Advances in Cognitive Systems 6 (2018) 1–16 Submitted 6/2018; published 8/2018

A Framework for Natural Cognitive System Training Interactions

Christopher J. MacLellan[1] CHRIS.MACLELLAN@SOARTECH.COM
Soar Technology Inc., 3600 Green Court, Suite 600, Ann Arbor, MI 48105 USA

Erik Harpstead[1] EHARPSTE@CS.CMU.EDU
Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213 USA

Robert P. Marinier III BOB.MARINIER@SOARTECH.COM
Soar Technology Inc., 3600 Green Court, Suite 600, Ann Arbor, MI 48105 USA

Kenneth R. Koedinger KOEDINGER@CMU.EDU
Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA 15213 USA

Abstract

Researchers have developed cognitive systems capable of human-level performance at complex tasks, but constructing these systems required substantial time and expertise. To address this challenge, a new line of research has begun to coalesce around the concept of cognitive systems that users can teach rather than program. A key goal of this research is to develop natural approaches for end users to directly train these systems to perform new tasks. However, there does not currently exist a language for describing the key components of cognitive system training interactions and how these components relate to the concept of naturalness for end users. This paper begins to explore this gap. To lay the foundation for this exploration, we review relevant prior machine learning and interaction frameworks as well as the human-computer interaction literature to identify characteristics of systems that have historically been natural for end users to interact with. Based on this review, we propose the Natural Training Interactions (NTI) framework, which decomposes cognitive system training interaction into patterns, types, and modalities, all of which support the acquisition of different kinds of knowledge. Finally, we discuss how this framework characterizes existing research within this space and how it can guide future research.

1. Introduction

In recent years, there has been a growth of research and development in the area of cognitive systems (Langley, 2012), with prior demonstrations showing that it is possible for cognitive systems to achieve human-level performance at complex tasks (e.g., Jones et al., 1999; Mcdermott, 1980). However, cognitive systems still remain largely out of reach for the general public (Laird et al., 2017). A major factor contributing to this disconnect is that our daily lives are filled with a wide range of tasks across multiple domains, whereas today's state-of-the-art cognitive systems are implemented to perform specific tasks in specific domains. Extending specialized cognitive systems to support a wider range of tasks requires substantial time and expertise. For example, the

[1] The first two authors contributed equally to the paper and should both be considered its first authors.

base IBM Watson system that famously beat two Jeopardy! champions required over a century of artificial intelligence expert development time.

To address this challenge, cognitive systems researchers have begun exploring approaches for users to create and extend the capabilities of cognitive systems by teaching them, rather than by programming them. This emerging area of research, which includes approaches such as Interactive Task Learning (Kirk & Laird, 2014; Laird et al., 2017) and Apprentice Learning (Maclellan, 2017; Maclellan, Harpstead, Patel, & Koedinger, 2016), aims to develop the computational and cognitive theory needed for building systems that support natural interactions and that possess general capabilities for learning across a wide range of domains and contexts. Similar to how research and development on computing hardware enabled the transition from corporate mainframes to personal computers, this research area aims to support the transition from monolithic cognitive systems (e.g., Watson) to personal cognitive systems (e.g., companion agents, Forbus & Hinrichs, 2006).

The literature contains several examples of teachable cognitive systems (Hinrichs & Forbus, 2014; Kirk & Laird, 2014; Maclellan et al., 2016) and each one implicitly instantiates a training paradigm. However, we lack a common language for discussing existing training paradigms, engaging in scientific discourse over their structures, and for supporting the systematic translation and reuse of their components. In this paper, we take the first steps towards creating such a language, which we present in the form of a framework. Additionally, prior work has generally focused on the perspective of the learning agent and its mechanisms rather than the teaching user and their interactions. Instead, we adopt a user-centered approach for teaching cognitive systems and center our framework around the question of what makes training interaction natural for human teachers. In doing so, we draw on the human-computer interaction perspective that an understanding of interaction is central to the design and development of usable technology. Ultimately, we intend this work to lay the foundation for the development of personal cognitive systems that users can naturally teach.

2. Why Develop a Framework?

The notion of decomposing human-agent interactions using a framework is not novel in itself and several decompositions exist in the literature (Bartneck & Forlizzi, 2004; Laird et al., 2017; Sheridan, 1992). However, we have found that existing frameworks provide an insufficient theoretical and practical basis for describing existing teachable cognitive systems and ultimately for designing novel cognitive system training paradigms that are natural and efficient for end users.

For example, one of the most common distinctions in the machine learning literature is between supervised, unsupervised, and semi-supervised approaches (Bishop, 2016). While this distinction appears to refer to the amount of training required by users and to map each level of supervision to the appropriate class of learning algorithms, the classical supervision dimension is actually independent of the user—it distinguishes on whether there are privileged attributes (typically prediction attributes) and whether ground truth values are available for them. From a cognitive system design perspective, it is possible to use supervised approaches without user involvement, if ground truth labels are available by other means, and unsupervised approaches with user involvement, if the user is annotating examples. Thus, these traditional distinctions are misaligned for providing guidance on the design of cognitive system training paradigms. Other traditional

machine learning distinctions, such as incremental vs. non-incremental or classification vs. regressions, suffer from similar problems—whether a system is using one approach or another is largely indistinguishable to users. In general, these kinds of machine-learning distinctions place the emphasis on the learning mechanisms rather than on the user and their training interactions.

A more relevant decomposition comes from Laird et al.'s (2017) review of interactive task learning, which divides task learning approaches by the mode of communication (natural language or demonstration) and the type of knowledge taught (goals, concepts, actions, and procedures). We view these distinctions as much more practical for cognitive system design because they better align with training interaction design choices (e.g., which modality a user experiences). Additionally, this distinction provides a basis for constructing theoretical hypotheses regarding the relationship between modality and type of knowledge being transferred. However, we claim that more guidance is needed. For example, this prior work provides little discussion regarding how to structure training of particular kinds of knowledge within particular modes. What is needed are additional components to describe when and how the user and system should act in different situations, such as interaction patterns. Additionally, we argue that Laird et al.'s modality should be further distinguished between the type of interaction being performed and the modality it takes because it is possible for interactions to be communicated via different modalities, such as a demonstration (an interaction type) being communicated using sketch, speech, or a graphical user interface (different modalities).

Another related line of work is Bartneck and Forlizzi's (2004) human-robot interaction framework, which has categories for patterns—called norms—and modalities. However, this framework focuses on robot's social interactions with humans more generally, rather than training interactions specifically, and so does not have dimensions for the types of knowledge being taught as seen in Laird et al.'s (2017) review. Additionally, like the Laird et al. work, it lacks a dimension for interaction types, which we claim provides an important intermediate layer of abstraction between patterns and modalities. Finally, as their work emphasizes physical robotics it also contains dimensions that are related to a robot's physicality (e.g., whether the robot's form is abstract or anthropomorphic). Our work is less concerned with the physical embodiment of agents, but it is not incompatible with our current thinking.

We also believe it is worth mentioning Sheridan's (1992) supervisory control framework and VanLehn's (2006) model of tutoring system behavior because they provide a broader context for training interactions. In particular, Sheridan views "teaching" as an important component of effective human supervisory control systems, along with "planning", "monitoring", "intervening" and "learning." However, his work primarily describes teaching as the process by which a human supervisor directly programs a system with the desired knowledge. Our work aims to provide a means by which human operators might more naturally teach a system without the technical knowledge necessary to program it—ultimately enabling the creation of more effective supervisory control systems. In contrast to Sheridan's work, which explores the situation of a human teacher and a machine student, VanLehn describes the behavior of computer tutoring systems that teach human students. His work distinguishes between the outer teaching loop, where a teacher models student knowledge over time and selects problems for a student to perform that appropriately challenge their current abilities, and the inner teaching loop, where a teacher provides immediate instruction and guidance to students regarding how to correctly complete each problem. In general,

our current work focuses primary on the inner teaching loop—or the training interactions with a system on a situational basis—and does not attempt to characterize how a teachable system should situates within the outer teaching loop, which would include estimating system knowledge or deliberately select training problems, or within supervisory control more broadly, which would include additional issues such as monitoring and control. However, we do acknowledge that any teachable system will ultimately be embedded within these greater contexts.

Given the context of this prior work, our goals in proposing a new framework are three fold. First, we hope to draw attention to the exciting nexus of cognitive system and human-computer interaction research, which ultimately has the potential to increase the accessibility and broader adoption of cognitive systems technology. Second, we aim to provide a common language for describing cognitive system training interactions that enables the framing of hypotheses regarding which means of training are more natural and efficient for end users in different situations. Finally, we intend our framework to support the design of cognitive systems by defining the space of training interactions components that can be translated and reused across different systems. While many existing frameworks share commonalities with the one we propose here, their focus is either more general (interaction broadly) or directed toward different kinds of interaction (non-training interactions). Thus, we aim to combine the best of these prior ideas in our framework and present a novel perspective on interaction that is better aligned with our high-level goal of building cognitive systems that are natural for end users to train.

3. What Makes an Interaction Natural?

In order to create an initial framework for natural training interactions, we must first contend with what it means for an interaction to be natural. While it is common to think of gesture and speech as lending naturalness to an interaction, the prior literature highlights that an interaction is not necessarily natural by virtue of its physical modality. Norman (2010) argues that so called natural user interfaces (e.g., speech- and gesture-based) are not inherently more natural than graphical user interfaces (e.g., screen-based widgets). For example, gestural interfaces lack the affordances to let users know what gestures they support, whereas graphical user interface widgets, such as buttons, readily advertise their supported interactions. In general, this work suggests that the naturalness of a modality alone is neither necessary nor sufficient for making an overall interaction natural.

Given that naturalness does not derive from modality, then what makes interaction natural? To address this question, we reviewed the HCI literature on natural interactions and identified four common characteristics of systems that support naturalness: they (1) support the goals of users, (2) do what users expect, (3) lets users work the way they want, and (4) leverage users' experience to minimize training. In this section, we review each of these characteristics.

Supports the goals of users. Systems supporting natural interactions should be able to support what users want to do (i.e., their goals). One temptation in developing these systems is to overemphasize ease of use at the expense of limiting what users can achieve. Myers, Hudson, and Pausch (2000) refer to this balance as the threshold and ceiling of tools. Thresholds refer to the barriers a user must overcome to use a tool, whereas the ceiling describes what the tool enables users to do. Many systems attempting to support natural interactions emphasize a low threshold, but often ignore the ceiling. For example, it is easy to interact with Apple's Siri, but it only supports

built-in commands—it is unable to learn new commands. To overcome this risk, systems should be developed with end-user goals and intents in mind (e.g. the desire to teach Siri new user-defined commands), so that the developers can ensure the system does not limit users' capabilities. Typically, increasing the generality of a learning system comes at the cost of reduced system robustness; however, a key point of this prior work is that developers can strike a better balance between these two outcomes by building systems to support the *right* subset of users goals, rather than simply supporting *more* goals.

Does what users expect. A common theme in research on natural interactions is an emphasis on the expectations users have for a system (Myers, Pane, & Ko, 2004). Humans typically follow patterns, scripts, or norms when engaging in everyday interactions (Bicchieri, 2006), which make it possible for the humans involved in the interaction to know how to respond. For example, tutors generally expect that their pupils will attempt to solve problems before asking for help. Systems that aspire to naturalness should support naturally occurring patterns of interaction and be aware of users' expectations within these patterns. It is worth noting that these patterns may arise from a user's particular cultural background (e.g., what roles their culture ascribes to teachers and students) or from their personal experiences (e.g., whether they are a Mac or PC user). Additionally, systems attempting to be natural should not require users to learn new (unnatural) patterns of interaction—deviations from typical scripts make it difficult for users to know what the system will do next and how to respond accordingly.

Lets users work the way they want. Given that natural systems support users' goals they should also let users execute those goals the ways they prefer or expect to. A key idea from the ubiquitous computing literature is that computing systems should become invisible because they seamlessly support the ways users want to do something (Weiser & Brown, 1996). They should not impede users or force them to achieve goals in unpreferred ways. For example, a common trend is to build systems around a speech interaction paradigm, but there are many situations where speech is an unnatural form of communication. In his study of architectural designers, Schön (1983) found that sketches of designs often better supported communication and reasoning than verbal articulations. This finding suggests that systems aiming to support natural architectural design should prefer sketch-based interactions over speech.

Leverages users experience to minimize necessary training. One of the most pervasive ideas within research on natural user interfaces is the idea of instant expertise (Wigdor & Wixon, 2011), or the idea that users should not have to learn how to control a system because the modality used is one they have immediate familiarity with. In the words of Buxton (Larsen, 2010), "[*natural user interfaces] exploit skills that we have acquired through a lifetime of living in the world, which minimizes the cognitive load and therefore minimizes the distraction."* Common approaches within this space include voice- and text-based natural language and gestural interfaces that take advantage of users' lived experiences interacting with other people. Additionally, many users have extensive training with artificial interfaces, such as QWERTY keyboards, that may be natural for many application contexts, so it is worth noting that these artificial modes of interaction should not be discounted.

Table 1. The natural training interactions framework.

Knowledge	Patterns	Types	Modalities
• Goals • Beliefs • Concepts • Experiences • Skills • Dispositions	• Passive Learning • Operant Conditioning • Direct Instruction • Apprentice Learning • After-Action Review • Collaborative Learning • Programming	• Command • Clarify • Acknowledge • Inform • Spotlight • Annotate • Reward • Demonstrate • Direct knowledge manipulation • Request <type>	• Command Line • Control device • GUI • Sketch • API • Gesture • Speech • Text • Multi-modal

4. A Preliminary Framework for Cognitive System Training Interactions

In order to design cognitive systems that support natural training interactions, we require a better understanding of how these systems could hypothetically interact. In this section, we propose a framework for characterizing cognitive system training interactions—the Natural Training Interactions (NTI) framework, see Table 1—that aligns with the four characteristics noted in the previous sections. In putting forth this framework, we do not intend for this work to be complete but hope that it provides a useful initial language to start talking about naturalness in the context of cognitive systems and their instructional interactions with users.

At a high-level, the framework is built on the assumption that the goal of training is to change some aspect of an agent's ***knowledge***. To update knowledge, agents and trainers interact according to instructional ***patterns***. Within patterns, trainers and agents employ in several ***types*** of interaction, and these interactions can be done through various ***modalities***. We review each aspect of the framework in turn.

Knowledge. We assume that the goal of any training interaction is to update the learner's knowledge. There are many types of knowledge that might be included in a cognitive system. Within the literature, there are several generally accepted types of knowledge (Laird, Lebiere, & Rosenbloom, 2017). For our preliminary framework, we include six such kinds of knowledge:

- *goals*, which fully or partially describe desirable states of the world;
- *beliefs*, which represent an agent's current worldview;
- *concepts*, which support semantic inference and enable an agent to augment its worldview with additional non-observable information;
- *experiences*, which organize past situations and problem-solving episodes;
- *skills*, which describe procedures for changing the world and updating beliefs; and
- *dispositions*, which specify an agent's problem-solving orientations (e.g., whether to explore or exploit).

Our current focus is primarily on symbolic forms of knowledge arising from interactions with a trainer, but future extensions of the framework might also include sub-symbolic knowledge (learning probabilistic grammar knowledge for parsing English sentences or equations such as Li, Schreiber, Cohen, & Koedinger, 2012). Further, we do not mean to imply that all cognitive systems must support all of these knowledge categories but rather that the nature of the knowledge being changed will likely dictate choices across the other dimensions of the framework.

Patterns. Within human-human instructional settings there are many naturally occurring interaction and training patterns. These patterns govern the relationship between trainer and trainee and establish the contours for how training interactions play out. Inspired by existing systems (Allen et al., 2007; Forbus & Hinrichs, 2006; Hinrichs & Forbus, 2014; Kirk & Laird, 2014; Maclellan et al., 2016) and models of instructional practice in humans (Chi & Wylie, 2014; Koedinger, Corbett, & Perfetti, 2012), our framework highlights several possible patterns. At its most simple, learning could be primarily *passive*, with agents observing training behaviors without agency or active input from instructors. Increasing complexity, agents can have some control over their actions and receive rewards from an instructor (*operant conditioning*) or instructors can explicitly coach an agent, without requiring agent decision making (*direct instruction*). An even more complex pattern, *apprentice learning* (Maclellan et al., 2016), incorporates aspects of both approaches—both explicit instruction and feedback on agent actions. Additionally, many other instructional patterns are possible, such as *after-action review*, where a trainee reviews past experiences and an instructor provides complementary instruction, *collaborative learning* (Olsen, Belenky, Aleven, & Rummel, 2014) where both actors are trainees and learning from one another, and even *programming*, which consists of directly manipulating knowledge structures and is probably the most prevalent human-computer training pattern.

Types. Within a pattern, an instructor and trainee engage in many types of interactions. For example, within the apprentice learning pattern (Maclellan et al., 2016), an instructor issues a *command*, which specifies the task for an agent to perform. If the agent does not know how to perform the task, then it might *request* a *demonstration* from the instructor, who provides one. On subsequent tasks, the agent might attempt the task (i.e., provide the instructor with a *demonstration*) and *request* feedback (i.e., a *reward*) on this attempt. Finally, the instructor provides the agent with the appropriate *reward*. Under this pattern, this process continues until the agent is correctly performing all of the tasks. Our framework also includes interaction types for supporting the other patterns. For example, Direct Instruction (Hinrichs & Forbus, 2014) allows instructors to directly *inform* agents about the world ("TicTacToe is a two-player game"), *spotlight* agents attention on particular parts of the world ("This [pointing] is a block"), and *annotate* demonstrations ("This is the move action [demonstrate drawing of X on board]") to facilitate efficient learning.

The types listed in Table 1 are drawn from existing systems as well as the literature on communicative acts (Allen, Blaylock, & Ferguson, 2002; Traum & Hinkelman, 1992). These include:

- *commands*, which compel an agent to perform some task or take on some state;
- *clarifications*, which disambiguate between competing interpretations or further elaborate or grounds prior interactions;
- *acknowledgments*, which signal that data has been successfully received and/or processed;
- *inform* acts, which provide an agent with additional information in addition to perception;

- *spotlighting*, which identifies certain elements in the environment as relevant or important;
- *annotations*, which augment an agent's current understanding with additional learning related information, such as skill labels or which elements of an example should be generalized;
- *rewards*, which consist of correctness feedback or a numerical reward;
- *demonstrations*, which are examples of behavior;
- *direct knowledge manipulation*; which describe new knowledge structures or changes to existing structures; and,
- *request <type>*, which are requests to provide any of the above types.

This is not meant to be an exhaustive list, but is representative of the types that commonly occur in current practice. It is important to note that when we refer to interaction types we are interested in the overall instructional act being performed and not how it is being performed. For example, orders delivered via a command line interface or spoken natural language are both instances of the *command* type.

Modalities. The different types of interactions ultimately ground out in particular modalities of interaction, with many different modalities, or potentially multiple simultaneous modalities, supporting each type. For example, command-line or graphical-user interfaces, are both capable of supporting all of the interaction types listed in Table 1. Often, systems that claim to support natural interaction leverage modalities commonly used in human-human interaction as the primary modes of interaction. For example, the Microsoft Kinect supports gesture- and speech-based modalities. A key aspect of modalities from our perspective is that they are cast in terms of what the trainer is doing and not necessarily how an action is being detected by an agent. For example, a gesture such as waving could be detected using either visual sensing with a camera or gyroscopic sensing with a wearable device (e.g., Taylor et al., 2017); in either case, the trainer would be using a gestural modality.

These four dimensions intentionally map to the four characteristics highlighted in the previous section. In particular, in the context of training, supporting a user's goals consists of supporting the types of knowledge transference they are trying to achieve. Users' expectations regarding training will derive from the social instructional patterns they have experience with. Thus, in order to naturally support training interactions, it is important for system designers to be aware of the interaction patterns that users expect. Further, users will want to interact in certain ways and system designers should be aware of the different types of interactions they want to perform. Finally, for each type of interaction, system designers should leverage modalities that draw on users' prior experience.

5. Analysis of Existing Systems

To ground the elements of our framework, we next review eight systems[2] from the literature and identify the *knowledge*, *patterns*, *types*, and *modalities* used by each. Reviewing all of the machine learning systems in the literature is well beyond the scope of this work, so we instead focus on

[2] We acknowledge that some of these systems may not strictly fit Langley's (2012) definition of cognitive systems, but mapping them to our framework is still informative.

highlighting a selection of examples that span the range of patterns from our framework and that emphasize user training interactions more so than their underlying machine-learning approach.

Our first example, is Google's Teachable Machine.[3] This system, which was built to demonstrate neural network learning with Google's TensorFlow, leverages the *direct instruction* pattern. In particular, users teach the system image-related *concepts* (knowledge) by *demonstrating* (type) examples of the concept via the provided webcam-based *GUI* (modality). While this system resembles a more traditional, supervised learning system, it blurs some of the classical machine-learning distinctions. For example, it is not outwardly clear to users whether the system engages in incremental vs. non-incremental learning. From the user perspective, the system appears to be incremental—the user can provide new examples at any time to update the system—but the underlying machine learning approach is opaque from a training interaction design perspective and somewhat irrelevant (as long as the system is responsive). Similarly, it is not clear whether this system is fully supervised; as far as the user is concerned, it could be learning directly from a pixel representation or using more complicated features pre-generated from some form of unsupervised learning (e.g., a pre-trained image autoencoder).

The Q-Learning agent used in the Sophie's room game (Thomaz, Hoffman, & Breazeal, 2006) takes a slightly different perspective on training. Similar to Google's teachable machine, this system emphasizes the teaching interactions rather than the underlying machine-learning approach. However, this system uses the *operant conditioning* pattern to acquire cake baking *skills* (knowledge), where the Q-Learning agent *demonstrates* (type) actions in a simple cooking simulation environment (*GUI*—modality), such as picking or mixing ingredients and users train the agent by providing, potentially delayed, *rewards* (type) via the *GUI* (modality). One interesting aspect of this work is the continuous time nature of the training interactions. In particular, agents can take action at any time and users can independently provide rewards at any time (agents do not need to wait for user rewards or vice versa). Also, while this system utilizes a reinforcement learning mechanism, this work is quite different from the vast majority of reinforcement learning systems, such as AlphaGo (Silver et al., 2016), that get their rewards directly from the environment (or environment simulator). The Sophie's room work explores the use of a reward signal that comes from human trainers rather than an objective signal from the environment. Interestingly, Thomaz et al. (2006) found that users often provided reward signals that violate many of the assumptions underlying most reinforcement learning algorithms. For example, they found that users often provided anticipatory rewards to guide the system towards desired behavior, even though the Q-learning algorithm treats rewards as feedback on prior, not future, actions.[4] It is possible users employed anticipatory rewards to compensate for not being able to work the way they wanted (i.e., directly show the agent the desired behavior via demonstration).

Another, system that supports natural training interactions is SUGILITE (Li, Azaria, & Myers, 2017). This system learns *skill* knowledge for performing tasks in arbitrary smartphone apps from *commands* and *demonstrations* (types) presented in *mixed GUI + speech* modalities. This differs from the prior systems by employing two patterns, *direct instruction* and *programming*, to support efficient training. Direct instruction starts when users issue a *command* (type) to the system using

[3] See `https://teachablemachine.withgoogle.com/`

[4] Users were told this in advance, but they had no technical machine-learning expertise and likely did not understand the training implications.

speech (modality). If the system does not know how to execute the command, the user *demonstrates* (type) the behavior that should be associated with the command and *annotates* (type) the demonstration via a provided GUI (modality) to specify which parts of the demonstration should be generalized (i.e., which interface constants should be replaced with variables). If the user issues the associated command in the future, then the system *demonstrates* (type) the associated generalized behavior back. In the event that the system exhibits undesirable behavior, the user can activate the *programming* pattern, which allows users to *directly manipulate the learned knowledge* (type) via a *GUI* (modality); e.g., users might delete a skill or replace a variable in a previously learned skill with a specific constant. SUGILITE provides this pattern for more technically advanced users, so that they can efficiently teach the system the behavior they want.

Another example of a trainable system is Betty's Brain (Leelawong & Biswas, 2008), which was designed for use in K12 education. Like SUGILITE, Betty's Brain adopts the *programming* pattern, wherein users teach Betty causal scientific *concepts* (knowledge) by *directly manipulating her knowledge* (type) via the provided concept mapping *GUI* (modality). This system was designed to exploit the learning-by-teaching phenomenon. Specifically, students learn causal scientific concepts by teaching them to Betty. One interesting aspect of this system is its central use of the programming pattern with non-technical K12 users. The computational aspect of the learning system enables the use of a pattern that is impossible with human-human learning. Despite the lack of a human analog, the younger users are able to efficiently teach Betty using this pattern, showing that natural patterns need not draw completely from human experience.

Rosie (Kirk & Laird, 2014; Mohan, Mininger, Kirk, & Laird, 2012) is a somewhat different example of a cognitive system that supports natural training interactions. Unlike the prior systems, Rosie can acquire *goal*, *concept*, and *skill* knowledge from *mixed GUI* and natural language *text* (modality) interactions with users. This system learns via a variant of the *direct instruction* pattern. Most instances of the direct instruction pattern center around the teacher *informing* or *demonstrating* to the student, with the trainee *requesting clarifications* occasionally. Interestingly, Rosie inverts this emphasis, with most interaction centering around *clarifications*.[5]

For example, when learning games (Kirk & Laird, 2014), interaction starts when the teacher *informs* (type) Rosie of a game, e.g., "the game is tic-tac-toe" in natural language *text* (modality). If Rosie does not know the game, then she *requests clarifications* (type) on characteristics of the game to build up a complete description of it. For example, she might say, "I don't know that game, how many players are there?" (*text*—modality). In response, the user might reply "two" (*clarification*—type, *text*—modality). Rosie also *requests clarifications* (type) on legal game actions ("please start by teaching me the name of a legal action in the game") as well as their parameters ("what are the verb and parameter arguments associated with this action?") and constrains ("Please list all constraints for this parameter, such as 'it is red' or 'it is on [parameter] 2', and then finished").

In another example (Mohan et al., 2012), Rosie uses a similar pattern to acquire concept and goal knowledge. In this case, the user might start interaction by issuing a *command* (type) for Rosie to perform, such as "Store the orange object" (*text*—modality). In response, if Rosie does not know

[5] The authors debated whether the direct instruction label is still appropriate with this inversion and ultimately decided it was because Rosie's adopts more conventional direct instruction behavior when she already has ample existing knowledge and does not require as much clarification.

the attribute "orange", then she might ask "what kind of attribute is orange?" (*request for clarification*—type) to which the user might respond "a color" (*clarification*—type). Rosie also supports some mixed-modality interaction. In particular, she might say "I don't see an orange object. Please teach me to recognize one" (*request for clarification*—type), to which the user might click on a GUI object and say "this is orange" (*clarification*—type, *mixed GUI + text*—modality).

In general, Rosie uses this clarification-guided approach to progressively transfer and ground concept, skill, and goal knowledge. It is worth noting that although Rosie supports interactive training, we do not classify her training paradigm as *apprentice learning* because she never requests or receives rewards on her own task-related decision making. One interesting aspect of this work is its almost universal use of a natural language text modality. From a user perspective, it seems like it would be difficult to transfer skill knowledge using this modality because tacit knowledge, such as skills or experiences, are noted for being challenging for people to textually or verbally articulate (Polanyi, 1966; Schön, 1983). In contrast, the text modality seems well suited for transferring concept and goal knowledge, which should be easier for people to verbally articulate. In either case, the alignment between modality and the type of knowledge being transferred should be investigated in future work.

Another related cognitive system is the Companion Architecture (Hinrichs & Forbus, 2014). This system also uses natural language text for interaction and leverages a *direct instruction* pattern. However, it differs in some key respects. Specifically, it utilizes a *multi-modal text + sketch* interface (modality) and the system only acquires *concepts* (knowledge). In this system, users *inform* (type) the system of different concepts via text statements and sketch demonstrations and the system *acknowledges* (type) that it correctly interpreted these actions. For example, a user might inform the system that "Tic-Tac-Toe is a 2 player game" or "X is a player <draws X in the sketch interface>", to which the system would respond "OK" in both cases. Once the system has acquired these concepts, the system compiles them into the Game Definition Language (Genesereth & Thielscher, 2014), a representation it leverages to actually play games with the user.

Like Betty's Brain, SimStudent and the Apprentice Learning Architecture were built to support K12 education. These systems employ the *apprentice learning* pattern to support a wide range of applications, such as modeling students learning in tutors (Li, Matsuda, Cohen, & Koedinger, 2010; Maclellan et al., 2016; Matsuda, Lee, Cohen, & Koedinger, 2009), studying the learning-by-teaching effect (Matsuda, Yarzebinski, Keiser, Cohen, & Koedinger, 2011), and supporting efficient tutor authoring (Li, Stampfer, Cohen, & Koedinger, 2013; MacLellan, Koedinger, & Matsuda, 2014; Matsuda, Cohen, & Koedinger, 2014). Users train an agent by presenting it with particular problems to solve—implicit *commands* (type)—within a tutor *GUI* (modality). When an agent lacks knowledge of how to solve the problem it *requests a demonstration* (type) from the user by popping up a GUI dialog message (modality). The user then *demonstrates* (type) a problem-solving step directly in the tutor *GUI* (modality). From these demonstrations the agent learns new problem-solving *skills* (knowledge) and, in the future, *demonstrates* (type) steps on similar problems for the user. After every agent demonstration, the system *requests a reward* (type) from the user, who provides correctness feedback (*reward*—type) via the provided *GUI* (modality). This system is effectively an active learning system (Settles, 2012) that interactively requests feedback and examples from users when it needs them.

The final system that we reviewed was PLOW (Allen et al., 2007). Although this system claims to be a collaborative problem-solving system, the system actually employs the *apprentice learning* pattern. Using this pattern, the system acquires *skills* (knowledge) from *mixed text + GUI* (modality) interactions. Users start interacting with the system by issuing a *command* (type) via natural language *text* (modality), such as asking the system to "list all the hotels within 20 miles." If the system already has the appropriate knowledge for the task, then it *demonstrates* (type) behavior directly in the task *GUI* (modality); e.g., it looks up and lists the hotels within 20 miles. If the demonstrations are incorrect than the user provides negative feedback, or *reward* (type), to the system by telling it "this is wrong" (*text*—modality). If the system does not know what to do, then the user provides a *demonstration* (type) of the desired behavior, such as looking up the appropriate list of hotels in the provided web-browser GUI (modality) and an *annotation* (type) on the demonstration by specifying, "let me show you how to list the hotels within 20 miles" (*text*—modality). Additionally, when the user provides demonstrations, the system can also *request clarifications* (type) related to the demonstrations.

Allen et al. (2007) performed preliminary user testing where they compared PLOW to variants that used different training interaction patterns. Specifically, one of the systems learned by passively observing users behavior within the GUI (the *passive learning* pattern), one had a sophisticated GUI for authoring task macros (the *programming* pattern), and one had users describe skills to the system completely in natural language (the *direct instruction* pattern). Allen et al. found that users preferred the base system to its variants and that it performed better on some initial usability metrics. Although not conclusive, these preliminary findings suggest that the *apprentice learning* pattern (used by PLOW) may be more appropriate for skill learning than the *passive learning* or *direct instruction* patterns.

6. Discussion and Future Work

In proposing this initial framework, we aim to achieve three objectives. First, we attempt to highlight what we view as a key opportunity within cognitive systems research: to better understand the space of training interaction and develop cognitive systems that are natural and efficient for users to teach and interact with. Recent research efforts, such as Rosie (Kirk & Laird, 2014), the Companion Architecture (Hinrichs & Forbus, 2014), and the Apprentice Learning Architecture (Maclellan et al., 2016), have begun exploring different combinations of patterns, types, and modalities to support training interactions with end users. Each of these systems represent particular choices across the dimensions of our framework. To reach a more complete understanding of training interaction design, researchers should explore additional approaches and new combinations of approaches in order to explore the space more broadly and ultimately direct work toward designing more natural means for training cognitive systems.

Second, organizing training interactions along an orthogonal set of dimensions enables a modular approach to the challenge of building cognitive systems to support natural training interactions. Individual researchers or developers need not contend with the whole problem and can instead focus on addressing subproblems. For example, one team of researchers might investigate which patterns are best for acquiring skills, whereas another team might investigate which patterns are best for acquiring concepts. Because these decisions are orthogonal, both teams can benefit from

each other's work and integrate their findings within the common structure of the framework to support the development of systems that can naturally learn both skills and concepts. Thus, the framework supports the unification of independent research efforts, even if these efforts do not explicitly describe their work within this framework.

Finally, towards the goal of actually building cognitive systems that people can naturally train, we intend our framework to provide a language for formulating scientific hypotheses about how such systems should interact with users to best achieve naturalness. Much of the existing work implicitly assumes that choosing natural approaches for only one of the components of the framework (patterns, types, or modalities) establishes the overall naturalness of a system. For example, Hinrichs and Forbus (2014) emphasize the use of multiple natural modalities, such as text and sketching, whereas MacLellan et al. (2016) emphasize the use of a natural pattern. Central to our framework, however, is the hypothesis that different combinations of patterns, types, and modalities of interaction are better suited for updating different kinds of knowledge.

Thus, we believe that systems that are natural for users to teach will not only support a wide range of patterns, types, and modalities, but flexibly choose the appropriate combination based on the type of knowledge being communicated, the trainer's preference, and potentially other contextual factors. There is evidence that learning in humans follows a similar logic, in that different kinds of knowledge are best taught by different forms of instruction (Koedinger et al., 2012). Given that an artificial intelligence need not represent a natural system, there is no inherent reason to transfer this logic (Simon, 1983). However, if we want to support humans in naturally training such systems, then it becomes important to understand these relationships and how they might impact different kinds of training. In conclusion, it is our hope that this framework will focus attention on this issue, provide a language for talking about training interactions and their naturalness, and guide future research on this exciting frontier of personal cognitive systems.

Acknowledgements

We thank the organizers and participants of the AAAI 2018 Spring Symposium on the User Experience of Artificial Intelligence, who provided feedback on an earlier version of this work. This material is based upon work supported by the OFFICE OF NAVAL RESEARCH under Contract No. N68335-18-C-0401. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the OFFICE OF NAVAL RESEARCH.

References

Allen, J. F., Blaylock, N., & Ferguson, G. (2002). A problem solving model for collaborative agents. *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems* (pp. 774–781). New York: ACM.

Allen, J. F., Chambers, N., Ferguson, G., Galescu, L., Jung, H., Swift, M., & Taysom, W. (2007). PLOW: A collaborative task learning agent. *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence* (pp. 1514–1519). Menlo Park, CA: AAAI Press.

Bartneck, C., & Forlizzi, J. (2004). A design-centred framework for social human-robot interaction. *Proceedings of the Thirteenth IEEE International Workshop on Robot and Human Interactive Communication* (pp. 591–594). New York: IEEE Press.

Bicchieri, C. (2006). *The grammar of society: The nature and dynamics of social norms*. New York: Cambridge University Press.

Bishop, C. M. (2016). *Pattern recognition and machine learning*. New York: Springer-Verlag.

Chi, M. T. H., & Wylie, R. (2014). The ICAP framework: Linking cognitive engagement to active learning outcomes. *Educational Psychologist*, *49*, 219–243.

Forbus, K. D., & Hinrichs, T. R. (2006). Companion cognitive systems: A step toward human-level AI. *AI Magazine*, *27*, 83–95.

Genesereth, M., & Thielscher, M. (2014). *General Game Playing*. San Rafael, CA: Morgan & Claypool.

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

Jones, R. M., Laird, J. E., Nielsen, P. E., Coulter, K. J., Kenny, P., & Koss, F. V. (1999). Automated intelligent pilots for combat flight simulation. *AI Magazine*, *20*, 27–41.

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

Koedinger, K. R., Corbett, A. T., & Perfetti, C. (2012). The knowledge-learning-instruction (KLI) framework: Toward bridging the science-practice chasm to enhance robust student learning. *Cognitive Science*, *36*, 757–798.

Laird, J. E., Gluck, K., Anderson, J., Forbus, K. D., Jenkins, O. C., Lebiere, C., … Kirk, J. R. (2017). Interactive task learning. *IEEE Intelligent Systems*, *32*, 6–21.

Laird, J. E., Lebiere, C., & Rosenbloom, P. S. (2017). A standard model of the mind: Toward a common computational framework across artificial intelligence, cognitive science, neuroscience, and robotics. *AI Magazine*, *38*, 13–26.

Langley, P. (2012). The cognitive systems paradigm. *Advances in Cognitive Systems*, *1*, 3–13.

Larsen, L. (2010). CES 2010: NUI with Bill Buxton. Retrieved April 10, 2017, from `https://channel9.msdn.com/Blogs/LarryLarsen/CES-2010-NUI-with-Bill-Buxton`

Leelawong, K., & Biswas, G. (2008). Designing learning by teaching agents: The Betty's Brain system. *International Journal of Artificial Intelligence in Education*, *18*, 181–208.

Li, N., Matsuda, N., Cohen, W. W., & Koedinger, K. R. (2010). Towards a computational model of why some students learn faster than others. *Proceedings of the AAAI Fall Symposium on the Cognitive and Metacognitive Educational Systems* (pp. 40–46). Menlo Park, CA: AAAI Press.

Li, N., Schreiber, A. J., Cohen, W. W., & Koedinger, K. R. (2012). Efficient complex skill acquisition through representation learning. *Advances in Cognitive Systems*, *2*, 149–166.

Li, N., Stampfer, E., Cohen, W. W., & Koedinger, K. R. (2013). General and efficient cognitive model discovery using a simulated student. *Proceedings of the Thirty-Fifth Annual Meeting of the Cognitive Science Society* (pp. 894–899). Austin, TX: Cognitive Science Society.

Li, T. J.-J., Azaria, A., & Myers, B. A. (2017). SUGILITE: Creating multimodal smartphone automation by demonstration. *Proceedings of the 2017 CHI Conference on Human Factors in*

Computing Systems (pp. 6038–6049). New York: ACM.

Maclellan, C. J. (2017). *Computational models of human learning: Applications for tutor development, behavior prediction, and theory testing*. Doctoral dissertation, Human-Computer Interaction Institute, Carnegie Mellon University, Pittsburg, PA.

Maclellan, C. J., Harpstead, E., Patel, R., & Koedinger, K. R. (2016). The Apprentice Learner architecture: Closing the loop between learning theory and educational data. *Proceedings of the Ninth International Conference on Educational Data Mining* (pp. 151–158). Boston, MA: International Educational Data Mining Society.

MacLellan, C. J., Koedinger, K. R., & Matsuda, N. (2014). Authoring tutors with SimStudent: An evaluation of efficiency and model quality. *Proceedings of the Twelfth International Conference on Intelligent Tutoring Systems* (pp. 551–560). Switzerland: Springer International.

Matsuda, N., Cohen, W. W., & Koedinger, K. R. (2014). Teaching the teacher: Tutoring SimStudent leads to more effective cognitive tutor authoring. *International Journal of Artificial Intelligence in Education*, *25*, 1–34.

Matsuda, N., Lee, A., Cohen, W. W., & Koedinger, K. R. (2009). A computational model of how learner errors arise from weak prior knowledge. *Proceedings of the Thirty-First Annual Meeting of the Cognitive Science Society* (pp. 1288–1293). Austin, TX: Cognitive Science Society.

Matsuda, N., Yarzebinski, E., Keiser, V., Cohen, W. W., & Koedinger, K. R. (2011). Learning by teaching SimStudent – An initial classroom baseline study comparing with cognitive tutor. *Proceedings of the Fifteenth International Conference on Artificial Intelligence in Education* (pp. 213–221). Berlin: Springer-Verlag.

Mcdermott, J. (1980). R1: An expert in the computer systems domain. *Proceedings of the First National Conference on Artificial Intelligence* (pp. 269–271). Menlo Park, CA: AAAI Press.

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

Myers, B., Hudson, S. E., & Pausch, R. (2000). Past, present, and future of user interface software tools. *ACM Transactions on Computer-Human Interaction*, *7*, 3–28.

Myers, B., Pane, J., & Ko, A. (2004). Natural programming languages and environments. *Communications of the ACM*, *47*, 47–52.

Norman, D. A. (2010). Natural user interfaces are not natural. *Interactions*, *17*, 6–10.

Olsen, J. K., Belenky, D. M., Aleven, V., & Rummel, N. (2014). Using an intelligent tutoring system to support collaborative as well as individual learning. *Proceedings of the Twelfth International Conference on Intelligent Tutoring Systems* (pp. 134–143). Switzerland: Springer International.

Polanyi, M. (1966). *The tacit dimension*. Chicago, IL: The University of Chicago Press.

Schön, D. A. (1983). *The reflective practitioner: How professionals think in action*. New York: Basic Books.

Settles, B. (2012). *Active learning*. San Rafael, CA: Morgan & Claypool.

Sheridan, T. B. (1992). *Telerobotics, automation, and human supervisory control*. Cambridge, MA: MIT Press.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., … Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*, 484–489.

Simon, H. A. (1983). Why should machines learn? In R. S. Michalski (Ed.), *Machine learning* (pp. 25–37). Berlin: Springer.

Taylor, G., Quist, M., Lanting, M., Dunham, C., Theisen, P., & Muench, P. (2017). Multi-modal interaction for robotic mules. *Proceedings of SPIE 10195, Unmanned Systems Technology XIX*. Anaheim, CA: International Society for Optics and Photonics.

Thomaz, A., Hoffman, G., & Breazeal, C. (2006). Reinforcement learning with human teachers: Understanding how people want to teach robots. *Proceedings of the Fifteenth IEEE International Symposium on Robot and Human Interactive Communication* (pp. 352–357). New York: IEEE Press.

Traum, D. R., & Hinkelman, E. A. (1992). Conversation acts in task-oriented spoken dialogue. *Computational Intelligence*, *8*, 575–599.

Vanlehn, K. (2006). The behavior of tutoring systems. *International Journal of Artificial Intelligence in Education*, *16*, 227–265.

Weiser, M., & Brown, J. S. (1996). Designing calm technology. *PowerGrid Journal*, *1*, 75–85.

Wigdor, D., & Wixon, D. (2011). *Brave NUI world: Designing natural user interfaces for touch and gesture*. Burlington, NJ: Morgan Kaufmann.