Interactive Task Learning

John E. Laird, *University of Michigan*

Kevin Gluck, *Air Force Research Laboratory*

John Anderson, *Carnegie Mellon University*

Kenneth D. Forbus, *Northwestern University*

Odest Chadwicke Jenkins, *University of Michigan*

Christian Lebiere, *Carnegie Mellon University*

Dario Salvucci, *Drexel University*

Matthias Scheutz, *Tufts University*

Andrea Thomaz, *University of Texas*

Greg Trafton, *Naval Research Laboratory*

Robert E. Wray, *Soar Technology*

Shiwali Mohan, *PARC*

James R. Kirk, *University of Michigan*

In interactive task learning, an agent actively tries to learn the actual definition of a task through natural interaction with a human instructor, not just how to perform a task better.

An enabling characteristic of humans and other intelligent entities is that we aren't limited to a fixed set of innate or preprogrammed tasks. We quickly learn new tasks through language, gestures, observation, and other forms of natural communication. After we learn the essence of a task, we can learn extensions and modifications to it, getting better and better with experience. In contrast, if we want to get a computer to do a completely new task, the only practical approach has been through programming. This has served us well so far, but with advances in artificial intelligence, cognitive science, and robotics, we're approaching a future populated with intelligent systems that have the cognitive and physical capabilities to perform many different tasks. Unfortunately, it will be impossible to preprogram all the tasks that creative, generative human users will want their agents and robots to perform. It will be similarly unlikely that billions of users in the general public will have the specialized skills necessary to customize and extend the capabilities of their agents using programming languages. So how will future intelligent systems learn the tasks we want them to perform?

1541-1672/17/$33.00 © 2017 IEEE
Published by the IEEE Computer Society

Throughout the history of cognitive science, many research efforts have studied how to perform a task better,[1–4] but few have focused on learning the underlying concepts that define the task. For example, games and puzzles are usually defined by the environment in which the game takes place (such as a board), the objects that are used (pieces or cards), the legal actions that can be taken, losing states, and the goal to be achieved.[5,6] Assembly tasks, such as cooking a dinner or building a piece of furniture, are usually defined by a set of physical actions, available tools, component pieces, and constraints on the final product. Classification tasks involve learning labels for sets of data. Most learning research *assumes* these aspects of a task are known to an agent, and the agent's learning task is to learn how to perform a task well. In contrast, we're interested in how the agent learns an initial representation of a task so that the task can be attempted and then the agent can learn to perform it well.

One promising approach takes inspiration from how we teach humans new tasks.[7] Usually the need to teach a task arises in the context of other activities. A person who wants to learn the new task (*the student*) or a person that wants to teach it (*the instructor*) will initiate an interaction, using language, in which an overview of the task is described. During the interaction, the task's purpose is described, including when it is appropriate, and its goal or termination conditions. The instructor might also describe constraints on what actions can be used and, depending on the student's knowledge, provide some form of feedback, scaffolding, or even step-by-step instructions. Typically, the instruction occurs while the student is attempting to perform the task, with the instructor referring to real-world objects (by pointing or linguistic reference) and, if it's a physical activity, possibly demonstrating what the task involves. Throughout, the student asks questions, especially when there's an ambiguous instruction, or when the instructor uses novel terminology. The instructor might even ask questions of the student to assess the student's understanding of some aspect of the task, environment, or previous instructions. After a learner understands the essence of the task, he or she can then learn to do the task well, possibly through practice but also through additional interactions with the instructor or other humans.

After a learner understands the essence of the task, he or she can then learn to do the task well, possibly through practice but also through additional interactions with the instructor or other humans.

We call this general approach *interactive task learning* (ITL). Although clearly drawing inspiration from the human–human interactions described in the previous paragraph, our emphasis in this article is on intelligent artificial agents or robots learning new tasks through natural interactions with humans. The learner actively tries to assimilate the meaning of the instruction while performing the task, and learning occurs in conjunction with that task's performance. This is an ambitious problem to tackle, but recent progress in many fields suggests that now is the time to make a cooperative and coordinated attack on ITL, making it an intriguing, multidisciplinary challenge problem that spans several fields, including AI, cognitive science, and robotics. It draws on research advances in those fields, forcing us to confront many fundamental unsolved problems that arise when we have the persistent machine intelligence required to continually extend and customize tasks.

A central challenge of ITL is converting externally specified descriptions of a task into internal representations that are incrementally integrated with existing knowledge. The agent must be able to interpret those representations not only to produce behavior but eventually to perform the task as efficiently as if it were preprogrammed with that knowledge. Because the agent is learning tasks from scratch, its processes for interpreting the instructions must be task-independent. Furthermore, we expect the agent to immediately learn from each and every interaction with an instructor, which requires one-shot learning instead of repeated practice over large datasets (although practice might help tune the learned knowledge). These characteristics suggest that techniques such as deep learning and reinforcement learning aren't the core learning approaches in ITL.

Rather, ITL requires the integration of most, if not all, the capabilities we associate with cognition, including extracting task-relevant meaning from perception, language processing, dialog and interaction management, integrated knowledge-rich relational reasoning, problem solving, planning, learning, and metacognition. ITL's focus on integration is in contrast to most current research in AI and cognitive science, which has become increasingly fragmented and focused on specific technologies and techniques, and on narrow problems.

ITL has the potential to fundamentally change the way we interact with intelligent agents and robotic systems. No longer will these systems be limited to preprogrammed tasks. Human users will teach agents new tasks and help them improve their performance via instruction, demonstration, and feedback. This will include software and robotic agents, with applications in healthcare, where agents will need to be customized to patient needs; in industry, where tasks will change and be unique to a specific workplace; for rescue and military applications, where agents will need to be dynamically tasked to novel missions in new environments; and for the home, where service robots and software assistants will need to be extended and customized to their users. There are also many applications in developing intelligent support software for business and intelligence analysis, where creating and maintaining models of dynamically unfolding events based on incomplete, inaccurate, and deceptive information requires updating information gathering and integration strategies; in entertainment, where developing rich virtual characters that can be taught by players will open new dimensions of immersion; and in education, where ever-changing curricula and learner needs will require software tutors and coaches that are extensible without reprogramming.

In 2014, we held a US National Science Foundation (NSF)-sponsored workshop to take the first steps to define ITL and build a community of researchers that studies and develops the science and technology to support it. Here, we present the major products of that workshop: an analysis of desiderata for ITL systems, a synopsis of related work, and a discussion of possible application areas for ITL systems.

ITL Desiderata

To further clarify ITL research goals, we developed a set of desiderata that help define ITL as well as provide dimensions for evaluating and comparing ITL agents. These are desiderata for a complete and comprehensive ITL agent, at least as we're able to conceive of it today. Many of these desiderata won't apply to ITL agents that are developed for specific domains or for limited sets of tasks—rather, they focus on generality, effectiveness, and efficiency across task learning, performance, and interaction, with the added desiderata that task-learning capabilities are integrated with the agent's overall performance and available at any time to learn about any aspect of a task that the agent is pursuing.

Most AI systems are developed to perform a single task, with the definition of the task "baked" into the agent's design and structure. In contrast, ITL agents must have the ability to learn information and then later interpret it to produce task-relevant behavior.

General Task Learning

Most AI systems are developed to perform a single task, with the definition of the task "baked" into the agent's design and structure. In contrast, ITL agents must have the ability to learn information and then later interpret it to produce task-relevant behavior. Thus, a general long-term challenge for ITL research is developing methods to learn representations of not just a single type of task but methods that can learn many different types of tasks. Although complete generality across all tasks is a goal, we expect that early research in ITL will naturally pursue *task clusters* that share common environments and limit the diversity of types of task knowledge that can be learned, such as simple games, puzzles, or household chores.

Learning a new task can require learning new goals, concepts, actions, and procedures across different types of environments. Each of these categories of knowledge has a different underlying structure, so being able to learn one type of goal or concept for a task might not imply the ability to learn similar structures for other tasks. For example, although many tasks can be characterized as attempting to achieve a goal (defined by a set of constraints on a state of the environment), other tasks involve maintaining the state of the world within some constraints (homeostatic goals), such as keeping a plane flying, patrolling a building, performing a dance, or maintaining accurate knowledge of an ongoing sports event. Some tasks involve goals that are internal to an agent, such as learning a map of a building or learning someone's likes and dislikes. In addition, many tasks include some form of optimization, such as reaching a goal as quickly as possible, using the fewest resources. This diversity spans other aspects of tasks (concepts, actions, procedures, environments, and so on) and presents a major challenge for general ITL agents, which need general learning methods and internal data structures for representing and reasoning over that diversity to produce effective behavior.

One approach to limiting the diversity of what must be learned is by constraining the environment so that an agent can perform a task even with very little knowledge. Consider the case of playing chess. If an agent plays the game with a physical set of pieces and a board, the agent must physically set up the game, determine for itself which moves are legal, and make each move by physically moving the piece. It must know when it can castle, and it must be able to detect when the game has ended in a win, loss, or draw. Contrast this with an agent that plays the game using a computer simulation of chess: the computer enforces all the rules and presents the agent with a list of legal moves for each of its positions. After the agent chooses a move, the environment makes the appropriate changes to the board and detects when the game ends. In the first case, the agent must know much more to make a legal move and play a legal game. In the second case, the agent can choose randomly between the moves presented to it; it's guaranteed to make legal moves and play a legal game. It's even possible that the agent in the second case can learn to play the game well, but it will never learn a complete characterization of the game that it could communicate to a human. Thus, as we develop task-learning systems, we must also consider what aspects of the task are constrained by the environment, what must be learned, and how much the agent actually learns about a task when its environmental interactions are constrained.

A complementary goal of achieving general task learning is also developing an understanding of the limits of the interaction-based approach to task learning. What tasks are amenable to that approach? What tasks are not?

Effective Task Learning

The primary goal of an interactive task learner is to learn a task from its interactions with an instructor and from its own experiences. It must have reasoning and learning capabilities to interpret instructions and demonstrations within the context of its prior knowledge, ground them in the current situation, extract information about the task, generalize across multiple examples, and store the extracted knowledge in its memories for future use. What makes this capability especially challenging, in comparison to most research on machine learning, is that the learning doesn't occur within the confines of the structure of a known task, where a learning mechanism can be chosen to learn a specific type of data under prespecified circumstances. In learning a new task, an agent must be effective in learning many different types of knowledge, at different times, and from different types of interactions with an instructor.

The primary goal of an interactive task learner is to learn a task from its interactions with an instructor and from its own experiences.

Efficient Task Learning

In contrast to many other forms of learning that depend on "big" data, an ITL agent has a paucity of data from which to learn: its interactions with an instructor and its own interactions with the task environment. The advantage in ITL is that each interaction is targeted toward the agent, with the human providing information that's directly relevant to helping the agent learn the task. Each interaction is a "golden nugget" of tailored and targeted instruction that the agent can mine for the knowledge embedded within it. Another reason for the need for efficiency is that a human instructor will have limited patience. The agent needs to minimize its interactions and maximize the knowledge it extracts from each interaction, avoiding questions for which the answer is obvious. Furthermore, because it's in a real-time interaction with an instructor, the learning must also be efficient in the absolute time it takes for the agent to assimilate new information. The underlying learning algorithms can't take minutes or even seconds to process new data as that could disrupt the interaction (and annoy the instructor). Even if that processing occurs in parallel with the agent's interaction with the instructor, the new knowledge must be immediately available, so that the agent can use it as it progresses through the task. One additional efficiency challenge is that the agent must maintain its reactivity, even as it's continually learning new tasks and building up its knowledge. Its learning algorithms must not slow down as knowledge accumulates.

Effective Task Performance

Once an agent has learned the definition of a new task, it should be able to effectively use that knowledge to pursue the task. This requires that the agent operationalize the knowledge, converting it from the external declarative structure of natural language into an internal procedural representation that it can use to perform the task: determining when it's appropriate to attempt the task, determining when actions are appropriate, performing those actions, obeying additional task constraints, and recognizing when it has succeeded or failed. Beyond operationalization of its knowledge, the agent should be able to use other background knowledge and knowledge it has learned from earlier tasks to enhance its performance, such as using appropriate problem-solving and planning methods, or strategies and heuristics that work across multiple tasks. It should also be able to employ additional learning mechanisms beyond ITL to further improve performance, such as reinforcement learning or inductive concept learning. Furthermore, the agent should be skillful at managing its performance of tasks, such as being able to decide when it can pursue multiple tasks, when it needs to interrupt low priority tasks with higher priority tasks, and when it should resume suspended tasks as they become relevant.

Efficient Task Performance

In addition to effective task performance, the agent should be able to efficiently use the knowledge it has learned from instruction. Achieving efficient execution might not happen immediately, requiring additional internal reasoning, analysis, and learning, but in the limit, the agent's execution of a new task should approach the efficiency achieved when the task is programmed by hand in the same underlying architecture. As with learning efficiency, the agent's execution efficiency shouldn't degrade as more tasks, concepts, and procedures are learned.

Effective Interaction

ITL provides a vision of how untrained humans can teach agents new tasks and task knowledge. The long-term goal is for instructors to use a combination of unrestricted natural language, gestures, sketches, and simple demonstrations, so that the communication is effective, natural, and efficient. Effective interaction requires that both the agent and the instructor understand the content and intentions of an interaction given the current environment and instructional context. The agent must be able to extract the meaning from the instructor's utterance, no matter what modality is used, and must also be able to express itself so that the instructor can understand it. Moreover, the agent should be robust to errors in the instructions so that the teacher doesn't have to provide perfect instructions. There might be many different ways of supporting robustness, including having the agent verify the correctness of instructions in some way (such as through internal simulation) or trying out the instruction and allowing the teacher to correct any errors. Recent research on the nature of robustness,[8] a domain-general methodology for quantifying robustness and stability,[9] and mechanisms that produce robust cognitive systems[10] provides a conceptual and methodological foundation for the formal assessment of the degree of robustness achieved by systems capable of ITL, as well as implementation guidance for producing greater robustness. Finally, the ability to learn from instruction can potentially be applied to interaction so that an agent can also learn to improve its interaction capabilities through instruction.

One of the promises of ITL is that it's an efficient means for an instructor to teach an agent a new task. This includes minimizing the information that has to be communicated for a new task in terms of concepts but also in terms of the actual form of that information.

Accessible Interaction

Communication with an agent should be unconstrained and natural for an untrained instructor. Accessibility can be increased by supporting multiple modalities, such as language, gestures, and diagrams, as well as by allowing access to shared background knowledge and analogies. Communication shouldn't require extensive knowledge of the internal mechanisms and representations of a system on the part of the human, but there should be means for the human to build up an accurate model of the agent's capabilities and knowledge of a task.

Efficient Interaction

One of the promises of ITL is that it's an efficient means for an instructor to teach an agent a new task. This includes minimizing the information that has to be communicated for a new task in terms of concepts but also in terms of the actual form of that information, such as the number of words, gestures, or demonstrations. Contrast this efficiency with the inefficiency of programming an agent to perform a task, which requires a very specific and detailed set of instructions (the program) to be communicated in a language that's quite different from the ones used for human–human communication. Interaction efficiency should approach that required when humans teach tasks to other humans. Just as in accessible interaction, efficient interaction requires that an instructor can easily construct an abstract model of the agent's processing, reasoning, and existing knowledge. An instructor with a good model of a student can skip instructions that the student already has or can infer on its own, and focus on those aspects of the task that will be challenging for the student to learn. This implies that the agent can explain its reasoning and performance when necessary. An additional implication might be that the interaction should be modeled on human-to-human interaction, such as the communication that occurs between human teachers and students, between trainers and trainees, and between teammates, although this is an area of future research.

Use-Specific Desiderata

For many uses of ITL, there will be additional desiderata that arise from the specifics of the types of tasks being learned. For example, in cognitive modeling, the agent's learned behavior should model human behavior, and the agent's learning process should model human learning. Conversely, for many tasks, there are safety concerns and other restrictions on behaviors that the agent can perform, most generally that an agent shouldn't be taught "evil" behaviors. How will we have confidence that ITL systems will remain reliable and safe when their fundamental advantage is that they can learn entirely new things that weren't anticipated at production? The recent experience Microsoft had with its chat bot, Tay, clearly demonstrates that some people will dedicate themselves to nefarious manipulation of machine intelligence. Certainly, some people will attempt similar manipulations of ITL systems, trying to get them to learn new tasks that are offensive or malicious or dangerous, or some combination of those. Beyond the possibility of unethical interactive corruption of ITL systems, there are, of course, the standard concerns about cybersecurity. We must assume that these systems won't be standalone, off-the-grid products, so they must be protected from cyberattack. This latter version of the safety issue is already part of the global concern about secure machines, but the former is relatively new and specific to learning machines and promises to be a significant challenge for ITL systems.

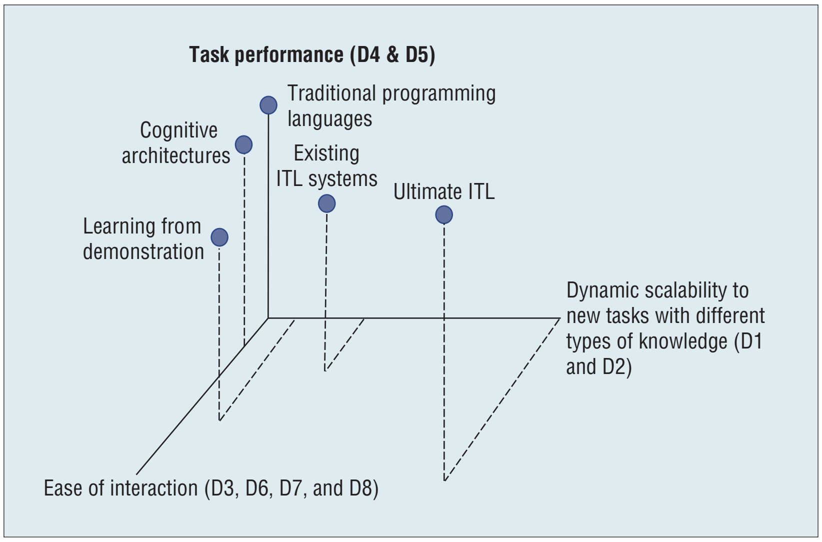
Task performance (D4 & D5)
Traditional programming languages
Cognitive architectures
Existing ITL systems
Ultimate ITL
Learning from demonstration
Dynamic scalability to new tasks with different types of knowledge (D1 and D2)
Ease of interaction (D3, D6, D7, and D8)

Figure 1. Comparison of task-specification techniques using major interactive task learning (ITL) desiderata.

Desiderata Summary

These desiderata allow us to compare our goal for ITL to the capabilities of other general approaches for specifying and developing agents (some of which are described in more detail in the following section). Figure 1 is an attempt to visualize a subset of the desiderata in three dimensions. The vertical dimension is a measure of task performance combined with efficient execution—the overall quality of task performance after learning. The horizontal dimension measures the ability to dynamically scale to new tasks. This is the essence of effective task learning together with the generality to learn many different tasks with different types of knowledge. The final dimension is the effectiveness, efficiency, and ease of communicating new knowledge, which involves teaching for an ITL system but programming for more traditional approaches. Not included in this figure are approaches (such as deep learning and reinforcement learning) that require massive training. Although those approaches are effective for learning some aspects of tasks, it isn't yet clear how they support efficient learning of many of the types of knowledge required in ITL.

Traditional programming languages support the development of systems with high task performance and efficient execution, but they're difficult to develop and don't inherently support dynamic extension to new tasks. Cognitive architectures[11] provide some important capabilities (such as memory structures, decision making, and learning mechanisms) that improve the ease of development and help support ITL while maintaining high task performance; however, they alone don't inherently provide dynamic learning of new tasks, although first-generation ITL systems have been developed using them. Learning from demonstration systems[12] usually results in good task performance on the tasks that they're taught, and they eliminate the need for programming, greatly simplifying the communication of knowledge. However, they're restricted to learning a few types of tasks, and there are significant limits on the types of knowledge that are easily communicated through demonstration. Some existing ITL systems can learn multiple tasks, although their efficiency and final performance is often not yet to the level of hand-programmed systems. They also don't have the flexibility or ease of teaching that we desire for the ultimate ITL system, which is at the extreme for all three dimensions.

Literature Review

Our review of related work is divided according to how information is communicated, either by natural language or demonstration. Within each, we proceed historically and discriminate between systems based on knowledge that's taught or specified, including task definition knowledge, but also considering knowledge that aids task performance, such as heuristics or policy information.

Learning from Language

Since the beginning of AI, there has been interest in developing agents that can take instruction in natural language. In 1958, John McCarthy proposed the Advice Taker,[13] which focused on taking advice about known tasks through natural language but didn't involve learning completely new tasks. SHRDLU, developed by Terry Winograd,[14] processed limited natural language commands, solved problems in a simulated blocks world, and acquired knowledge about its environment from language including the definition of composite concepts; however, it didn't learn new tasks.

The first attempt at task learning was Understand,[15,16] the goal of which was to extract the actions and goals from natural language specifications of multiple isomorphisms of the Tower of Hanoi puzzle. This effort shared many of the goals of ITL. However, the learning wasn't interactive, much of the translation from language to internal data structures was done through hand simulation, and it was never applied to other problems. In 1975, the Instructable Production System (IPS) project[17] was launched at Carnegie Mellon University with the goal of developing a system that could learn from instruction; however, it failed, in large part because of the lack of a general formulation of task representations.

An alternative approach to using language for teaching was explored by a team[18,19] whose noninteractive approach explored learning three heuristics for the game of Hearts using logical expressions as input. For example, a possible bit of advice is, "Avoid taking tricks with hearts in them." Jack Mostow's BAR system would take in a logical representation of that statement and transform it (a process called *operationalization*) so that a problem solver could use it in playing the game.

Colleen Crangle and Patrick Suppes[20] presented a theoretical examination of the issues and challenges that arise in commanding robots through natural language, as well as teaching them new tasks. Their analysis didn't include any specific computational systems but considered many of the problems that arise in ITL from a theoretical perspective.

Since the beginning of AI, there has been interest in developing agents that can take instruction in natural language.

Instructo-Soar[21] was an agent developed in Soar that learned simple but novel block manipulation tasks from simple natural language instructions in a simulated robotic domain. It could learn to compose learned tasks when learning additional tasks. Instructo-Soar pushed the state of the art in cognitive architecture (at the time) to its limits.

Alfredo Weitzenfeld and colleagues[22] created a method for "coaching" a robot that plays soccer by teaching it basic behaviors trained from a sequence of existing actions and hierarchical multirobot strategies. The language was constrained for the domain and wasn't interactive. Stephanie Tellex and colleagues[23] created a system that learned a mapping from natural language commands to robotic tasks. It learned through the analysis of a large corpora of such interactions and not through one-on-one interaction with a human. Similarly, David Chen and Raymond Mooney[24] created a system that learned mappings from language for simple navigation tasks, but as with Tellex and colleagues,[23] their system doesn't learn new tasks.

Advances in the Soar architecture[25] made it possible to return to many of the ideas first pursued in Instructo-Soar. These advances have led to the development of a robotic agent named Rosie,[26,27] which can interactively learn basic concepts such as attributes of objects (color: red, size: large), spatial relationships (right-of), and simple actions (move) through language interaction in a real-world robotic environment. Rosie also acquires new vocabulary of adjectives, nouns, prepositions, and verbs that are grounded in basic concepts and can be used in interactions. Rosie has been extended to learn 17 simple games and puzzles, such as Tower of Hanoi and Tic-Tac-Toe.[28,29] Rosie also works on a mobile robot that can learn office delivery tasks and strategies for finding objects that aren't directly observable.[30]

Rehj Cantrell and colleagues[31] describe a mobile robotic system implemented in DIARC that can be taught individual task actions via verbal commands by specifying preconditions ("you are at a closed door"), action definitions ("you push it one meter"), and postconditions ("you will be in the room") but can't learn completely new tasks as goals the agent must achieve on its own or tasks that have complex internal structure.

Thomas Hinrichs and Kenneth Forbus[32] developed a system based on the Companions architecture that learns Tic-Tac-Toe and Hexapawn through a combination of language and demonstration via sketching. It transforms the language and sketching information into the Game Description Language (GDL),[33] which is then interpreted to play the games.

Maxime Petit and Yiannis Demiris[34] use language to teach an iCub robot its body parts and then to label proto-actions (such as folding a thumb closed) that the robot generates during motor babbling so that no primitive actions must be known by the robot a priori. The instructor can then teach more complex motor actions, such as closing a hand, but using sequences of taught proto-actions.

Additional systems[35,36] allow a human to teach a task by guiding the agent through a solution to a problem, rather than teaching the problem specification. These systems often use specialized languages or interfaces, and are often characterized as programming by demonstration, although using language to lead the agent instead of pure demonstration. These systems learn new tasks or modifications to existing tasks but are limited in the types of tasks they can learn, the methods for specifying the tasks, and their ability to only accept tasks that are defined by sequences of actions. For example, the Tailor system[37] allows the user to use natural language instructions to modify task information and checks the modification to ensure there are no undesirable side effects. PLOW[35] is a collaborative task-learning agent that acquires procedural knowledge through a collaborative session of demonstration, learning, and dialog. The human teacher provides a set of tutorial instructions accompanied by related demonstrations. The agent uses these to acquire new procedural knowledge. Although the learning is human-demonstration driven, the agent controls certain aspects of its learning by making generalizations without requiring the human to provide a large number of examples. LIA[38] is an interactive, instructable intelligent agent that can learn new commands for managing email by being given step-by-step instructions in natural language.

There are also systems that attempt to learn from reading and possibly diagrams.[39–41] These systems don't learn interactively, nor do they learn new tasks, but they do use language as a means for acquiring new knowledge.

Learning from Observation and Demonstration

An alternative to using language to teach a task has been to use demonstrations of tasks. In this approach, the agent must induce the goals, policies, and possibly rules of a task by observing a human or other robot perform it (learning by observation) or by being led through the task through tele-operation or physical manipulation of a robot's effectors (learning by demonstration[12]). This latter approach is popular for teaching control policies for action execution, including object manipulation and locomotion, and differs from systems in which a human directs the robot through natural language commands because it relies on physical movement instead of language. For example, Claude Sammut and colleagues[42] created a system that learned to fly a simulated plane by observing human piloting. Some work has been done on learning task goal concepts from demonstration.[43] Learning by demonstration has also been used to learn qualitative models by observing the actions of an instructor in a game (Freeciv), using causal models to improve task performance within the game.[44]

An alternative to using language to teach a task has been to use demonstrations of tasks.

A few systems have been developed that learn game rules in physical environments through observation of gameplay.[45,46] These projects focus on learning a single task and require large numbers of demonstrations and the labeling of illegal gameplay.

A few systems combine learning by observation or demonstration with additional communication via language to learn policy knowledge but not task specification. Language can be used to refine or clarify what's learned by demonstration[47] or to provide a goal structure as context for learning more complex tasks.[48,49] Other methods provide more structure for learning from observation, including segmented observation data.[50] Some of these systems[47,50] also refine their knowledge by learning from experience.

Domains and Associated Applications for ITL Research

For a given domain, there can be multiple applications—for example, interactive mobile robots can be used in medical care, industry, the military, and the home. Thus, there appear to be many "vertical" slices of domains, each with clusters of tasks, where interactive task learning could have a positive impact.

Games and Puzzles

Games and puzzles have many properties that make them attractive as testbeds for initial investigations of ITL. They vary in complexity from simple games such as Tic-Tac-Toe, which can be taught in a few minutes and played with pencil and paper, to complex computer strategy games such as *Civilization* or

Starcraft, which require a computer to manage the complex rules and possible dynamics of the game. In general, games have well-defined actions, rules, and goals that are embedded in spatial domains. They also are tasks whose formulation can usually be learned independent of a specific strategy or policy for task performance. For example, when learning chess, an agent can learn about the board, pieces, their movements, and the rules of the game without ever learning a strategy. For many other tasks, such as cooking, learning to perform the task is intertwined with what the task is (baking a chocolate cake) and how the task is achieved (the individual steps of measuring and combining ingredients). Games and puzzles also have the advantage that there are many existing examples that have diverse types of rules. In addition, people are familiar with teaching and learning games, making it natural for them to teach computers new games. The ITL approach to learning games is in contrast to the goals of the General Gameplay Competition, where the rules of games are described in GDL,[33] which is based on Prolog, and there's no interaction between a teacher and learning.

Although there has been success with simple games with limited numbers of objects, it's unclear whether current techniques can scale up to more complex games. Some of the dimensions of complexity include the number of movable pieces and places (such as in chess or Go), the number and complexity of rules and their interactions (such as in strategy games), the responsiveness/speed of gameplay (such as in speed chess), and the number and types of relevant spatial relations (such as terrain in *Civilization*). Below are some possible sets of games that could be challenge problems for ITL systems:

- *Simple puzzles and games*. James Kirk and John Laird[29] have collected 17 such games, which could be expanded to include chess, checkers, card games, and so forth.
- *Freeciv*. This open source version of the turn-based strategy game *Civilization* involves complex rules, terrain, and multiple players and is well suited as a challenge problem for ITL systems.
- *Video games*. There are many video games that are freely available and can easily be interfaced to ITL systems. These include classic Atari games, such as Frogger and Space Invaders. However, for many video games, the task specification is trivial: move an object through a world, score points, and avoid death, a simplicity that has made them an excellent training ground for reinforcement learning, but at least for many of them, there is little need for ITL.

Assistive Support Robots

Assistive robots offer an obvious domain for ITL as rarely is it possible to predict all the different tasks a human user will want a robot to do. Moreover, research in mobile, multipurpose robots has greatly expanded in recent years, with more robots having the capabilities needed to perform many different tasks interactively with a human, with such application areas as domestic, medical, industrial, and military robots. As described early, there's growing research in dynamic teaching of robots, including learning by demonstration, observation, and through instruction. However, there are some challenges as well:

- *Integration and real-world systems*. Robotic systems require an integrated architecture and functioning systems that have at least partial solutions for object identification, manipulation, navigation, natural language, safety, human–robot interaction, and learning. While current researchers have partial solutions for many of these areas, integrating them all into a real-time system that can learn from instruction is a major challenge.
- *Symbol grounding and uncertainty*. Robots sense and act in a highly uncertain world, which creates major challenges for endowing taskable systems with basic knowledge about the world. Such symbol grounding is critical to bridge high-level reasoning and taskability with low-level sensing and actuation. This inherent uncertainty takes several forms, primarily perceptually grounding basic axioms about the world as symbols estimated from noisy sensor signals, predicting the effects resulting from actions to ensure goals can be reached by executing actions in sequence, and estimating the intentions and goals of human teachers demonstrating tasks to robots.
- *Metrics*. Another challenge is how to define metrics for evaluating an integrated interactive task-learning system. Most such systems will be less computationally efficient than specialized systems optimized for a specific task, so metrics and evaluations need to span multiple tasks and domains.
- *Interoperability and time-sharing*. Robotics has the ongoing problem that a particular algorithm often doesn't work on a different platform (sensor or robot). Science is, at its core, the ability to verify knowledge and findings independently through reproducible experimentation. To share knowledge, the challenge of the same running architecture across different physical platforms must be addressed.

Some possible classes of challenge problems include assembly tasks taught through interactive instruction, cleaning up novel toys in a kid's playroom, preparing meals with new tools in a novel kitchen, and a taskable home care robot for an elderly or disabled individual, capable of cleaning up a home.

Personal Assistants

Many important kinds of intelligent agents today are purely software, providing services in the cyberworld where so much of our thinking and work take place. Intelligence analysts build models of countries, factions, parties, and people, while business analysts build models of products, markets, supply chains, and other aspects of their operating environment. Both types of analysts must deal with information that's incomplete, uncertain, and sometimes aimed at deception. Scientists, engineers, and doctors must handle an explosion of literature, within which there is contradiction, uncertainty, and much that is irrelevant. Software assistants need to be taskable and customizable to the needs of individual users, and they need to learn new analytic practices and rapidly get up to speed in new areas of interest without programming.

One of the major bottlenecks for such agents is deep natural language understanding. Progress in question-answering systems provides an interesting illustration. Accuracy for most question-answering systems has remained at roughly 30 percent for years, as the results from the TREC competition illustrated. By contrast, IBM's Watson achieved accuracy in the top end of the 80th percentile, using a combination of structured representations (including a 900 million-element frame representation, automatically constructed via reading text) and statistical machine learning, which was enough to routinely beat human champions at the real-time TV quiz show *Jeopardy!* Watson's learning processes and reasoning processes were completely hand-tailored for the problem at hand by a team of human experts who put in a person-century of work to succeed. Obviously, this approach doesn't scale. Interactive task learning could potentially enable anyone to customize and extend, in other words, to truly personalize their software assistants.

The state of the art in such systems is still quite primitive. Learning-by-reading systems[39,40] can extract deep knowledge from texts. For example, Kate Lockwood and Kenneth Forbus[41] describe a system that, after processing a textbook chapter expressed in simplified English and sketched diagrams, was capable of correctly answering 12 out of 15 questions posed at the end of the chapter. Dario Salvucci[51] has shown that knowledge from Semantic Web resources can be combined with ACT-R to provide answers to "factoid" questions. Project Halo[39] focused on exploring the kinds of reasoning needed to solve Advanced Placement test problems in multiple domains, while Forbus's Companion cognitive architecture has been used to learn to solve AP Physics problems[52] and new physics problems via cross-domain analogies.[53] While progress continues to be made in all of these areas, the building of integrated systems that use them to experiment with interactive task learning could push the state of the art forward even faster, especially if improving the knowledge of such systems became an important class of experimental tasks. Here are some possible challenge problems:

Many important kinds of intelligent agents today are purely software, providing services in the cyberworld where so much of our thinking and work take place.

- *Learning background knowledge.* Using human-normed textbooks, including reading comprehension books, the challenge would be to learn enough about the world to do well on human-normed tests, such as the New York State Regents Exams.
- *Analyst's assistant.* Given background materials on a new topic, the challenge would be to build up a conceptual model of the topic, extending the model as new information becomes available and responding to queries about it. Topics might include countries, crises, products, or research findings, while the background materials might include text, sketches, pictures, and video. Queries might include prediction questions, abductive questions (inference to best explanation, determining what sort of business strategy is most consistent with company X purchasing company Y), and historical analogies.

Constructive Agents and Virtual Humans

We use the terms "constructive agents" and "virtual humans" here to be inclusive and attempt to group several clusters of research and development that have emerged around closely related terminology. These have somewhat different connotations across education, training, and entertainment applications. Constructive agents and virtual humans are implemented in software and run without the direct involvement of (or with low-level) remote control from human operators. This distinguishes them from the avatars that represent human users in computer games and virtual world environments such as *Second Life* or OpenSim.

The term "virtual human" is more often used when the agent takes on some form of simulated physical embodiment and when it's designed to exhibit human-like behavior. An example might be a virtual character in a role-playing game that talks with the human game player. The term "constructive agent" is more common when there's no physical embodiment perceived by humans interacting with the agent and when human-like performance levels aren't necessarily a constraint (it might be acceptable or even desirable for the agent to behave in a perfectly predictable and reliable manner every time it completes a task). An example of a constructive agent could be one that controls an airplane or tank in a simulation training application. These terms and associated underlying technologies have fuzzy boundaries, and the terminology is often used interchangeably across and within research communities.

Various forms of constructive agent and virtual human capability have been the focus of scientific and technical investments going back at least 30 years. Two summative National Research Council (NRC) reports[54,55] have been published on the topic. Intended applications are most commonly in education and training, although they're increasingly commonplace in video games[56] and in film and television.[57] Subtypes of these systems tend to cluster into opponents, teammates, and crowds.

Constructive agents or virtual humans are typically necessary to provide a realistic and interactive social environment in which human experience takes place in a virtual environment. However, these capabilities are costly to develop, and, because they're often specialized during implementation for a particular application, they typically require significant additional manual refinement to adapt to a new application.[58] Consistent issues associated with brittleness and development costs have plagued research and application efforts. Even modest progress in the direction of ITL systems could result in a dramatic decrease in development costs and generalizability.

These challenges have motivated researchers to explore the use of machine learning to develop constructive agent and virtual human capabilities. For example, Laird and colleagues[48,49] employ traces of entity behavior in simulation, along with annotated descriptions of goals and plans, to learn an agent behavior model. Automatic construction of interactive game characters has also been explored in the game and entertainment research community.[59] In almost all cases, there's little interaction (natural or otherwise) between the actual user of the virtual human, who might be in the best position to describe requirements, and the development process. A software developer acts as the intermediary, interpreting requirements and creating new behavior models from those requirements. This separation has led to the development of abstract programming interfaces, such as graphical programming tools and high-level languages, but these agents would be more useful, less brittle, and have a longer and more flexible life cycle if users could customize and specialize their behavior via natural interactions.

The term "virtual human" is more often used when the agent takes on some form of simulated physical embodiment and when it's designed to exhibit human-like behavior.

Learning behavior through natural interaction is being explored, however. Juatin Permar and Brian Magerko[60] describe a system that learns to improvise a dance with a human partner. Although the scope of the learning interaction is limited to movements, the system readily creates new dance behaviors without being dependent on specialized programming or user training. Moving from dance partners to mission teammates, the Air Force Research Laboratory (AFRL) has been developing a synthetic teammate capability.[61] It uses a theory of language processing[62,63] combined with a dynamically constructed situation model[64] to interpret and respond to text chat among teammates in simulated UAV missions. The synthetic teammate's relevance to ITL objectives is in the natural, interactive manner in which it develops its internal mission situation model through text chat with teammates.

ITL could make a significant contribution toward several important challenges in constructive agents and virtual humans. First, conversational interactions that allowed an agent to gain insight into the specific goals and requirements of a particular user experience could be important for an agent that was primarily manually programmed. By analogy, imagine actors getting input about a scene or story arc from a director or a squadron of pilots receiving their mission brief. These conversations inform the players in carrying out their behavior in the scene and having greater context to make good decisions autonomously. Second, interactive task learning would help constructive agents and virtual humans migrate to new applications or even domains. An agent that knows how to fly a fighter plane should be able to learn to fly a commercial airliner or a cargo plane, for example. Below are some possible challenge problems:

- *At-a-distance actors.* Given guidance from a teacher, the agent would learn to exhibit/behave within given/prescribed norms. The challenge addresses the need for realistic "background characters" in film and television and in some kinds of simulation training. To address this challenge, learning systems might need to "take in" cultural cues and learn to exhibit them. Addressing this challenge likely requires embodiment (virtual human).
- *Platform operator.* For this challenge, the agent would learn to control a simulated physical platform, such as a car, tank, or aircraft. Different vehicles and applications will have different requirements for physical fidelity and thus the complexity of control task will vary depending on the domain (ship at sea versus supersonic aircraft). However, solutions likely share common properties and requirements that could be exploited across multiple research groups. This constructive agent approach probably does not require embodiment.
- *Role-players.* For this challenge, assume some constructive agent or virtual human has been developed that exhibits a high degree of capability and knowledge of a domain and application. The challenge is to learn to be an effective role-player within that domain—that is, learn to customize how behavior is manifest in different situations based on requirements for that context. This challenge has some potential pitfalls in terms of how the prior knowledge would need to be structured to support this capability. Further, it's a narrower problem than ITL is pursuing in general because it's focused on producing variation of behavior within a domain. However, it would offer immediate practical utility as a new feature for existing models/virtual actors. It might also be worthwhile as a goal of ITL research to frame ITL capability as one that could learn to exploit a virtual human or constructive agent, rather than necessarily being integrated within it.

Cognitive Science Research

Whereas recent AI approaches to task learning have primarily emphasized functionality and scope, recent cognitive science approaches have primarily focused on psychological plausibility and validity with respect to human learning. For example, several recent efforts have explored how computational models can, like people, learn by reading and following instructions.[51,65,66] In such work, the models encode instructions as declarative representations, and then proceduralize the instructions over time, thus moving from novice to expert performance. Christian Lebiere and colleagues[67] extend this approach to building shared mental models in the domain of human–robot interaction and later apply it to sensemaking for geospatial intelligence analysis.[68] These models combine instructions specifying a decision-making procedure with learning from demonstration and exploration to make individual decisions.

Different vehicles and applications will have different requirements for physical fidelity and thus the complexity of control task will vary depending on the domain (ship at sea versus supersonic aircraft). However, solutions likely share common properties and requirements that could be exploited across multiple research groups.

That said, many challenges remain in developing cognitive models of task learning. One of the most important is moving beyond instruction following to other common forms of learning, such as learning by example. People are very adept at mixing forms of learning as they study new tasks, whereas current models by and large follow a single method of learning. Another important challenge involves developing better models of natural language understanding. Current cognitive science models have typically incorporated basic parsing and understanding but only in a very limited scope, whereas interactive task learning and collaboration will require a much richer and more flexible model of understanding. Perhaps the most fundamental challenge is modeling not just the initial learning to perform the task from instructions and background knowledge but the entire learning curve including higher levels of expertise with the task. This requires the acquisition of high-level task concepts that aren't directly instructable but instead must be discovered through experience. Below are some possible challenge problems:

- *Simulated students.* Using a cognitive architecture as an underlying framework, we can develop simulated students that interactively receive instruction and, at the same time, practice new tasks and improve on these tasks through learning. Such a simulated student could serve to predict the difficulty of new problems, and could also be incorporated into intelligent tutoring systems for more accurate inference of a student's current skill level. Some foundational research in this direction has been completed,[69–72] yet challenges remain in improving on the generality and degree of natural interaction associated with these capabilities.

- *Interactive learner for psychological experiments*. Another potential challenge problem involves using a cognitive architecture to develop a model that can learn to perform psychological experiments in the way that humans do. In a standard experiment, a person enters with a large base of underlying procedural and declarative knowledge; during an instruction period and during the experimental task itself, the person augments this knowledge with an understanding of how to perform the specific task being asked. A preliminary model in this direction has been developed,[51] but there remain significant challenges to making a broader interactive learner. First, the learner requires a significant natural language component to accept instructions and translate them to actions. Second, although there has been significant progress in learning from instructions, the experiment learner needs to acquire knowledge in multiple ways—especially in learning by example—and these avenues remain largely unexplored in cognitive modeling. Third, the experiment learner's background knowledge is extremely important and necessitates large-scale understanding and development of useful bodies of knowledge that can be applied to tasks.
- *Cognitive debiaser*. In contrast to physically embodied domains such as robotics, information manipulation in cyberspace is becoming an increasingly important part of both everyday life and professional activities. A potential challenge is to develop a system that could serve as a personal assistant to an intelligence analyst, modeling his cognitive processes at an individual level to alleviate shortcomings such as cognitive biases and attentional bottlenecks. The task could be defined as receiving layers of information and issuing probability judgments over a space of hypotheses. This challenge would advance the state of the art in cognitive modeling by requiring the integration of a predictive, quantitative theory of myriad cognitive biases documented in the decision-making literature. A potential application would be to evaluate the impact of structured analytic techniques[73] that have been proposed as a solution to cognitive biases. Metrics would be designed to quantify biases as deviations from rational normative performance.

Many challenges remain in developing cognitive models of task learning. One of the most important is moving beyond instruction following to other common forms of learning, such as learning by example.

We've tried to provide an overview of the landscape of interactive task learning, including its definition, the associated desiderata for evaluating future progress, a review of previous research in related areas, and potential application areas. Interactive task learning is a constellation of approaches and research problems that if realized, holds the promise of dramatically changing the way we develop and extend the capabilities of intelligent agents. No longer will we need to rely on programmers to anticipate all of the potential tasks that our agents will perform. ■

References

1. K.A. Ericsson and A.C. Lehmann, "Expert and Exceptional Performance: Evidence of Maximal Adaptation to Task Constraints," *Ann. Rev. Psychology*, vol. 47, 1996, pp. 273–305.
2. P.M. Fitts and M.I. Posner, *Learning and Skilled Performance in Human Performance*, Brock-Cole, 1967.
3. F.J. Lee and J.R. Anderson, "Does Learning a Complex Task Have to Be Complex? A Study in Learning Decomposition," *Cognitive Psychology*, vol. 42, 2001, pp. 267–316.
4. K.A. Ericsson, R.T. Krampe, and C. Tesch-Romer, "The Role of Deliberate Practice in the Acquisition of Expert Performance," *Psychological Rev.*, vol. 100, 1993, pp. 363–406.
5. W.G. Chase and H.A. Simon, "The Mind's Eye in Chess," *Visual Information Processing*, W.G. Chase, ed., Academic Press, 1973, pp. 215–281.
6. F. Gobet and H.A. Simon, "Recall of Rapidly Presented Random Chess Positions Is a Function of Skill," *Psychonomic Bulletin & Rev.*, vol. 3, 1996, pp. 159–163.
7. M.D. Merrill, "First Principles of Instruction," *Educational Technology Research and Development*, vol. 50, 2002, pp. 43–59.
8. K.A. Gluck et al., "Robustness in a Variable Environment," *Evolution and the Mechanisms of Decision Making*, J.R. Stevens and P. Hammerstein, eds., MIT Press, 2012, pp. 195–214.
9. M.M. Walsh, E.H. Einstein, and K.A. Gluck, "A Quantification of Robustness," *J. Applied Research in Memory and Cognition*, vol. 2, 2013, pp. 137–148.

10. M.W. Walsh and K.A. Gluck, “Mechanisms for Robust Cognition,” *Cognitive Science*, vol. 39, 2014, pp. 1131–1171.
11. P. Langley, J.E. Laird, and S. Rogers, “Cognitive Architectures: Research Issues and Challenges,” *Cognitive Systems Research*, vol. 10, 2009, pp. 141–160.
12. B.D. Argall et al., “A Survey of Robot Learning from Demonstration,” *Robotics and Autonomous Systems*, vol. 57, 2009, pp. 469–483.
13. J. McCarthy, “Programs with Common Sense,” *Symp. Mechanization of Thought Processes*, Nat’l Physical Laboratory, 1958, pp. 299–308.
14. T. Winograd, *Understanding Natural Language*, Academic Press, 1972.
15. H.A. Simon, “Artificial Intelligence Systems That Understand,” *Proc. 5th Int’l Joint Conf. Artificial Intelligence*, 1977, pp. 1059–1073.
16. H.A. Simon and J.R. Hayes, “The Understanding Process: Problem Isomorphs,” *Cognitive Psychology*, vol. 8, 1976, pp. 165–190.
17. M. Rychener, “The Instructable Production System: A Retrospective Analysis,” *Machine Learning: An Artificial Intelligence Approach*, R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, eds., Morgan Kaufmann, 1983, pp. 429–459.
18. F. Hayes-Roth, P. Klahr, and D.J. Mostow, “Advice Taking and Knowledge Refinement: An Iterative View of Skill Acquisition,” *Cognitive Skills and Their Acquisition*, J.A. Anderson, ed., Erlbaum, 1981, pp. 231–253.
19. D.J. Mostow, “Machine Transformation of Advice into a Heuristic Search Procedure,” *Machine Learning*, J.G. Carbonell, R.S. Michalski, and T.M. Mitchell, eds., Tioga Publishing, 1983, pp. 367–403.
20. C. Crangle and P. Suppes, “Language and Learning for Robots,” CSLI lecture notes no. 41, Centre for the Study of Language and Communications, Stanford Univ., 1994.
21. S.B. Huffman and J.E. Laird, “Flexibly Instructable Agents,” *J. Artificial Intelligence Research*, vol. 3, 1995, pp. 271–324.
22. A. Weitzenfeld, A. Ejnioui, and P. Dominey, “Human Robot Interaction: Coaching to Play Soccer via Spoken-Language,” *Proc. IEEE/RAS Humanoids 2010 Workshop Humanoid Robots Learning from Human Interaction*, 2010; http://weitzenfeld.robolat.org/wp-content/uploads/2015/01/HRLHI10Weitzenfeld-EjniouiDominey.pdf.
23. S. Tellex et al., “Understanding Natural Language Commands for Robotic Navigation and Mobile Manipulation,” *Proc. 25th AAAI Conf. Artificial Intelligence*, 2011, pp. 1507–1514.
24. D. Chen and R. Mooney, “Learning to Interpret Natural Language Navigation Instructions from Observations,” *Proc. 25th AAAI Conf. Artificial Intelligence*, 2011, pp. 859–865.

THE AUTHORS

John E. Laird is the John L. Tishman Professor of Engineering at the University of Michigan. He received a PhD in computer science from Carnegie Mellon University. Contact him at laird@umich.edu.

Kevin Gluck is a principal cognitive scientist with the Air Force Research Laboratory. He received a PhD in cognitive psychology from Carnegie Mellon University. Contact him at kevin.gluck@us.af.mil.

John Anderson is the Richard King Mellon Professor of Psychology and Computer Science at Carnegie Mellon University. He received a PhD in psychology from Stanford University. Contact him at ja@cmu.edu.

Kenneth D. Forbus is the Walter P. Murphy Professor of Computer Science and Professor of Education at Northwestern University. He received a PhD in computer science from MIT. Contact him at forbus@northwestern.edu.

Odest Chadwicke Jenkins is an associate professor of computer science and engineering at the University of Michigan. He received a PhD in computer science from the University of Southern California. Contact him at ocj@umich.edu.

Christian Lebiere is part of the research faculty in the Psychology Department at Carnegie Mellon University. He received a PhD in computer science from Carnegie Mellon University. Contact him at cl@cmu.edu.

Dario Salvucci is a professor of computer science at Drexel University. He received a PhD in computer science from Carnegie Mellon University. Contact him at salvucci@drexel.edu.

Matthias Scheutz is a professor of cognitive and computer science in the Department of Computer Science at Tufts University. He received a PhD in philosophy from the University of Vienna and a joint PhD in cognitive science and computer science from Indiana University Bloomington. Contact him at matthias.scheutz@tufts.edu.

Andrea Thomaz is an associate professor of electrical and computer engineering at the University of Texas at Austin, where she directs the Socially Intelligent Machines lab. She received a PhD in computer science from MIT. Contact her at athomaz@ece.utexas.edu.

Greg Trafton is a section head in the artificial intelligence branch of the Naval Research Laboratory. He received a PhD in cognitive science from Princeton University. Contact him at greg.trafton@nrl.navy.mil.

Robert E. Wray is senior scientist at Soar Technology. He received a PhD in computer science and engineering from the University of Michigan. Contact him at wray@soartech.com.

Shiwali Mohan is an AI researcher at PARC. She received a PhD in computer science and engineering from the University of Michigan. Contact her at shiwali.mohan@parc.com.

James R. Kirk is a PhD student in the Computer Science and Engineering department at the University of Michigan. Contact him at jrkirk@umich.edu.

25. J.E. Laird, *The Soar Cognitive Architecture*, MIT Press, 2012.
26. S. Mohan and J.E. Laird, "Learning Goal-Oriented Hierarchical Tasks from Situated Interactive Instruction," *Proc. 28th AAAI Conf. Artificial Intelligence*, 2014, pp. 113–130.
27. S. Mohan et al., "Acquiring Grounded Representation of Words with Situated Interactive Instruction," *Advances in Cognitive Systems*, vol. 2, 2012, pp. 113–130.
28. J.R. Kirk and J.E. Laird, "Interactive Task Learning for Simple Games," *Advances in Cognitive Systems*, vol. 3, 2014, pp. 13–30.
29. J.R. Kirk and J.E. Laird, "Learning General and Efficient Representations of Novel Games through Interactive Instruction," *Proc. 4th Conf. Advances in Cognitive Systems*, 2016; www.cogsys.org/papers/ACS2016/Papers/Kirk_Laird-ACS-2016.pdf.
30. A. Mininger and J.E. Laird, "Interactively Learning Strategies for Handling References to Unseen or Unknown Objects," *Proc. 4th Conf. Advances in Cognitive Systems*, 2016; www.cogsys.org/papers/ACS2016/Papers/Mininger_Laird-ACS-2016.pdf.
31. R. Cantrell et al., "Tell Me When and Why to Do It! Run-Time Planner Model Updates via Natural Language Instruction," *Proc. 7th Ann. ACM/IEEE Int'l Conf. Human-Robot Interaction*, 2012, pp. 471–478.
32. T. Hinrichs and K. Forbus, "X Goes First: Teaching Simple Games through Multimodal Interaction," *Proc. 2nd Conf. Advances in Cognitive Systems*, 2013, pp. 205–218.
33. M. Genesereth, N. Love, and B. Pell, "General Game Playing: Overview of the AAAI Competition," *AI Magazine*, 2005, pp. 62–72.
34. M. Petit and Y. Demiris, "Hierarchical Action Learning by Instruction Through Interactive Grounding of Body Parts and Proto-Actions," *Proc. Int'l Conf. Robotics and Automation*, 2016, pp. 3375–3382.
35. J. Allen et al., "PLOW: A Collaborative Task Learning Agent," *Proc. 22nd AAAI Conf. Artificial Intelligence*, 2007, pp. 1514–1519.
36. N.A. Taatgen, "The Nature and Transfer of Cognitive Skills," *Psychological Rev.*, vol. 120, 2013, pp. 439–471.
37. J. Blythe, "Task Learning by Instruction in Tailor," *Proc. Conf. Intelligent User Interfaces*, 2005, pp. 191–198.
38. A. Azaria, J. Krishnamurthy, and T. Mitchell, "Instructable Intelligent Personal Agent," *Proc. 30th AAAI Conf. Artificial Intelligence*, 2016, pp. 2681–2689.
39. K. Barker et al., "Learning by Reading: A Prototype System, Performance Baseline, and Lessons Learned," *Proc. 22nd AAAI Conf. Artificial Intelligence*, 2007; www.cs.utexas.edu/users/mfkb/papers/AAAI13BarkerK.pdf.
40. K. Forbus et al., "Integrating Natural Language, Knowledge Representation and Reasoning, an Analogical Processing to Learn by Reading," *Proc. 22nd AAAI Conf. Artificial Intelligence*, 2007, pp. 1542–1547.
41. K. Lockwood and K. Forbus, "Multimodal Knowledge Capture from Text and Diagrams," *Proc. 5th Int'l Conf. Knowledge Capture*, 2009, pp. 65–72.
42. C. Sammut et al., "Learning to Fly," *Proc. 9th Int'l Conf. Machine Learning*, 1992, pp. 385–393.
43. C. Chao, M. Cakmak, and A.L. Thomaz, "Towards Grounding Concepts for Transfer in Goal Learning from Demonstration," *Proc. IEEE Int'l Conf. Development and Learning*, 2011, pp. 1–6.
44. T. Hinrichs and K. Forbus, "Learning Qualitative Models via Demonstration," *Proc. 26th AAAI Conf. Artificial Intelligence*, 2012, pp. 1430–1438.
45. A. Barbu, S. Narayanaswamy, and J.M. Siskind, "Learning Physically-Instantiated Game Play through Visual Observation," *Proc. IEEE Int'l Conf. Robotics and Automation*, 2010, pp. 1879–1886.
46. Ł. Kaiser, "Learning Games from Videos Guided by Descriptive Complexity," *Proc. 26th AAAI Conf. Artificial Intelligence*, 2012, pp. 963–970.
47. M.N. Nicolescu and M.J. Mataric, "Natural Methods for Robot Task Learning: Instructive Demonstrations, Generalization and Practice," *Proc. 13th Int'l Conf. Autonomous Agents and Multiagent Systems*, 2003, pp. 241–248.
48. T. Könik and J.E. Laird, "Learning Goal Hierarchies from Structured Observations and Expert Annotations," *Machine Learning*, vol. 64, 2006, pp. 263–287.
49. M. van Lent and J.E. Laird, "Learning Procedural Knowledge through Observation," *Proc. Int'l Conf. Knowledge Capture*, 2001, pp. 179–186.
50. D. Bentivegna, C. Atkeson, and C. Gordon, "Learning Similar Tasks from Observation and Practice," *Robotics and Autonomous Systems*, vol. 47, 2004, pp. 163–169.
51. D.D. Salvucci, "Integration and Reuse in Cognitive Skill Acquisition," *Cognitive Science*, vol. 37, 2013, pp. 829–860.
52. M. Klenk and K. Forbus, "Domain Transfer via Cross-Domain Analogy," *Cognitive Systems Research*, vol. 10, 2009, pp. 240–250.
53. M. Klenk and K. Forbus, "Exploiting Persistent Mappings in Cross-Domain Analogical Learning of Physical Domains," *Artificial Intelligence*, vol. 195, 2013, pp. 398–417.
54. R. Pew and A. Mavor, eds., *Modeling Human and Organizational Behavior: Application to Military Simulations*, Nat'l Academies Press, 1998.
55. G.L. Zacharias, J. MacMillan, and S.B. Van Hemel, eds., *Behavioral Modeling and Simulation: From Individuals to Societies*, Nat'l Academies Press, 2008.
56. A. Nareyek, "Game AI Is Dead. Long Live Game AI! ," *IEEE Intelligent Systems*, vol. 22, 2007, pp. 9–11.
57. M. Merrill, "Where's Massive? Past, Present and Future for 'The Lord of the Rings' Crowd Software," *Metro Magazine*, 2004; https://www.questia.com/magazine/1G1-112861526/where-s-massive-past-present-and-future-for-the.
58. R.E. Wray et al., "Requirements for Future SAFs: Beyond Tactical Realism," *Proc. Interservice/Industry Training, Simulation, and Education Conf.*, 2015.
59. Y. Chang et al., "Learning and Evaluating Human-Like NPC Behaviors in Dynamic Games," *Proc. 7th AAAI Conf.*

Artificial Intelligence and Interactive Digital Entertainment, 2011, pp. 8–13.
60. J. Permar and B. Magerko, "A Conceptual Blending Approach to the Generation of Cognitive Scripts for Interactive Narrative," *Proc. 9th AAAI Conf. Artificial Intelligence and Interactive Digital Entertainment Conf.*, 2013, pp. 44–50.
61. J. Ball et al., "The Synthetic Teammate Project," *Computational & Mathematical Organization Theory*, vol. 16, 2010, pp. 271–299.
62. J. Ball, "A Bi-Polar Theory of Nominal and Clause Structure and Function," *Ann. Rev. Cognitive Linguistics*, vol. 5, 2007, pp. 27–54.
63. J. Ball, "A Naturalistic, Functional Approach to Modeling Language Comprehension," *Papers from the AAAI Fall 2008 Symp., Naturally Inspired Artificial Intelligence*, 2008, pp. 1–8.
64. S. Rodgers et al., "Toward a Situation Model in a Cognitive Architecture," *Computational and Mathematical Organization Theory*, vol. 19, 2013, pp. 313–345.
65. J.R. Anderson et al., "An Integrated Theory of the Mind," *Psychological Rev.*, vol. 111, 2004, pp. 1036–1060.
66. N.A. Taatgen and F.J. Lee, "Production Compilation: A Simple Mechanism to Model Complex Skill Acquisition," *Human Factors: J. Human Factors and Ergonomics Soc.*, vol. 45, 2003, pp. 61–76.
67. C. Lebiere, F. Jentsch, and S. Ososky, "Cognitive Models of Decision Making Processes for Human-Robot Interaction," *Proc. Int'l Conf. Virtual, Augmented and Mixed Reality*, R. Shumaker, ed., Springer-Verlag, 2013, pp. 285–294.
68. R. Thomson, C. Lebiere, and S. Bennati, "Human, Model and Machine: A Complementary Approach to Big Data," *Proc. Workshop Human Centered Big Data Research*, 2014, p. 27.
69. B. MacLaren and K.R. Koedinger, "When and Why Does Mastery Learning Work: Instructional Experiments with ACT-R SimStudents," *Proc. 6th Int'l Conf. Intelligent Tutoring Systems*, 2002, pp. 355–366.
70. C.J. MacLellan, K.R. Koedinger, and N. Matsuda, "Authoring Tutors with SimStudent: An Evaluation of Efficiency and Model Quality," *Proc. 12th Int'l Conf. Intelligent Tutoring Systems*, 2014, pp. 551–560.
71. N. Matsuda et al., "Predicting Students' Performance with SimStudent That Learns Cognitive Skills from Observation," *Proc. Int'l Conf. Artificial Intelligence in Education*, R. Luckin, K.R. Koedinger, and J. Greer, eds., IOS Press, 2007, pp. 467–476.
72. N. Matsuda et al., "Cognitive Anatomy of Tutor Learning: Lessons Learned with SimStudent," *J. Educational Psychology*, vol. 105, 2013, pp. 1152–1163.
73. R.J. Heuer and R.H. Pherson, *Structured Analytic Techniques for Intelligence Analysis*, CQ Press, 2010.