful ITL agent will require an internal model of the world that is reliable enough to extrapolate to new conditions and still provide good predictions.

We use the term *mental simulation* to refer to the process of predicting some future state(s). Other common terms include prediction, projection, look-ahead, envisionment, simulation, and model-based reasoning. In all these cases, prediction involves imagining future states and actions. In general, systems capable of such mental simulation will have some form of *mental model*, which we take to be the internal representation that supports this process.

The Necessity of Mental Simulation

To clarify why we believe that mental simulation is a required core component for ITL, we first note that humans do this automatically in ITL situations. For example, imagine hearing verbal instructions for tying a knot (as discussed earlier). It is likely that you may imagine or attempt to visualize the manual operations described during the delivery of those instructions. The question of why this occurs then arises. As discussed below, the knowledge to predict the outcome of an action allows an agent to use instructions with incomplete sequences of actions, repair mistakes, and perform goal-directed planning.

The requirement to consider things that are not currently true in the world extends farther than just visualization based on a set of instructions. When humans perform such tasks, they also make use of hypotheticals, counterfactuals, and so on. That is, they can spontaneously consider various possible futures, not just the one that is likely to be arrived at based on following a novel set of instructions. This may be thought of as a kind of analogy over experience, which enables both predication and abduction, as well as generating more than one alternative when multiple prior experiences are retrieved (Forbus and Gentner 1997). Requirements for ITL may also implicate a continuous rather than a discrete future/timeline.

Importance of Context

Mental simulation is more useful if the learner can relate instructions to the world and reason about the consequences of carrying out the instructions. Instructions, however, are often given as a list of actions to be accomplished, whether in recipes, checklists in aviation, instructions for electronic devices, or instructions on how to connect to the office printer. The disadvantage of these types of instructions is that the purpose of each step is not always clear, making it much harder to simulate outcomes mentally. Humans are sometimes able to infer this knowledge, but without this knowledge, it is very hard to modify a procedure or to repurpose the knowledge for other tasks.

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

We find clear support for this statement in research on the flight management system (FMS) used in commercial aviation (Taatgen et al. 2008). An FMS is an onboard computer that can almost autonomously fly an airplane. To enter routes, change routes, and program the FMS for other tasks, pilots must learn list-based procedures. If pilots do not know what the purpose of the individual steps are in these procedures, they have difficulty memorizing them and generalizing the knowledge to novel situations, even though this is often required in everyday aviation. Taatgen et al. (2008) developed an alternative instruction: subjects were taught the context within which a particular step had to be understood, and the consequences of that step (i.e., a pre- and a postcondition). Compared to the traditional list instruction, subjects in the context condition learned much faster and were able to solve novel problems that required them to step outside of the bounds of the original procedures. These instructions can be supplemented with context, allowing pilots to perform mental simulation based on a richer mental model. Context allows pilots to reason about the outcomes of their actions, enabling them to compensate for missing knowledge and derive new sequences of actions for novel problems.

Imagination and Counterfactuals

Given that human task-based instructions are often underspecified and incomplete (e.g., details are left out, steps are skipped, some task knowledge is assumed), the learner is often required to fill in gaps which can be achieved using a variety of mechanisms, including imagining concrete task-based settings and applying instructions in the imagined environment, or constructing counterfactual scenarios to determine the extent to which an instruction is applicable. Counterfactuals obtained by making changes to the current state, which turn representations of the actual context into a hypothetical context, can serve multiple purposes in the learner's making sense of an instruction:

- They help the learner understand why the instructor gave the instruction in a particular way.
- They permit the learner to correct wrong assumptions about the nature of the task.
- They focus the learner's attention on relevant task aspects.
- They can suggest questions the learner might ask the instructor to clarify instructions.

In addition, counterfactuals may allow the learner to generalize and transfer knowledge to other cases (Wilson et al. 2016).

Machine learning for classification algorithms often requires careful calibration of the training set to balance stimulus dimensions and categories. Otherwise, undesirable biases such as frequency bias can result in skewed outcomes. Real-time learning mechanisms in human cognition, and by extension ITL, do not allow for such batch techniques to be applied. Counterfactual reasoning can be a flexible and efficient way to achieve similar goals (reducing biases that derive from initial sampling) in a more naturalistic way. For instance, consider a selection problem where one choice yields a deterministic payoff while the other follows a probabilistic distribution between higher and lower values, averaging to a somewhat higher payoff than the deterministic choice (i.e., the well-known two-armed bandit problem). If one chooses according to expected values established from past history of each choice, as typical in instance-based learning models of decision making (e.g., Thomson et al. 2015), a run of poor luck with the probabilistic choice will lead to a lower expectation, resulting in the deterministic payoff being chosen. Since no information is typically available for foregone payoffs, this suboptimal policy can persist indefinitely.

However, counterfactual reasoning can alleviate the tendency to fall into local minima. Engaging in an explicit consideration of alternative choice can lead to a plausible instance similar to the one actually experienced. That plausible, imagined instance can then result in a rebalancing of the choice distribution, future choices that take the alternative outcome, and, over time, normalizing of choice expectations. Counterfactual reasoning, though, requires at least simple causal models of the domain for it to be effective. Otherwise it can just as well result in a confirmation of the original choice, if the decision maker gives in to confirmation bias and similar self-confirming tendencies.

Discrete versus Continuous Prediction

In all the above considerations about the use of mental simulation for prediction, we have remained agnostic as to the type of prediction. Some mental models used for prediction are based on some sort of discrete time step. For instance, a mental simulation of navigating through a house might consist of having a current belief about what room one is in, and then imagining the effect of a discrete action, such as "go to the living room," to be a discrete change in location (perhaps via a series of navigation steps). While this is a common case, it does not seem to be the same type of mental simulation that is involved in the tying of a knot. For this case, mental simulation seems to be continuous in time, as are the imagined movements of one's hands and fingers.

Exactly how such continuous mental models are supported is not clear to us. To capture the timing and dynamics of prediction, it certainly appears that dynamical systems models and various kinds of neural networks may be more useful than symbolic or rule-based systems. However, the more important question may be how these systems are learned. In a situation based on discrete time steps, predicting the future is often thought of as learning the function that maps from the current state and action onto the next state. To support this in a continuous time domain, some sort of discretization may have to be used (e.g., predicting the state one second from now), but this imposes a particular time step that may not be appropriate. Research into qualitative representations provides a quantization of continuous behavior that is broadly compatible with human event decompositions in perception and language (Forbus 2011), and hence may be useful for ITL.

One way to address this may be to consider the approach observed in the human hippocampus (Rubin et al. 2014). Here, there seem to be two different ways of representing the future. One is to represent a sequence over time by actually having the mental model (in this case neural activity) change over time in a similar way (although perhaps much faster than the original action). This sort of neural replay uses time to represent time (although at a different scale). The second method is to represent the episode over time as a single static pattern. This might be thought of as using space to represent time, with the temporal sequence being collapsed into a single event. Particular parts within that sequence may be classified, resulting in something like a chunk (with slots for the different temporally ordered events within that sequence). We note, however, that different levels of discretization of the temporal sequence are possible.

The Research Community and Functional Knowledge Requirements

As the community seeks more definitive understanding in this new interdisciplinary field, there will be many different viewpoints on the goals, research practices, fundamental assumptions, and understanding of what is needed to move ITL research and development forward, and how we might work on common goals. Nonetheless, it is still possible to share methods, tools, and goals (in the form of challenge problems), as we outline below.

Options for Shared Methodologies, Tools, and Infrastructure

Shared methods and tools could be very helpful in removing barriers to entry in ITL research. Figure 3.1 elaborates on the conceptual framing of the ITL system from Mitchell et al. (this volume). It highlights seven areas where sharing could be beneficial and productive (indicated in the elliptical labels). While a comprehensive review of specific sharable components is outside the scope of this chapter, we do identify a few examples. We also recommend that the community establish a shared portal (e.g., website, Wiki, repository) where sharable components can be cataloged and stored for community use.

Architectures and Architectural Components

Architectures and architectural components may be part of the learning system, the instructional system (in the case of a synthetic agent as instructor), or both. Examples of architectures include cognitive architectures, such as ACT-R and Soar, as well as robotic frameworks, such as the Robotic Operating System.

From “Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,”
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

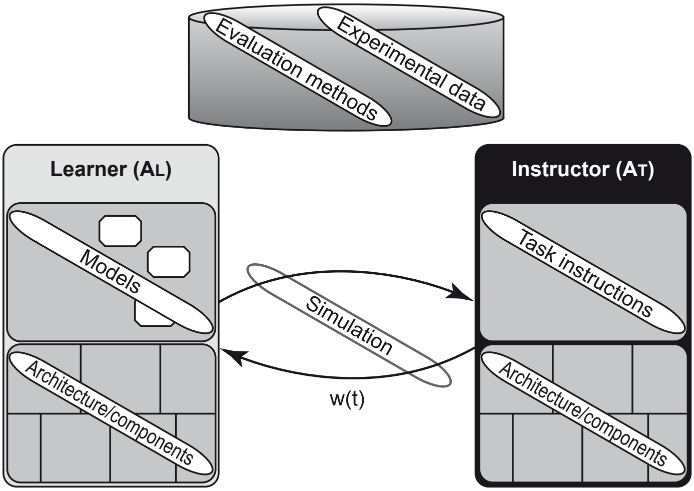

Figure 3.1 Conceptual framing of ITL: ellipses indicate computational artifacts and knowledge products that could be shared across the ITL research community.

Previously developed components may be especially useful for moving into ITL from other areas of research, such as in constructing a natural language understanding system. However, there are many existing components from which the capability for natural language understanding can be constructed. The specific components to choose (and the gaps that must be filled between them) depend on the level of sophistication and completeness required by the research.

Knowledge Components and Models

Knowledge components and models are elements expressed in some representation (typically, the representations of the underlying architectures or components). They are likely to be difficult to share directly unless different researchers are using the same architecture, but some functional descriptions could be shared. It may also be useful to share and exchange knowledge-level summaries of models and model components, even if the encoded representations are not directly sharable across researchers using different architectures and systems.

Simulations

Simulations are not required for ITL, but they may be convenient for speeding research and exploring options that are not cost-effective or safe to explore initially in physical settings. The sharing of simulations, such as Gazebo, has

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

been useful in the robotics community for comparable reasons. Simulations could fall into any or all of the following categories:

- Those which simplify the real world to support or enable task learning in a simulated environment (e.g., a robot simulation or a virtual machine simulation of a mobile device).
- Those designed to mimic the role of a player or actor in the task learning interaction (e.g., a simulated instructor who interacts with a learning agent).
- Those that are the target domains or environment for task learning (e.g., a virtual human character).

Instruction, Instructional Strategies, and Simulated Instructors

Examples of instructions which summarize (and possibly codify) effective human instruction could be useful for bootstrapping and enabling systematic exploration of alternative instructional strategies for evaluating ITL. For example, Koedinger's SimStudent (Harpstead et al. 2015; MacLellan et al. 2014; Matsuda et al. 2007) captures and encodes human instructional steps. SimStudent's representation might provide an initial template for codifying and sharing instructional activity generally for ITL systems. Further, in the educational technology community, there are ongoing attempts to encode various human instructional strategies for use in intelligent tutoring and computer-based training (Wray and Woods 2013). An instructional strategy could be the pattern and choice of specific demonstration examples or the conditional introduction of special cases after a student demonstrates competence in the more prototypical examples. Tools and representations such as these could support the definition and reuse of instructional strategies for ITL. VanLehn (this volume) offers the perspective that ITL instructional strategies may not need to be terribly sophisticated, because actual human instruction is often not as sophisticated as it is hypothesized to be.

Research Data and Artifacts

There are many different classes of methods and data that could be shared within the ITL community, including evaluation data, tools and methods for data analysis, and repositories of task specifications. As we discuss further below, currently there is a gap in the community's ability to specify tasks in a way that meets all requirements. Task specification languages (Yost 1992) have been developed in the past, and recent efforts in general game playing and planning have included generalized representations for some kinds of tasks (Love et al. 2008). A more general task specification language is, however, needed for effective sharing of task specifications.

Challenge Problems for ITL

Above we listed a collection of tasks to describe the scope of ITL. Here, we recommend specific tasks that are plausibly feasible in the near term for the community to work on and share progress. In particular, a few well-specified "challenge problems" could help focus the community. Defining such problems is difficult: we want the challenge problems to be limited enough to be achievable, yet complex enough to push research in directions that will encourage progress toward large-scale tasks that we eventually want to be able to handle through ITL.

To keep these eventual goals of ITL in mind, we believe challenge problems can be identified in the five different domains identified in the goals of this Forum: assistive robotics, healthcare, education, training, and gaming. Developing specific problems in each of these complex domains should be possible to achieve in a two- to five-year research horizon, and would both stretch and focus the field. The complex real-world interaction problems and practical problems that are present in these domains would force the field to go beyond laboratory simplification. Focus would come through multiple research groups simultaneously investigating solutions to similar problems, enabling more immediate and impactful sharing of results.

However, there is an important practical problem to consider as well. The first four domains all involve real-world interaction. *Assistive robotics* and *healthcare* applications generally involve robotics, whereas *education* and *training* involve interactions with human participants. This creates a large barrier to entry for these four areas. Thus, a potential concern is that this additional complexity, inherent in the first four domains, will result in an overemphasis on *gaming* applications.

It is certainly the case that a gaming challenge task would be excellent for ITL. Indeed, large numbers of applicable games are already available in a convenient format (e.g., GitHub, FreeCiv), and games are already being widely used by deep reinforcement learning research. By using such games, it should be possible to show the advantages of ITL, especially if gaining expertise in such games through ITL occurs in a much smaller number of trials. Current noninteractive learning of these games often involves millions of trials, several orders of magnitude more than would be expected by ITL systems, or than is tolerable by people.

However, if gaming has a much lower barrier to entry than the other four domains, then the field runs the risk of spending too much effort in one area and ignoring the inherent challenges of the other areas. It may also fall into research traps, where some ITL research ends up only being applicable to the gaming domain and thus of little use to researchers in other domains.

For that reason, we believe it is vital to identify particular problems within the other four domains—assistive robotics, healthcare, education, and training—that also have a low barrier to entry. In particular, we are looking for problems where research progress can be made without research requirements that involve physical robotics or human subjects, due to the time needed to coordinate and to conduct studies with those participants. This may involve publicly available robotics simulators and publicly available data sets from human participants. We will also need researchers working in these domains with real robots and real interaction with human participants. By identifying helpful challenge tasks that do not have this large barrier to entry, researchers can contribute to the community without extensive overhead.

An example of a possible low-entry challenge domain is to build a trainable "pet" agent on a mobile phone or tablet. For instance, it could be taught new skills through the input/output capabilities of a phone or tablet. Input might consist of text, drawings, speech, or photos. The agent could be taught to answer simple questions or to play games. It could produce text and speech output or draw things itself. It would store knowledge gained through interaction and build on that in subsequent sessions. A potential useful application of this capability would be in primary education: children could teach their "pets" a particular skill and then all the agents in the class could then compete with each other to see whose agent has best learned the skill. For example, in a teachable agent project, students would "teach" their systems about a domain and then compete with other agents, either alone or in teams (Biswas et al. 2016).

Summary of Challenges and Opportunities

In this chapter we have reviewed and synthesized perspectives from multiple disciplines on the functional roles of knowledge and architecture to increase understanding of ITL systems and to assist in the development of synthetic ITL systems. Here we summarize the major themes that emerged from our analysis, focusing on the most important gaps that were identified in the existing state of knowledge as a basis for future, impactful research contributions.

What Are Suitable Formalisms for the Representation of Task Knowledge?

To understand how someone or something can learn a task, ITL needs a shared method of formalizing and describing tasks. In our discussion, we have attempted to characterize a "task" but acknowledge that this description is not sufficient, especially in terms of formalization. Formalization is important not only for practical purposes, such as for knowledge sharing and comparisons of task learning, but also to enable more productive communication and clarification within the field. Questions requiring further consideration include:

- Can suitable formalisms, ones that are comprehensive (express most tasks) and function-general (not assume specific architectural approaches or underlying computational representations), be constructed?

- How can tasks be formalized so that requirements for performing the task specify or indicate incremental modification of the representation of task behavior within the learning or performing system? A task specification may need to include task-level primitives, task concepts and terms, and task strategies as well as the specification of intermediate concepts that should be developed during task optimization. One example is the ability to recognize the board configurations, as in master chess players (Chase and Simon 1973).
- What methods can be developed to enable systematic characterization of tasks and the challenges for an agent learning that task? The community would benefit from being able to evaluate individual contributions focused on varying tasks if there was some way to understand the relative complexity of the tasks being learned. Relatedly, a researcher unfamiliar with a new task may wish to gauge the complexity of a task prior to attempting to get an agent to learn and to perform it.

Are Existing Computational Architectures Sufficient for ITL? What New Architectural Primitives Are Necessary for ITL?

Do existing cognitive and computational agent and learning architectures meet the functional requirements for ITL as summarized in this chapter? This is an empirical research question. As we noted, however, the highly dynamic and extremely fast dynamics of human speech, which is required for many kinds of ITL, may not be feasible in today's architectures. Some potential limitations may be attributable to a mismatch in the architectural primitives needed to support such interaction, rather than atheoretical constraints, such as available processing power. Additional questions that arise from this architectural perspective include:

- What constitutes a functionally sufficient or appropriate set of architectural primitives to support human conversational interaction at human timescales (as determined empirically, analytically, and from social psychological research)?
- Are novel architectural representations and mechanisms needed to support automatic prediction in computational agents? How can tractable automatic prediction, useful for agent learning and understanding, be supported at the many different timescales available in human prediction?
- How should architectures (better) support robust sharing of attention and reference during interaction? How can architectures more directly support the establishment and maintenance of local common ground during interaction?

- What constitute suitable representations and mechanisms to support behavior composition (and decomposition) in a scalable way? Are such functions and representations (necessarily) architectural?

What Novel Requirements for Learner Knowledge Representation Result from ITL?

Knowledge representations are shaped by the tasks and domains in which they will be applied, the ontological commitments of the representation to the situation that the representation will express (Davis et al. 1993). ITL forces a new consideration of the requirements for knowledge representations because specific task requirements cannot, by definition, be known in advance for a (general) ITL system. The psychology of learning and developmental psychology may be particularly useful in helping researchers understand how task knowledge is formed and shaped during all phases of task learning. Specific questions include:

- What are the requirements for a knowledge representation so that it will readily support generalization and transfer of task knowledge (task primitives, task terms and concepts, and task strategies) to new tasks? This question assumes that ITL requires that some distinct agent tasks share task knowledge.
- What are the necessary and sufficient knowledge primitives that will support efficient and general ITL across a wide range of domains?
- What is the ontology/taxonomy of task knowledge that should be acquired by a learner? How do the representations of different tasks interact with one another?
- How can an agent learn relatively complex procedures (such as counting) on the basis of instruction? In the human realm, it takes several years of experience for young children to learn to count (in a general way). How much time and effort is needed for an instructor to convey complex procedural information? How should such information be incorporated into existing agents (e.g., to what extent are production rules, often used to model procedural representation in humans, apt for procedural representation in ITL)?

What Are the Requirements for the Instructor and Learning Environment to Support ITL?

Unlike most artificial learning systems, ITL systems will, by definition, co-construct a learning environment with an instructor/teacher. In ITL, this will generally involve a triadic relationship, where the learner, the instructor, and the shared environment exert influences on one another to produce task learning (for more on this topic, see Shah et al., this volume). Such an arrangement is relatively novel for artificial learning systems. However, research from psychology and education regarding teacher–student interactions (see Van Lehn, this volume) may provide guidance in creating effective learning environments for artificial agents. Additional questions to consider include:

- How do human learners and teachers co-construct learning tasks during learning? How can ITL environments support such co-construction?
- How can ITL technologies take advantage of the troubleshooting and repair processes inherent in natural human dialogue to accelerate learning in ITL systems?
- What methods other than, or in addition to, natural language can be used to achieve common ground in instructional communication? Are there domain- and task-neutral technologies or methods that would allow a human instructor and agent learner to achieve effective common ground without using natural language?

How Might ITL Contribute to the Broader Goal of AI Systems?

The use of ITL for agent learning tends to assume that ITL will be a consumer of AI algorithms and tools, such as specific machine learning algorithms or functional components, like simultaneous localization and mapping or a natural language understanding component. We suggest that ITL may contribute both to the solution of outstanding challenges in AI as well as to the conceptualization of future AI systems (even if ITL is not used). Key questions to address include:

- How can an agent extract or produce symbol-like representations from continuous sensors and actions? Such symbol grounding—a long-standing goal of AI research—is still unmet, although significant progress was recently achieved in grounding for language learning, leveraging the recent availability of cheap and pervasive video (Perera and Allen 2014). ITL stresses symbol grounding, both in its requirements for conversational interactions at human scale as well as in the need to learn new task concepts, which may be expressed symbolically by an instructor but grounded in the perceptual and motor experience of the agent.
- What methods are fruitful for the integration and control of multiple learning systems? Most artificial ITL systems can be described as integrated, cognitive systems, a sub-area of research in AI. The focus on integration for a particular functional purpose (learning from instruction) may contribute to improved understanding in the composition of integrated cognitive systems generally.
- How might a human interact with, teach, and learn from nonhuman intelligence? Much of today's AI is dominated by methods that are functionally powerful, but they lack transparency and understandability. Increasingly, AI-inspired algorithms influence the day-to-day lives

From "Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,"
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.

of most living humans, sometimes with unintended and perverse consequences (O’Neil 2016). Thus, it is important that we understand their function and to direct them if necessary. ITL offers an approach that could serve as a foundation for more effective interaction, instruction, and understanding between human and nonhuman systems.

- What methods and techniques are useful for evaluating the generality of AI systems? Much of AI is focused on optimizing solutions to specific domain problems. Still, the goal of AI, as originally conceived, was to produce artificial general intelligence (McCorduck 2004). Because ITL is, by definition, not focused on single tasks or domains, it is likely to encourage generalization. If ITL is successful, it may also nudge the AI community away from its current single-task focus. The development of methods and tools to support the evaluation of ITL may offer a path to evaluate artificial general intelligence as well, by being more targeted and measurable.

From “Interactive Task Learning: Humans, Robots, and Agents Acquiring New Tasks through Natural Interactions,”
edited by K. A. Gluck and J. E. Laird. Strüngmann Forum Reports, vol. 26,
J. R. Lupp, series editor. Cambridge, MA: MIT Press. ISBN 978-0-262-03882-9.