Investigating Machine-Learning Interaction with Wizard-of-Oz Experiments

Rob Sheline
Soar Technology, Inc.
Ann Arbor, MI 48105
rob.sheline@soartech.com

Christopher J. MacLellan
Soar Technology, Inc.
Ann Arbor, MI 48105
chris.maclellan@soartech.com

Abstract

Machine-learning research has historically focused on improving learning algorithms but often neglects the context within which a learning technology is situated, and only minimally evaluates how natural and efficient these systems are for users to interact with. To better investigate these aspects of machine-learning, we introduce a novel experimental methodology, which we refer to as *dual-sided*, *restricted-perception Wizard-of-Oz* experiments, and describe a web-based framework for executing them. In classic *Wizard-of-Oz* experiments, a researcher rapidly tests a hypothetical Artificial Intelligence system with real users by manually simulating its capabilities. We extend this approach to support the prototyping and evaluation of interactive learning systems by instead simulating the learning technology using a naïve experimental participant, who learns the task by receiving the same inputs and outputs as the hypothetical learning system. We present proof-of-concept efforts to use this paradigm to evaluate hypothetical learning systems in the domain of service robotics.

1 Introduction

Machine learning has come a long way in a short time. Research has shown that learning-enabled systems can achieve or exceed human-level performance on a wide variety of complex tasks (He et al., 2015; Mnih et al., 2015), *given sufficient engineering effort*. As the field matures, more attention is being given to machine-learning cost and developmental complexity, rather than pure performance. For instance, Simard et al. (2017) makes a call for differentiating machine learning from *machine teaching*. While the former focuses on developing new algorithms to improve the accuracy of the "learner," the latter centers on increasing the efficacy of the "teachers." The hope is that, as the field matures, we will see increased productivity and turnaround—as we have seen in recent decades within the software development community. Nonetheless, for now at least, designing and tuning machine-learning systems remains expensive.

This shift in focus highlights the need for further study of how users interact with machine-learning systems. However, this creates a chicken-and-egg dilemma: we wish to study the space of machine-learning system interaction designs, but evaluating different interactions requires machine-learning components to exist in the first place. While this is not a problem for evaluating existing machine-learning systems, we are particularly interested in exploring hypothetical machine-learning systems for which necessary capabilities do not yet exist or for which building theses capabilities is practically infeasible.

Our solution to this dilemma is inspired by *Wizard-of-Oz* (WoZ) studies (Dahlbäck et al., 1993), a classic approach to design within the Human-Computer Interaction (HCI) community. WoZ studies circumvent the up-front cost of fully engineering an intelligent system or agent (Maulsby et al., 1993) by substituting a human "behind the curtain" who simulates the desired behavior. This allows for rapid,

Learning by Instruction Workshop (NIPS 2018), Montréal, Canada.

iterative design at lower cost. Recent work has explored the use of WoZ studies to investigate social interaction strategies that hypothetical robots might use when interacting with humans (Sequeira et al., 2016). This work leverages a novel *restricted-perception* paradigm, that constrains the information available to the "wizard" to precisely what would be available to the hypothetical social robot. This constraint reduces experimental confounds which may arise due to perceptions uniquely available to the "wizard", and enables researchers to test whether the information being provided to "wizard" is sufficient for them to execute a particular social interaction strategy.

While these methodologies have been successfully applied to investigate AI and robotic system interactions, they have not been employed to investigate machine-learning system interactions. To extend these approaches to support the rapid prototyping of learning systems, we propose the use of *dual-sided*, restricted-perception, Wizard-of-Oz experiments. Rather than having a knowledgeable researcher/experimenter as a "wizard" on the back end, this new approach employs a naïve participant that must learn from the same inputs as the hypothetical learning system. Like existing restricted-perception methods, our variant constrains the interactions allowed between the "wizard" and the external world to those available to the hypothetical learning system. Although this new approach is preliminary, we believe it should enable researchers to cost effectively evaluate the interactions of the hypothetical system as well as preliminary test whether the information provided to system is sufficient for learning, under the assumption that if the humans cannot learn from the inputs then a machine is unlikely to be able to.[1].

To test the feasibility of our approach, we conducted a small pilot to evaluate three hypothetical learning systems in the domain of service robotics. Each hypothetical system used a different machine-learning interaction pattern from the Natural Training Interactions framework (MacLellan et al., 2018). The patterns used were: direct instruction, operant conditioning, and apprentice learning. See Appendix B for a more detailed explanation of how the patterns function.

Our results, while preliminary, demonstrate that this approach can be successfully applied to the *rapid* evaluation of hypothetical learning system interaction designs and that this approach can yield *useful* quantitative measures for assessing potential designs.

2 Prior Work

2.1 Machine Teaching

At present, intelligent applications (ML and AI systems) typically require painstaking and skilled efforts to design and implement. There is a clear demand for developers of intelligent systems, not met by the available supply of expert labor. Simard et al. (2017) recommends disambiguating machine-learning from *machine-teaching*, so that developers can receive specialized training in how best to both develop and teach learning systems.

In traditional machine-learning, the focus of effort and research is primarily on developing better algorithms. In contrast, machine-teaching is concerned with improving how developers interact with learning systems, including amount and sophistication of human intervention required. Professional software development has matured over the past 50 years. Abstractions such as high-level languages, design patterns, documentation, software architecture, unit-testing, and version control have created a world where software can be produced in a consistent and reliable way. Software artifacts can be made reusable, portable, and maintainable, and need not be married to the engineers who implemented them. Contrast this with the state of modern machine-learning, where models can be frighteningly opaque and robust interoperability standards are often non-existent.

This distinction between machine learning and teaching is recent and, thus, machine teaching is a nascent field. Producing machine-learning enabled systems involves developing models (in the traditional sense) integrated with more typical software objects. However, since models are often coupled tightly with these environments. It can be difficult to apply the kind of iterative development pattern (e.g., agile, waterfall) common in software engineering to teaching learning systems: the model is an over-constrained black box, creating a developmental bottleneck. We seek to alleviate this issue by providing a method with which to learn about how best to interact with learning systems *without* depending on a previously-existing, well-behaved model.

[1]We recognize that this assumption is dubious, but believe it serves as a reasonable diagnostic

Knowledge	Patterns	Types	Modalities
• Goals • Beliefs • Concepts • Experience • 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

Table 1: The Natural Training Interaction Framework.

2.2 Natural Training Interactions Framework

Much of the existing research on knowledge transfer centers on the development of new machine-learning capabilities and only minimally evaluates how natural and efficient learning systems are for users to interact with. MacLellan et al. (2018) draw attention to the question of what makes interaction with a learning system natural for human instructors (i.e., the users) and then work backwards to develop the necessary computational capabilities for a system to learn from these interactions. This user-centered approach has yet to be explored within the machine-learning community, but has been widely successful within the HCI community. In particular, it has been found to be a consistently reliable approach for developing usable technologies.

To support this process MacLellan et al. (2018) present the Natural Training Interactions framework, which describes the space of possible interaction patterns a learning system might employ (see Table 1) The framework assumes that the goal of teaching a system is to change some aspect of its **knowledge**. To update knowledge, a learning system and a trainer interact according to instructional **patterns**. Within patterns, they employ several **types** of interactions, and these interactions can be done through various **modalities**.

This framework describes multiple commonly used patterns within existing learning systems, such as operant conditioning (reinforcement learning), direct instruction (learning-by-demonstration), and apprentice learning (interactive task learning). It also provides a language for formulating scientific hypotheses about how learning systems should interact with users to best achieve naturalness.

Based on this framework, MacLellan et al. (2018) hypothesize that different combinations of patterns, types, and modalities of interaction are better suited for updating different kinds of knowledge, mediated by the trainer's preferences and potentially other contextual factors. From an engineering prospective, this hypothesis provides a means of decomposing the problem of efficiently transferring knowledge to a learning system into multiple sub-problems that can be tackled with individual research efforts. It seems most obvious to break the problem down by the type of knowledge being transferred (e.g., focusing on skill knowledge). Equipped with this hypothesis, it should be possible to explore approaches for each knowledge type independently and then combine the different approaches together into an overall architecture that supports all knowledge types.

3 Natural Training Interaction Testbed

In order to investigate different learning system interaction patterns, we developed the dual-sided, restricted-perception WoZ experimental methodology, mentioned previously. We extended restricted-perception WoZ experiments to study the implications that choice of learning patterns has on naturalness of user-experience as well as performance in computer-embeddable tasks. In conception, it is similar to prior work on using WoZ experiments to study social interaction (Sequeira et al., 2016), but our interest is in applying the approach to investigate interaction with machine-learning

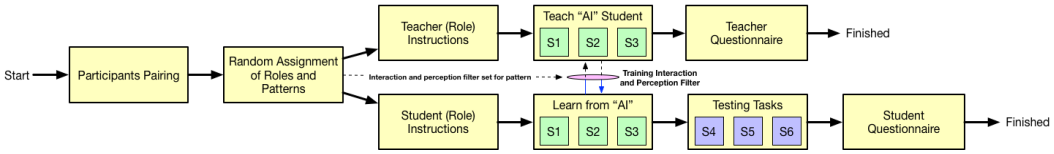

Figure 1: The Dual-Sided, Restricted-Perception, Wizard-of-Oz Experimental Design

systems. Figure 1 shows the general experimental design for our approach, where teacher and student participants interact *in parallel* through a perception filter.

Our method relies on the assumption that a human subject is a useful proxy for an arbitrary learning system. We understand that this assumption breaks down in the limit, but believe that for a wide variety of currently desirable and feasible AI tasks, it is a reasonable assumption to make. To that end, test subjects (students) should be agnostic of the task, as much as is possible. Of course humans bring prior knowledge to the table: at the very least, "common sense" notions such as causality, how to navigate ≤ 3 dimensional spaces, etc. Though a potential complication, similar generic capabilities could be in principal supplied to a learning system, and is in any event unavoidable.

To test the feasibility of this methodology, we engineered the *Natural Training Interactions Testbed* for conducting these experiments (see Figure 2). We created a training user interface (UI) which embeds the task (an HTML/JavaScript page where the task is nested inside a single). The training UI handles training interactions, and essentially is the window/filter through which subjects view the experiment, and accordingly have their perception restricted. The interface supports different modalities: currently command line, text, and GUI in the form of clickable buttons, but in principal any found in table 1. The training UI communicates with the testbed (experiment server, see section 3) in order to administer the experiment.

The testbed allows novel tasks to be studied in a WoZ-experiment without having to start from scratch, given an implementation of the *testbed interface*. The testbed uses a variety of modern software technologies (opensource where possible). See Appendix A for a more thorough examination of these components.

In designing the testbed we specified a conceptual API, here implemented in Python as endpoints of a flask web-server. The API is implemented on by the client side, with the caveat that a client can be either a teacher or student. Upon receiving a request to participate, the server will send clients an *initial state*, or alternatively tell them to generate one. The clients have access to a UI with two sub-modules, the *training-UI* and the *task-UI*. In the course of administering a particular learning pattern, the server can lock and unlock either of these UIs (canonically, either the teacher *or* the student will have an unlocked UI at one time, but this need not be the case). When an action is taken in the task-UI, the server forwards it to the other participant. When an action is taken in the training-UI (such as "disallow previous action" or "request assistance") the server will update the task-state appropriately (rewind the game state, let the teacher take a move, respectively).

Our goal is the ability to prototype Wizard-of-Oz experiments using the same interface the machine-learner uses, or is envisioned to use. Additionally, the architectural components (task, interaction pattern, learner) and software components (server, client, data store) are intended to be as modular as possible so learned-lessons can be incorporated into future work, without having to recreate each component from scratch.

4 Pilot Study

4.1 Method

A simple service-robotics task was implemented in Unity/WebGL (see bottom portion of figure 2). Players maneuver around a randomly generated maze-like set of corridors and rooms, and encounter objects which they can pick up and put down. In addition to maintaining the game state/physics, the Unity object implements the Task API. In this way, aside from implementing this interface, the Unity object is agnostic to the existence of the experiment, training interaction, student and teacher. The testbed is (as much as possible) likewise agnostic about the state of the unity object.

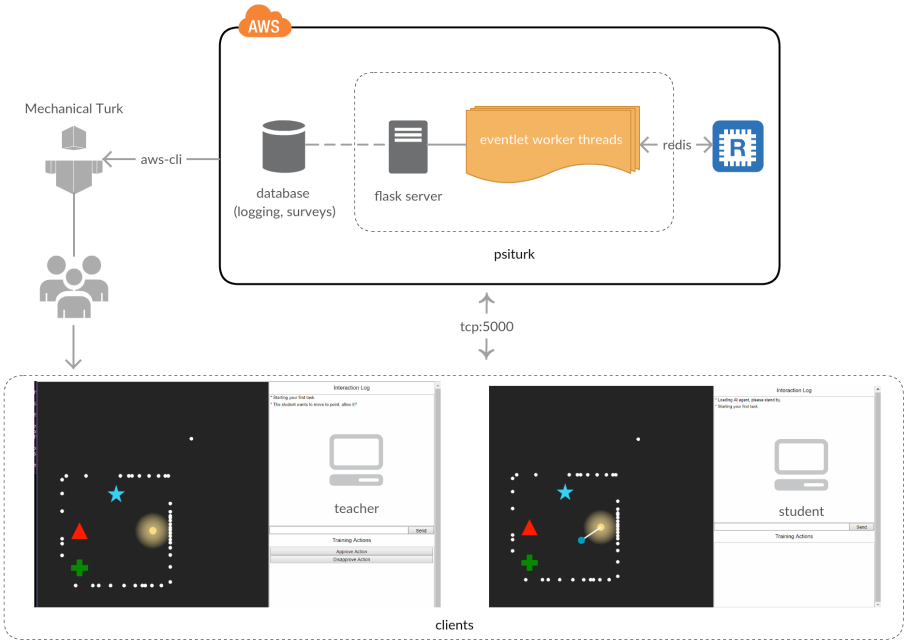

Figure 2: Architecture for Natural Training Interaction Testbed

Using the Natural Training Interaction testbed configured against the robot navigation task, we performed dual-sided, restricted-perception, WoZ experiments using participants sourced through Amazon Mechanical Turk. Upon viewing and accepting the experiments advertising on the MTurk worker dashboard, users were redirected to an amazon EC2 instance under our control. Subjects were first shown a consent and release form, and then instructions. The instructions explained how to use the training interface, and in the case of the teacher, specifics detailing the task. Teachers and students were told that they would receive a bonus corresponding to how accurately, quickly, and efficiently the student performed the task (so that both were motivated to actually learn the task).

While subjects read the instructions, teacher-student pairs were created and assigned one of three interaction patterns: Operant Conditioning, Direct Instruction, Apprentice Learning, see Table 1 and prior work for more details MacLellan et al. (2018). Additionally, five game scenarios from a procedurally-generated and verified pool were assigned to each pair. Of these five scenarios, three were used as a training set for teachers and students; the remaining two were used as a test set for the student.

4.2 Results

We ran a small-scale pilot experiment to assess the feasibility of our methodology and experimental testbed. At the end of the pilot, we asked participants to provide seven ratings on a scale of 1-5. The ratings were designed to ascertain how natural and effective they felt their session was. Appendix C lists the specific evaluation questions we used, which were based on a previous set of metrics for evaluating interactive learning systems (Laird et al., 2017). Figure 3 shows a summary of pilot participants answers on this survey . Although these results suggest some differences between the different interaction patterns we tested, we do not intent any conclusions to be drawn from these results. Instead, we provide them as a demonstration of the kinds of evaluations that can be performed on hypothetical learning systems using our methodology (e.g., naturalness and effectiveness). Additionally, they provide a demonstration of the kinds of comparisons that might be made between different kinds of interactions patterns, such as direct instruction being less easy for teachers (Q7) and apprentice learning being generally better for teachers (along all dimensions).

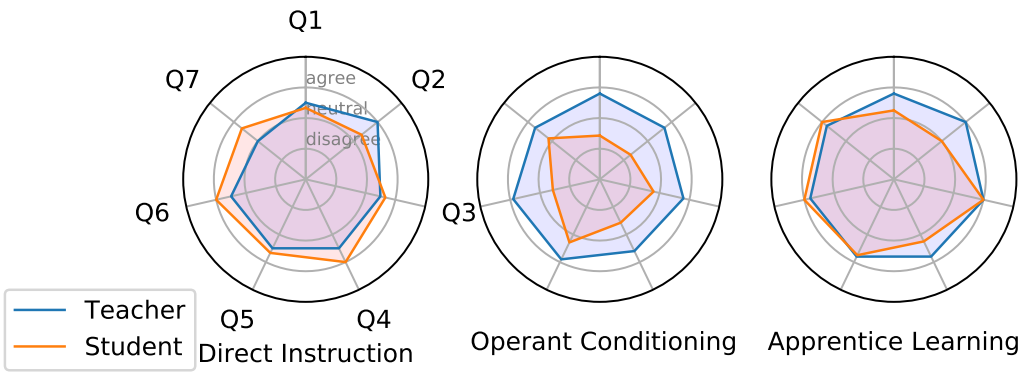

Figure 3: Survey results. Average Likert score for questions Q1-Q7

5 Limitations and Future Work

This work is preliminary, and while results were promising, we did encounter a number of practical issues that made executing these experiments challenging. In particular, it was difficult to get users to cooperate with a partner for long enough to complete a session. During a pilot with researchers from our lab, we found that the human learners often struggled to learn the dynamics of our 2D service robotics task. Occasionally, the teacher would get frustrated with the student and give up. Our testbed does not not currently have the means to appropriately capturing these frustrations, which we should explore in future work. Additionally, we encountered numerous difficulties with participants falling out of sync, in terms of task as well as participation. Future work should explore techniques for enabling a more seamless experience.

In conclusion, we presented a new experimental approach for evaluating machine-learning interaction designs: dual-sided, restricted-perception, WoZ experiments. To demonstrate the feasibility of this approach we developed the Natural Training Interactions testbed and used it to conduct a small pilot study. The results of our pilot show that it is possible to use this approach to rapidly evaluate hypothetical machine-learning systems.

6 Acknowledgements

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 authors and do not necessarily reflect the views of the OFFICE OF NAVAL RESEARCH.

References

Dahlbäck, N., Jönsson, A., and Ahrenberg, L. (1993). Wizard of oz studies—why and how. *Knowledge-based systems*, 6(4):258–266.

Hawkins, R. X. (2015). Conducting real-time multiplayer experiments on the web. *Behavior Research Methods*, 47(4):966–976.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

Laird, J. E., Gluck, K., Anderson, J., Forbus, K. D., Jenkins, O. C., Lebiere, C., Salvucci, D., Scheutz, M., Thomaz, A., Trafton, G., Wray, R. E., Mohan, S., and Kirk, J. R. (2017). Interactive task learning. *IEEE Intelligent Systems*, 32(4):6–21.

MacLellan, C., Harpstead, E., Mariner III, R. P., and Koedinger, K. R. (2018). A Framework for Natural Cognitive System Training Interactions. *Advances in Cognitive Systems*, 6:1–16.

Maulsby, D., Greenberg, S., and Mander, R. (1993). Prototyping an intelligent agent through wizard of oz. In *Proceedings of the INTERACT'93 and CHI'93 conference on Human factors in computing systems*, pages 277–284. ACM.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529.

Sequeira, P., Alves-Oliveira, P., Ribeiro, T., Di Tullio, E., Petisca, S., Melo, F. S., Castellano, G., and Paiva, A. (2016). Discovering social interaction strategies for robots from restricted-perception wizard-of-oz studies. In *The Eleventh ACM/IEEE International Conference on Human Robot Interaction*, pages 197–204. IEEE Press.

Simard, P. Y., Amershi, S., Chickering, D. M., Pelton, A. E., Ghorashi, S., Meek, C., Ramos, G., Suh, J., Verwey, J., Wang, M., et al. (2017). Machine teaching: A new paradigm for building machine learning systems. *arXiv preprint arXiv:1707.06742*.

Appendix A. Natural Training Interaction Testbed components

Amazon Mechanical Turk (MTurk) is an online marketplace for human intelligence tasks. Accessible via programmatic API, workers can be requisitioned at any time of the day, and from a wide variety of sub-populations according to experimental need. Most existing web-based experiments do not involve participants interaction in a persistent sense, due to technical limitations of HTTP request methods. (Hawkins (2015)). A variety of modern web programming tools allow us to go beyond these technical limitations.

Psiturk is an open-source python framework for conducting experiments with MTurk. It streamlines some of the boilerplate obstacles most experiments will likely face. Small modifications needed to be made to to the framework to support bi-directional socket communication required by the experiment.

Flask Under the hood, PsiTurk hosts the experiment via Flask, a python-based web-server. The testbed is implemented as a flask blueprint, which PsiTurk registers within its own flask server. Thus, the server implementation is not coupled to PsiTurk, and could function independently. In effect, PsiTurk wraps the experiment server and acts as a broker between it and MTurk, in addition to providing the aforementioned experiment boilerplate.

Socket.io "enables real-time, bidirectional and event-based communication between the browser and the server". Unlike HTTP and like websockets, socket.io is bi-directional, so a message can be initiated by either party. Thus, socket.io is the communication medium of the testbed; any computer-based task to be studied and ultimately learned will need to be able to implement the testbed API.

Redis is an-memory data store, which the testbed uses as a cache and message broker between worker threads. Flask uses eventlet worker threads under the hood for concurrency. Eventlet brokers socket.io communications to threads using a FIFO queue. When a message is received, the thread retrieves the corresponding experiment state from redis, mutates it according to the interaction pattern being used, and stores it back in redis. Because of the current experimental design (teacher and student take turns interacting), there is little to no risk of the users or experimental state becoming desyncronized. Nonetheless, because of concerns surrounding this and potential non-deterministic outcomes in unity, custom logic was embedded in the server to detect and fix possible desynchrony. Future effort may be directed at this problem, though it is only likely to be an issue at very large scale (many concurrent connections) or with asynchronous interaction patterns (teacher and student can interrupt each other).

Appendix B. Learning patterns

The three learning patterns used in the pilot study (see table 1). In all cases, teachers and students were instructed how to use the training interface, but only teachers were instructed how to use the task interface.

In all cases, students were instructed to ascertain how the task interface functioned, as well as what the goal within the task was. Then at testing time, they were instructed to execute the goal. (The goal

might be, for example, "bring the green cross to the blue star": something readily understandable in english, but also able to be scored programmatically.)

In the case of Operant Conditioning and Apprentice Learning, students had the additional option within the training interface to "mark task complete". When they believed they had completed the task, they would ask the teacher to approve. In the case of direct instruction, teachers simply indicated the task was complete.

When the teacher decided that the task was complete (whether demonstrating or approving), the experiment proceeded to the next scenario.

- **Direct Instruction** Teachers were instructed to execute the task. Students were instructed to passively observe.
- **Operant Conditioning** Students were allowed to act. After each action within the task interface, teachers were given the option approve/deny. If they denied, the game-state was rewound and the student informed that their action was incorrect.
- **Apprentice Learning** An extension of operant conditioning. Students took actions, which were approved/denied by the teacher. Additionally, students could "request demonstration" from the teacher, who would take one action.

Appendix C. Exit Surveys

After participating in the experiment, subjects were instructed to respond to the following statements on a scale of 1-5 (Strongly Disagree, Disagree, Neither Agree nor Disagree, Agree, Strongly Agree):

Student Likert Statements

1. I learned to correctly perform the task by the end of training.
2. I only needed a few examples to learn.
3. I was able to quickly decide what actions to take next.
4. Learning from the AI teacher was natural and intuitive.
5. The instructional feedback provided by the AI teacher was always useful.
6. I always received the instruction from the AI teacher that I wanted.
7. Learning from the AI teacher was easy.

Teacher Likert Statements

1. The AI student learned to correctly perform the task by the end of training.
2. The AI student only needed a few examples to learn.
3. The AI student quickly decided what actions to take next.
4. Instructing to the AI student was natural and intuitive.
5. The instructional options I was presented with were always useful.
6. I was always provided with all the instructional options I wanted.
7. Instructing the AI student was easy.