Data Efficient Paradigms for Personalized Assessment of Taskable AI Systems – Dissertation Abstract

Pulkit Verma

Thesis Advisor: **Siddharth Srivastava**

Autonomous Agents and Intelligent Robots Lab,

School of Computing and Augmented Intelligence, Arizona State University, USA

verma.pulkit@asu.edu

Abstract

The vast diversity of internal designs of taskable black-box AI systems and their nuanced zones of safe functionality make it difficult for a layperson to use them without unintended side effects. The focus of my dissertation is to develop algorithms and requirements of interpretability that would enable a user to assess and understand the limits of an AI system's safe operability. We develop a personalized AI assessment module that lets an AI system execute instruction sequences in simulators and answer the queries about its execution of sequences of actions. Our results show that such a primitive query-response capability is sufficient to efficiently derive a user-interpretable model of the system's capabilities in fully observable, and deterministic settings.

1 Introduction

The growing deployment of AI systems presents a pervasive problem of ensuring the safety and reliability of these systems. The problem is exacerbated because most of these AI systems are neither designed by their users nor are their users skilled enough to understand their internal working, i.e., the AI system is a black-box for them. Hence such systems may be used by non-experts who may not understand how they work or what they can and cannot do. Ongoing research on the topic focuses on the significant problem of answering such a user's questions about the system's behavior (Chakraborti et al. 2017a; Dhurandhar et al. 2018; Anjomshoae et al. 2019). However, most non-experts hesitate to ask questions about new AI tools (Mou and Xu 2017) and often do not know which questions to ask for assessing the safe limits and capabilities of an AI system. This problem is aggravated in situations where an AI system can carry out planning or sequential decision making. Lack of understanding about the limits of an imperfect system can result in unproductive usage or, in the worst-case, serious accidents (Randazzo 2018). This, in turn, limits the adoption and productivity of the AI systems.

My dissertation work aims to create general algorithms and methods for interpretability which when used with a black-box AI system, can help generate a description of its capabilities by interrogating it. Consider a situation where a logistics company buys new delivery robots. The person managing these robots is unsure whether the robots correctly understand a task, or if they can even execute it safely. If the manager was dealing with a delivery person, it might ask them questions such as "do you think it would be alright to bring refrigerated items in a regular bag?" If the answer is "yes", it might be a cause for concern. Answers to such questions can help the manager develop an understanding of the robot's frame of knowledge, or "model" while placing a minimal introspective requirement on the robot.

Figure 1: The personalized AI assessment module uses the user's preferred vocabulary, queries the AI system, and delivers an interpretable model of the AI system's capabilities.

I will next explain the focus of my dissertation (Sec. 2), followed by a short discussion on related work (Sec. 3), and will finally discuss some preliminary results (Sec. 4).

2 Focus of My Dissertation

In my dissertation, I plan to develop a *personalized AI-assessment module* (AAM), shown in Fig. 1, which can derive the model of capabilities of a black-box AI system in terms of an user-interpretable vocabulary. AAM takes as input using as input (i) the agent (ii) a compatible simulator using which the agent can simulate its primitive action sequences; and (iii) the user's concept vocabulary, which may be insufficient to express the simulator's state representation. Such assumptions on the agent are common. In fact, use of third-party simulators for development and testing is the bedrock of most of the research on taskable AI systems today (including game playing AI, autonomous cars, and factory robots). Providing simulator access for assessment is reasonable as it would allow AI developers to retain freedom and proprietary controls on internal software while supporting calls for assessment and regulation using approaches like ours. AAM then queries the AI system and receives its responses. At the end of the querying process, AAM returns a user-interpretable model of the AI system's capabilities. This approach's advantage is that the AI system need not know the user vocabulary or the modeling language.

Most simulator-based and analytical-model-based AI systems can easily answer the kind of questions discussed earlier. However, identifying the high-level capabilites of the AI system and generating the right set of questions to ask the AI system to efficiently learn a model of system's capabilities is a challenging problem. The focus of this new direction of research is on solving this problem. In context of this work, "actions" refer to the core *functionality* of the agent, denoting the agent's decision choices, or primitive actions that the agent could execute (e.g., a keystrokes in a video game). In contrast, "capabilities" refer to the *high-level behaviors* that the AI system can perform using its AI algorithms for behavior synthesis, including planning and learning (e.g., navigating to a room, opening a door, etc.). Thus, actions refer to the set of choices that a tabular-rasa agent may possess, while capabilities are a result of its agent function (Russell 1997) and can change as a result of algorithmic updates even as the agent uses the same actions.

Additionally, this proposed method, when used with any AI system, would also help make them compliant with Level II assistive AI – systems that make it easy for users to learn how to use them safely (Srivastava 2021).

2.1 Generating Interrogation Policies

I aim to create an interrogation policy that will generate the queries for the AI system, and use the AI system's answers to estimate its model in the user-interpretable vocabulary. I plan to generate these queries by reducing the query generation to a planning problem and then use an interrogation algorithm to iteratively generate new queries actively, based on responses to previous queries.

2.2 Inferring the Action Model

Given the predicates and actions, there is an exponential number of PDDL (McDermott et al. 1998) models possible. To avoid this combinatorial explosion, I plan to use a top-down process that eliminates large classes of models, inconsistent with the AI system, by computing queries that discriminate between pairs of *abstract models*. When an abstract model's answer to a query differs from that of the AI system, we can eliminate the entire set of possible models that are refinements of this abstract model.

I plan to start research on this front with simplistic queries in deterministic fully observable environments and expand the scope to more general settings. I plan to first extend this to settings where the model of an AI system adapts itself to work with the user in a better way, or due to some other reason. This will avoid relearning the complete model from scratch, and will learn the AI system's model much faster. In the future, this mechanism can be extended to more general forms of queries. Similar to active learning, information theoretic metrics can also be utilized to ascertain which queries will be better at any given time in the querying process.

2.3 Discovering the Capabilities and Learning their Descriptions

As mentioned earlier, I want the assessment module to discover the high-level capabilities of the AI system that can plan (using search or a policy), and not just the action model of an AI system. I plan to collect a set of state observations capturing the behavior of the AI system in form of the state transitions. I would then discover the high-level capabilities of the AI system's behavior using those state transitions, and then learn the description of these capabilities similar to the learning of action model discussed earlier. I plan to extend this to settings where either the capabilities are stochastic even though the low level transition system is deterministic, or the low level transition itself is stochastic, thereby resulting in capabilities that are stochastic.

3 Related Work

Learning action models Several action model learning approaches (Gil 1994; Yang, Wu, and Jiang 2007; Cresswell, McCluskey, and West 2009; Zhuo and Kambhampati 2013; Aineto, Celorrio, and Onaindia 2019) have focused on learning the AI system's model using passively observed data. Jiménez et al. (2012) and Arora et al. (2018) present a comprehensive review of such approaches. These approaches do not feature any interventions, hence are susceptible to learning buggy models. Unlike these approaches, our approach queries the AI system and is guaranteed to converge to the true model while presenting a running estimate of the accuracy of the derived model; hence, it can be used in settings where the AI system's model changes due to learning or a software update.

Differential assessment Bryce, Benton, and Boldt (2016) address the problem of learning the updated mental model of a user using particle filtering given prior knowledge about the user's mental model. However, they make a strong assumption that the user knows enough to point out errors in the learned model if needed. Model reconciliation literature (Chakraborti et al. 2017b; Sreedharan et al. 2019; Sreedharan, Chakraborti, and Kambhampati 2021) deals with inferring the differences between the user and the agent models and removing them using explanations. These methods consider white-box known models whereas our approach works with black-box AI systems.

Learning high-level models Given a set of options encoding skills as input, Konidaris, Kaelbling, and Lozano-Perez (2018) and James, Rosman, and Konidaris (2020) propose methods for learning high-level propositional models of options representing various "skills." They assume access to predefined options and learn the high-level symbols that describe those options at the high-level. While they use options or skills as inputs to learn models defining when those skills will be useful in terms of auto-generated symbols (for which explanatory semantics could be derived in a post-hoc fashion), our approach uses user-provided interpretable concepts as a priori inputs to learn AI system capabilities: high-level actions as well as their interpretable descriptions in terms of the input vocabulary.

4 Preliminary Results

We developed three preliminary versions of the personalized AI assessment module, each focusing on one specific sub-

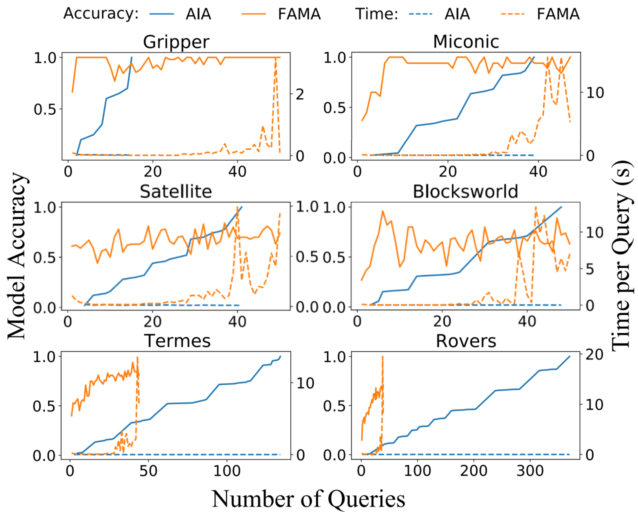

Figure 2: Performance comparison of AIA and FAMA in terms of model accuracy and time taken per query.

problem of the overall larger goal.

Learning the Action Model The first preliminary version of the AI assessment module, called the agent interrogation algorithm (AIA) (Verma, Marpally, and Srivastava 2021), efficiently derives a user-interpretable model of the system in stationary, fully observable, and deterministic settings. In the context of this initial work, user-interpretable means STRIPS-like (Fikes and Nilsson 1971) models because such models can be easily translated into interpretable descriptions, and they also allow interventions and assessment of causality. In the future, I plan to learn more general and more expressive models of the AI system.

Also, in this version, we used *plan outcome queries* which are parameterized by an initial state and a plan; and ask the AI system, the length of the longest prefix of the plan that it can execute successfully when starting in the given initial state, as well as the final state that this execution leads to. E.g., "Given that the truck $t1$ and package $p1$ are at location $l1$, what would happen if you executed the plan $\langle load\_truck(p1, t1, l1),\ drive(t1, l1, l2),\ unload\_truck(p1, t1, l2)\rangle$?".

We compared AIA with the closest related work FAMA (Aineto, Celorrio, and Onaindia 2019) in terms of; the accuracy of the learned model, the number of queries asked, and the time taken to generate those queries. Fig. 2 summarizes our findings for systems initialized with IPC domains. AIA takes lesser time per query and shows better convergence to the correct model. FAMA sometimes reaches nearly accurate models faster, but its accuracy continues to oscillate, making it difficult to ascertain when the learning process should be stopped. This is because the solution to FAMA's internal planning problem introduces spurious palm tuples in its model if the input traces do not capture the complete domain dynamics. Also, in domains with negative preconditions like Termes, FAMA was unable to learn the correct model.

We also showed that AIA can be used with simulator-based systems that do not know about predicates and report states as images. To test this, we wrote classifiers to detect predicates from images of simulator-states in the PDDLGym (Silver and Chitnis 2020) framework. This framework provides ground-truth PDDL models, thereby simplifying the estimation of accuracy. We initialized the AI system with one of the two PDDLGym environments, Sokoban and Doors. AIA inferred the correct model in both cases, and the average number of queries (over 5 runs) used to predict the correct model for Sokoban and Doors were 201 and 252, respectively.

Finally, we also show that the models that we learn capture the correct causal relationships in the AI system's behavior in terms of how the system operates and interacts with its environment (Verma and Srivastava 2021), unlike the models learned by approaches that only use observational data. We call such causal model a *generalized dynamical causal model* of the AI system capturing under what conditions it executes certain actions and what happens after it executes them.

Differential Assessment We developed a *differential assessment* version of the personalized AI assessment module, called DAAISy (Nayyar, Verma, and Srivastava 2022). This addresses the problem of accurately predicting the behavior of a black-box AI system that is evolving and adapting to changes in the environment it is operating in.

The algorithm for differential assessment utilizes an initially known PDDL model of the AI system in the past, and a small set of observations of AI system's execution. It uses these observations to develop an incremental querying strategy that avoids the full cost of assessment from scratch and outputs a revised model of the system's new functionality.

We refer to a predicate in an action's precondition or effect as a *pal-tuple*, and it can have three modes; positive, negative, or absent, depending on whether that predicate is present in the action's precondition (or effect) in as a positive literal, a negative literal or is absent. To assess the performance of our approach with increasing drift, we employed two methods of generating the initial domains: (a) dropping the *pal-tuples* already present, and (b) adding new *pal-tuples*. For each experiment, we used both types of domain generation. We generated different initial models by randomly changing modes of random *pal-tuples* in the IPC domains. Thus, in all our experiments an IPC domain plays the role of ground truth model and a randomized model is used as the initial known model.

We evaluated the performance of DAAISy along two directions; the number of queries it takes to learn the updated model of the AI system with increasing amount of drift, and the correctness of the model DAAISy learns as compared to the AI system's updated model.

As shown in the plots in Fig. 3, the computational cost of assessing each AI system, measured in terms of the number of queries used by DAAISy, increases as the amount of drift in the AI system's model increases. This is expected as the amount of drift is directly proportional to the number of *pal-tuples* affected in the domain. This increases the number

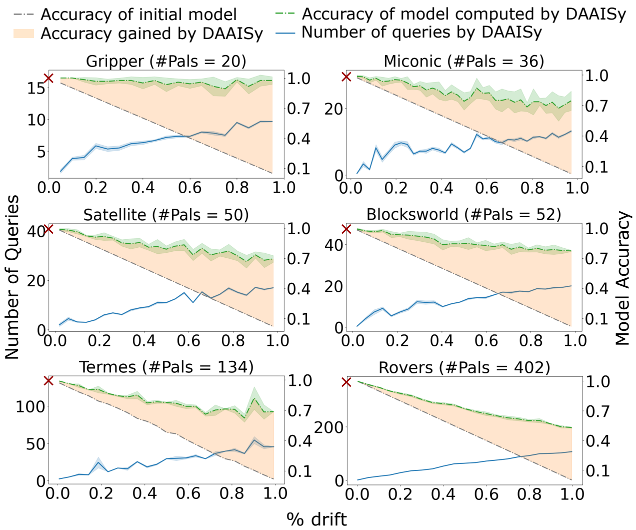

Figure 3: The number of queries used by DAAISy and AIA (marked × on y-axis), as well as accuracy of model computed by DAAISy with increasing amount of drift. Amount of drift equals the ratio of drifted *pal-tuples* and the total number of *pal-tuples* in the domains (#Pals).

of *pal-tuples* that DAAISy identifies as affected, and hence ends up asking more questions.

Also, DAAISy always took fewer queries as compared to AIA to reach reasonably high levels of accuracy because AIA does not use information about the initial known model of the AI system and thus ends up querying for all possible *pal-tuples*. DAAISy, on the other hand, predicts the set of *pal-tuples* that might have changed based on the observations collected from the AI system and thus requires significantly fewer queries.

Discovering the capabilities and learning their descriptions We also developed a version of AAM that can discover high-level capabilities of an AI planning agent expressible in terms of the user-interpretable concept vocabularies (Verma, Marpally, and Srivastava 2022). The descriptions of these capabilities as a model are returned to the user as opposed to the model of agent's primitive actions.

We initialized the agents using the General Video Game Artificial Intelligence framework (Perez-Liebana et al. 2016). For each agent, we create a random game instance with the goal of achieving one of the user's specified properties of interest (implemented as predicates). We use the solution to that instance to generate an execution trace that is used to discover the capabilities of the agent. We then ask the agent a sequence of queries and use the responses to complete the descriptions of these capabilities in a STRIPS-like form. Note that these queries are generated in high-level user vocabulary that the agent does not understand, hence we split each query into multiple sub-queries in a form that agent can answer. The multiple agent responses are also converted to the high-level responses used to complete the capability descriptions. The approach is guaranteed to compute capability descriptions that are correct in the sense that they are consistent with the execution traces, and refinable and executable with respect to the true capabilities of the agent.

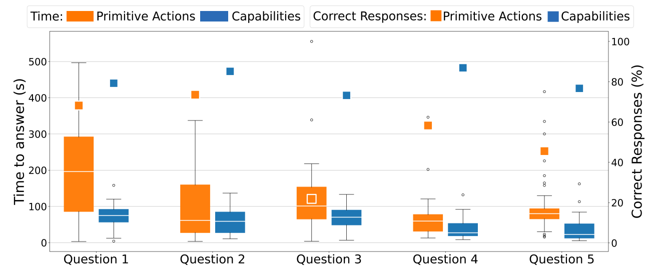

Figure 4: Data from behavior analysis shows that using computed capability descriptions took lesser time and yielded more accurate results.

We also conducted a user study to evaluate interpretablity of the capability descriptions computed by our approach. Intuitively, our notion of interpretability matches that of common English and its use in AI literature, e.g., as enunciated by Doshi-Velez and Kim (2018): *"the ability to explain or to present in understandable terms to a human."* We evaluate this through the following operational hypothesis:

H1. The discovered capabilities make it easier for users to analyze and predict outcome of agent's possible behaviors.

We designed a user study to evaluate H1. This study compares the predictability and analyzability of agent behavior in terms of the agent's low-level actions and high-level capabilities. Each user is explained the rules of an ATARI-like game. One group of users – called the primitive action group – are presented with text descriptions of the agent's primitive actions, while the users in the other group – called the capability group – are presented with a text description of the six capabilities discovered by our approach. The capability group users are asked to choose a short summarization for each capability description, out of the eight possible summarizations that we provide, whereas the primitive action group users are asked to choose a short summarization for each of the five primitive action description, out of the five possible summarizations that we provide. Then each user is given the same 5 questions in order. Each question contains two game state images; start and end state. The user is asked what sequence of actions or capabilities that the agent should execute to reach the end state from the start state. Each question has 5 possible options for the user to choose from, and these options differ depending on their group. We then collect the data about the accuracy of the answers, and the time taken to answer each question.

The results for the behavior analysis study are shown in (Fig. 4) The users took less time to answer questions and they got more responses correct when using the capabilities as compared to using primitive actions. This validates H1 that the discovered capabilities made it easier for the users to analyze and predict the agent's behavior correctly.

References

Aineto, D.; Celorrio, S. J.; and Onaindia, E. 2019. Learning Action Models With Minimal Observability. *Artif. Intell.*, 275: 104–137.

Anjomshoae, S.; Najjar, A.; Calvaresi, D.; and Främling, K. 2019. Explainable Agents and Robots: Results from a Systematic Literature Review. In *Proc. AAMAS*.

Arora, A.; Fiorino, H.; Pellier, D.; Métivier, M.; and Pesty, S. 2018. A Review of Learning Planning Action Models. *The Knowledge Engineering Review*, 33: E20.

Bryce, D.; Benton, J.; and Boldt, M. W. 2016. Maintaining Evolving Domain Models. In *Proc. IJCAI*.

Chakraborti, T.; Sreedharan, S.; Zhang, Y.; and Kambhampati, S. 2017a. Plan Explanations as Model Reconciliation: Moving Beyond Explanation as Soliloquy. In *Proc. IJCAI*.

Chakraborti, T.; Sreedharan, S.; Zhang, Y.; and Kambhampati, S. 2017b. Plan Explanations as Model Reconciliation: Moving Beyond Explanation as Soliloquy. In *Proc. IJCAI*.

Cresswell, S.; McCluskey, T.; and West, M. 2009. Acquisition of Object-Centred Domain Models from Planning Examples. In *Proc. ICAPS*.

Dhurandhar, A.; Chen, P.-Y.; Luss, R.; Tu, C.-C.; Ting, P.; Shanmugam, K.; and Das, P. 2018. Explanations based on the Missing: Towards Contrastive Explanations with Pertinent Negatives. In *Proc. NeurIPS*.

Doshi-Velez, F.; and Kim, B. 2018. *Considerations for Evaluation and Generalization in Interpretable Machine Learning*, 3–17. Springer International Publishing.

Fikes, R. E.; and Nilsson, N. J. 1971. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. *Artificial Intelligence*, 2(3-4): 189–208.

Gil, Y. 1994. Learning by Experimentation: Incremental Refinement of Incomplete Planning Domains. In *Proc. ICML*.

James, S.; Rosman, B.; and Konidaris, G. 2020. Learning Portable Representations for High-Level Planning. In *Proc. ICML*.

Jiménez, S.; De La Rosa, T.; Fernández, S.; Fernández, F.; and Borrajo, D. 2012. A Review of Machine Learning for Automated Planning. *The Knowledge Engineering Review*, 27(4): 433–467.

Konidaris, G.; Kaelbling, L. P.; and Lozano-Perez, T. 2018. From Skills to Symbols: Learning Symbolic Representations for Abstract High-Level Planning. *Journal of Artificial Intelligence Research*, 61(1): 215–289.

McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D. S.; and Wilkins, D. 1998. PDDL – The Planning Domain Definition Language. Technical Report CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control.

Mou, Y.; and Xu, K. 2017. The Media Inequality: Comparing the Initial Human-Human and Human-AI Social Interactions. *Computers in Human Behavior*, 72: 432–440.

Nayyar, R. K.; Verma, P.; and Srivastava, S. 2022. Differential Assessment of Black-Box AI Agents. In *Proc. AAAI*.

Perez-Liebana, D.; Samothrakis, S.; Togelius, J.; Schaul, T.; and Lucas, S. 2016. General Video Game AI: Competition, Challenges and Opportunities. In *Proc. AAAI*.

Randazzo, R. 2018. What went wrong with Uber's Volvo in fatal crash? Experts shocked by technology failure. *The Arizona Republic*.

Russell, S. J. 1997. Rationality and Intelligence. *Artificial Intelligence*, 94(1-2): 57–77.

Silver, T.; and Chitnis, R. 2020. PDDLGym: Gym Environments from PDDL Problems. In *ICAPS 2020 Workshop on Planning and Reinforcement Learning*.

Sreedharan, S.; Chakraborti, T.; and Kambhampati, S. 2021. Foundations of Explanations as Model Reconciliation. *Artificial Intelligence*, 103558.

Sreedharan, S.; Hernandez, A. O.; Mishra, A. P.; and Kambhampati, S. 2019. Model-Free Model Reconciliation. In *Proc. IJCAI*.

Srivastava, S. 2021. Unifying Principles and Metrics for Safe and Assistive AI. In *Proc. AAAI*.

Verma, P.; Marpally, S. R.; and Srivastava, S. 2021. Asking the Right Questions: Learning Interpretable Action Models Through Query Answering. In *Proc. AAAI*.

Verma, P.; Marpally, S. R.; and Srivastava, S. 2022. Discovering User-Interpretable Capabilities of Black-Box Planning Agents. In *Proc. KR*.

Verma, P.; and Srivastava, S. 2021. Learning Causal Models of Autonomous Agents using Interventions. In *IJCAI 2021 Workshop on Generalization in Planning*.

Yang, Q.; Wu, K.; and Jiang, Y. 2007. Learning Action Models from Plan Examples Using Weighted MAX-SAT. *Artificial Intelligence*, 171(2-3): 107–143.

Zhuo, H. H.; and Kambhampati, S. 2013. Action-Model Acquisition from Noisy Plan Traces. In *Proc. IJCAI*.