Fanoos: Multi-Resolution, Multi-Strength, Interactive Explanations for Learned Systems

David Bayani , Stefan Mitsch
School of Computer Science, Carnegie Mellon University
dcbayani@alumni.cmu.edu, smitsch@cs.cmu.edu

Abstract

Machine learning becomes increasingly important to control the behavior of safety and financially critical components in sophisticated environments, where the inability to understand learned components in general, and neural nets in particular, poses serious obstacles to their adoption. Explainability and interpretability methods for learned systems have gained considerable academic attention, but the focus of current approaches on only one aspect of explanation, at a fixed level of abstraction, and limited if any formal guarantees, prevents those explanations from being digestible by the relevant stakeholders (e.g., end users, certification authorities, engineers) with their diverse backgrounds and situation-specific needs. We introduce Fanoos, a flexible framework for combining formal verification techniques, heuristic search, and user interaction to explore explanations at the desired level of granularity and fidelity. We demonstrate the ability of Fanoos to produce and adjust the abstractness of explanations in response to user requests on a learned controller for an inverted double pendulum and on a learned CPU usage model.

1 Problem Overview

Explainability and safety in machine learning (ML) are a subject of increasing academic and public concern. As ML continues to grow in success and adoption by wide-ranging industries, the impact of these algorithms' behavior on people's lives is becoming highly non-trivial. Unfortunately, many of the most performant contemporary ML algorithms—neural networks (NNs) in particular—are widely considered black-boxes, with the method by which they perform their duties not being amenable to direct human comprehension. The inability to understand learned components as thoroughly as more traditional software poses serious obstacles to their adoption [Anjomshoae *et al.*, 2019; Adadi and Berrada, 2018; Chakraborti *et al.*, 2019; Guidotti *et al.*, 2019; Xiang and Johnson, 2018; Garcıa and Fernández, 2015; Yasmin *et al.*, 2013; Miller *et al.*, 2017] due to safety concerns, difficulty to debug and maintain, and explicit legal requirements.[1] Symbiotic human-machine interactions can lead to safer and more robust agents, but this task requires effective and versatile communication [Veloso *et al.*, 2015; Rosenthal *et al.*, 2010].

Interpretability of learned systems has been studied in the context of computer science intermittently since at least the late 1980s, particularly in the area of rule extraction (e.g., [Andrews *et al.*, 1995]), adaptive/non-linear control analysis (e.g., [David, 1988]), various rule-learning paradigms [Muggleton, 1999; Agrawal *et al.*, 1993; Piatetsky-Shapiro and Frawley, 1991; Fayyad *et al.*, 1996]), and formal analysis (e.g., [Clarke *et al.*, 2003; Wen *et al.*, 1996; 1997; Walter and Jaulin, 1994; Kearfott, 1996; Moore, 1966]). Notwithstanding this long history, main-stream attention has risen only recently due to increased impact on daily life of opaque AI [Adadi and Berrada, 2018] with novel initiatives focused on the problem domain [Gunning, 2017; Neema, 2017; Aha *et al.*, 2017; XAI, 2018].

Despite this attention, however, most explanatory systems developed for ML are hard-coded to provide a single type of explanation with descriptions at a certain fixed level of abstraction and a fixed type of guarantee about the system behavior, if any. This not only prevents the explanations generated from being digestible by multiple audiences (the end-user, the intermediate engineers who are non-experts in the ML component, and the ML-engineer for instance), but in fact limits the use by any single audience since the levels of abstraction and formal guarantees needed are situation and goal specific, not just a function of the recipient's background. When using a microscope, one varies between low and high magnification in order to find what they are looking for and explore samples; these same capabilities are desirable for XAI for much the same reasons. For example, most consumers of autonomous vehicles may prefer to ask general questions (e.g., "What do you do when you detect a person in front of you?") while an engineer might require more details, specifying precise input parameters and checking actuator compliance; the context of use and the audiencedetermine which level of abstraction is best, and supporting multiple types of abstractions in turn supports more use-cases and au-

[1] For example, the "right to an explanation" legislation [Europ. Parliament, Council of the EU, 2016] adopted by the European Union.

diences. Further, the explanations for such a component need to range from formal guarantees to rough tendencies—one might want to formally guarantee that the car will avoid collisions always, while it might be sufficient that it usually (but perhaps not always) drives slowly when its battery is low. Explainable ML systems should enable such search and smooth variation in need—but at the moment they do not in general.

To address these needs, we introduce Fanoos,[2] an algorithm blending several technologies to interactively provide explanations at varying levels of abstraction and fidelity (i.e., probabilistic versus formal guarentees) to meet user's needs.

2 The Fanoos Approach

Fanoos is an interactive system that allows users to pose a variety of questions grounded in a domain specification (e.g., what environmental conditions cause a robot to swerve left), receive replies from the system, and request that explanations be made more or less abstract. Crucially, Fanoos provides explanations of high fidelity while considering whether the explanation should be formally sound or probabilistically reasonable (which removes the "noise" incurred by measure-zero sets that can plague formal descriptions). Here, we do not focus on the information presentation aethetics so much as ensuring that the proper information can be produced (see Appendix A.5).

2.1 Knowledge Domains and User Questions

In the following discussion, let L be the learned system under analysis (which we will assume is piece-wise continuous), q be the question posed by the user, S_I be the (bounded) input space to L, and S_O be the output space to L, $S_{IO}{=}S_I{\cup}S_O$ be the joint of the input and output space[3], and r be the response given by the system. In order to formulate question q and response r, a library listing basic domain information D is provided to Fanoos; D lists what S_I and S_O are and provides a set of predicates, P, expressed over the domain symbols in S_{IO}, i.e., for all $p \in P$, the free variables $FV(p)$ are chosen from the variable names $V(S_{IO})$, that is $FV(p) \subseteq V(S_{IO})$.

For queries that formally guarantee behavior (see the first three rows in Table 1), we require that the relevant predicates in P can expose their internals as first-order formulas; this enables us to guarantee they are satisfied over all members of a given set via typical SAT-solvers (such as Z3 [De Moura and Bjørner, 2008]). The other query types require only being able to evaluate question q on a variable assignment provided. The members of P can be generated in a variety of ways, e.g., by forming most predicates through procedural generation and then using a few hand-tailored predicates to capture particular cases.[4]. Further, since the semantics of the predicates are grounded, they have the potential to be generated from demonstration.

[2]Meaning lantern in Farsi, it shines a light on black-box AI. Source code and an extended article can be found at https://github.com/DBay-ani/Fanoos.

[3]Subscripts I for input, O for output, etc., are simply symbols.

[4]For example, operational definitions of "high", "low", etc., might be derived from sample data by setting thresholds on quantile values—e.g., 90% or higher might be considered "high".

2.2 Reachability Analysis of L

Having established what knowledge the system is given, we proceed to explain our process. First, users select a question type q_t and the content of the question q_c to query the system. That is, $q = (q_t, q_c)$, where q_t is a member of the first column of Table 1 and q_c is a sentence in disjunctive normal form (DNF) over a subset of P that obeys the restrictions listed in Table 1. To ease discussion, we will refer to variables and sets of variable assignments that p accepts (AC) and those that p illuminates (IL), with the intuition being that the user wants to know what configuration of illuminated variables result in the variable configurations accepted by q_c; see Table 1 for example queries.

With question q provided, we analyze the learned system L to find subsets in the inputs S_I and outputs S_O that agree with configuration q_c and may over-approximate the behavior of L. Specifically, we use CEGAR [Clarke *et al.*, 2000; 2003] with boxes (hyper-cubes) as abstractions and a random choice between a bisection or trisection along the longest axis as the refinement process to find the collect of box tuples, B, specified below:

$$B{=}\{(B_I^{(i)}, B_O^{(i)}) \in \mathrm{BX}(S_I){\times}\mathrm{BX}(S_O) \,|B_O^{(i)} \supseteq L(B_I^{(i)}) \\ \wedge (\exists\, (c,d) \in \mathrm{T}.(\mathit{AC}_q(B_c^{(i)}) \wedge \mathit{IL}_q(B_d^{(i)})))\}$$

where $\mathrm{BX}(X)$ is the set of boxes over space X and $\mathrm{T}{=}\{(O,I),(I,O),(IO,IO)\}$. For feed-forward neural nets with non-decreasing activation functions, B may be found by covering the input space, propagating boxes through the network, testing membership to B of the resulting input- and output-boxes, and refining the input mesh as needed over input-boxes that produce output-boxes overlapping with B. The exact size of the boxes found by CEGAR are determined by a series of hyper-parameters[5] which the system maintains in *states*, a fact we will return to in Section 2.4.

2.3 Generating Descriptions

Having generated B, we produce an initial response, r_0, to the user's query in three steps as follows: (1) for each member of B, we extract the box tuple members that were illuminated by q (in the case where S_{IO} is illuminated, we produce a joint box over both tuple members), forming a set of joint boxes, B'; (2) next, we heuristically search over P for members that describe B' and compute a set of predicates covering all boxes; (3) finally, we format the box covering for user presentation. A sample result answer is shown in listing 1 , and details on steps (2) and (3) how to produce it follow below.

```
1 Description:
2 (0.45789160, 0.61440409, 'x Near Normal Levels')
3 (0.31030792, 0.51991449, 'pole2Angle_rateOfChange Near
      ↪ Normal Levels')
```

Listing 1: Example initial response. Also see Appendix A.5 and B.

Producing a Covering of B'

Our search over P for members covering B' is largely based around the greedy construction of a set covering using a carefully designed candidate score.

[5]For details see [Clarke *et al.*, 2000; 2003; Biere *et al.*, 2003].

Table 1: Description of questions that Fanoos can respond to

Type q_t	Description	Question content q_c			Example
		accepts	illum.	restrictions	
When Do You	Tell all sets (formal consideration of all cases) in the input space S_I that have the potential to cause q_c	Subset s of S_O s.t. there exists a member of s that causes q_c to be true. Found with SAT-solver.	S_I	No variables from S_O	when_do_you move_at_high_speed? (Predicate $p \in D$)
What Do You Do When	Tell all possible learner responses in the collection of input states that q_c accepts	Subset s of S_I s.t. there exists a member of s that causes q_c to be true. Found with SAT-solver.	S_O	No variables from S_I	what_do_you_do_when *and*(close_to_target_orientation, close_to_target_position)?
What are Circumstances in Which	Tell information about what input-output pairs occur in the subset of input-outputs accepted by q_c	Subset s of S_{IO} s.t. there exists a member of s that causes q_c to be true. Found with SAT-solver.	S_{IO}	None	what_are_the_circumstances_in_which *and*(close_to_target_pos,steer_to_right) *or* move_at_low_speed?
...Usually	Statistical tendency. Avoids measure-zero sets that are unlikely seen in practice.	q_c was found to be true at least once via statistical sampling. Discussion in Appendix A.4			when_do_you_usually move_at_low_speed *or* steer_to_left? what_do_you_usually_do_when moving_toward_target_position?

For each member $b \in B'$ we want to find a set of candidate predicates capable of describing the box and for which we would like to form a larger covering. We find a subset $P_b \subseteq P$ that is consistent with b in that each member of P_b passes the checks called for by q_t when evaluated on b (see the Description column of Table 1). This process is expedited by a feasibility check of each member of P on a vector randomly sampled from b, prior to the expensive check for inclusion in P_b. Having P_b, we filter the candidate set further to P'_b: members of P_b that appear most specific to b; notice that in our setting, where predicates of varying abstraction level co-mingle in P, P_b may contain many members that only loosely fit b. The subset P'_b is formed by sampling outside of b at increasing radii (in the ℓ_∞ sense), collecting those members of P_b that fail to hold true at the earliest radius. Importantly, looking ahead to forming a full covering of B, if none of the predicates fail prior to exhausting[6] this sampling, we report P'_b as empty, allowing us to handle b downstream; this avoids "difficult" boxes forcing use of weak predicates that would "wash out" granular details. We try to be specific at this phase assuming that the desired description granularity was determined earlier, presumably during the CEGAR analysis but also (in possible extensions) by altering[7] P.

We next leverage the P'_b sets to construct a covering of B', proceeding in an iterative greedy fashion. Specifically, if C_i is the covering at iteration i, we increment to C_{i+1} as follows:

$$C_{i+1} = (C_i \cup \{p_{i+1}\}) \setminus \{p' \in C_i \mid \forall b \in B'.\, p' \in P'_b \implies p_{i+1} \in P'_b\}$$

with p_{i+1}=argmax$_{p \in P \setminus C_i} \mu(p, C_i)$ where the cover score μ is

$$\mu(p, C_i) = \sum_{b \in B'} \mathbb{1}(|\mathsf{UV}(b, C_i) \cap \mathsf{FV}(p)| > 0)\mathbb{1}(p \in P'_b)$$

and $\mathsf{UV}(b, C_i)$ is the set of variables in b that are not constrained by $C_i \cap P_b$; since the boxes are multivariate and our predicates typically constrain only a subset of the variables, we select predicates based on how many boxes would have open variables covered by them. Let C_F be the final covering produced by this process. Notice that C_F is not necessarily an approximately minimal covering of B with respect to members of P—by forcing $p \in P'_b$ when calculating the cover score μ, we enforce additional specificity criteria that the covering should adhere to.

After forming C_F, any boxes that fail to be covered even partially (e.g., because P_b or P'_b are empty) are reported with a box-range predicate: atomic predicates that simply list the variable range in the box. In other words, we extend C_F to a set C'_F by introducing new predicates specific to each completely uncovered box so that C'_F does cover all boxes in B'.

Cleaning and Formatting Output for User

Having produced C'_F, we collect the covering's content into a formula in DNF. Let

$$d_0 = \bigcup_{b \in B'} \{c \subseteq C'_F \mid \forall p \in c.\ \text{(p covers b)}\}\ .$$

The information needed to compute d_0 is easily gathered from bookkeeping while computing C'_F. Ultimately, the members of d_0 are conjunctions of predicates,[8] with their membership to the set being a disjunction. Prior to actually converting d_0 to DNF, we form d'_0 by removing any $c \in d_0$ that are redundant given the rest of d_0—in practice, d_0 is small enough to simply do full pair-wise comparison.

Finally, r_0 is constructed by listing each c that exists in d'_0 sorted by two relevance scores: first, the proportion of the volume in B' uniquely covered by c, and second by the proportion of total volume c covers in B'.

2.4 User Feedback and Revaluation

Based on the initial response r_0, users can request a more abstract or less abstract explanation. We view this alternate explanation generation as another heuristic search, where the system searches over a series of states to find those that are

[6] The operational meaning of "exhausting", as well as the radii sampled, are all parameters stored in the state.

[7] For example, filtering using a (possibly learned) taxonomy.

[8] From hereon, we use "conjunct" only to reference non-degenerate (i.e., not 1-ary or 0-ary) cases.

deemed acceptable by the user. The states primarily include algorithm hyper-parameters, the history of interaction, the question to be answered, and the set B. Abstraction and refinement operators take a current state and produce a new one, often by adjusting the system hyper-parameters and recomputing B. This state-operator model of user response allows for rich styles of interaction with the user, beyond and alongside of the three-valued responses of acceptance, increase, or decrease of the abstraction level shown in listing 2. As future work, we are exploring the use of active learning leveraging user interactions to select from a collection of operators, with particular interest in bootstrapping the learning process using operationally defined oracles to approximate users.

```
1 Description:
2 (0.16153820, 0.31093854, 'And(endOfPole2_x Near Normal
      ↪ Levels, pole1Angle Low, pole1Angle_rateOfChange
      ↪ High, pole2Angle Near Normal Levels,
      ↪ pole2Angle_rateOfChange High, x High)')
3 (0.14268581, 0.18653883, 'And(endOfPole2_x Near Normal
      ↪ Levels, pole1Angle Low, pole1Angle_rateOfChange
      ↪ High, pole2Angle Near Normal Levels,
      ↪ pole2Angle_rateOfChange Near Normal Levels, x
      ↪ High)')
```

Listing 2: Response to "less abstract" than listing 1

2.5 Capturing the Concept of Abstractness

The criteria to judge degree-of-abstractness in the lay sense are often difficult to capture. We consider abstractness a diverse set of relations that subsume the part-of-whole relation, and thus also generally includes the subset relation. Discussions of formalisms most relevant to computer science can be found in [Cousot and Cousot, 1977; Tennent, 1976; Standardization, 1996][9] and an excellent discussion of the philosophical underpinnings can be found in [Floridi, 2008]. For our purposes, however, defining the notion is not necessary, since we simply wish to utilize the fact of its existence.

In this work, the primary method of producing explanations at desired levels of abstraction is implicit—being without explicit tracking of what predicates are considered more or less abstract. Instead, we leverage the groundedness of our predicates to naturally form partial orderings over semantic states (their "abstractness" level) which in turn are appropriately reflected in syntax. While Fanoos can easily be adapted to leverage explicit information (e.g., taxonomies as in [Srikant and Agrawal, 1995]), we reserve agreement with such "expert labels" as an independent metric of performance for evaluation. As a side benefit, by forgoing direct supervision, we demonstrate that the concept of abstractness is recoverable from the problem semantics and structure.

3 Related Work and Discussion

Many methods are closely related to XAI, stemming from a diverse body of literature and various application domains, e.g., [David, 1988; Andrews *et al.*, 1995; Hayes and Scassellati, 2016; Katz *et al.*, 2017]. Taxonomies of explanation families can be found, e.g. in [Guidotti *et al.*, 2019; Friedrich and Zanker, 2011; Chakraborti *et al.*, 2019], with popular divisions being (1) between explanations that leverage internal mechanics of systems to generate descriptions (decompositional approaches) versus those that exclusively leverage input-output relations (pedagogical)[10], (2) the medium that comprises the explanation (contrast, for example, [Ribeiro *et al.*, 2016; Koul *et al.*, 2019; Hayes and Scassellati, 2016; Kim *et al.*, 2018; Huang *et al.*, 2019; Kim *et al.*, 2018]), (3) theoretical criteria for a good explanation (see, e.g., [Miller *et al.*, 2017]), and (4) specificity and fidelity of explanation. Overall, most of these approaches advocate for producing human-consumable information—whether in natural language, logic, or visual plots—conveying the behavior of the learned system in situations of interest.

Rule-based systems such as expert systems, and work in the (high-level) planning community have a long history of producing explanations in various forms; notably, hierarchical planning [Hayes and Scassellati, 2016; Mohseni-Kabir *et al.*, 2015] naturally lends itself to explanations of multiple abstraction levels. All these methods, however, canonically work on the symbolic level, making them inapplicable to most modern ML methods.

High fidelity, comprehensible rules describing data points can also be discovered with weakly-consistent inductive logic programming [Muggleton, 1999]. However, these approaches are typically pedagogical—not designed to leverage access to the internals of the system—do not offer a variety of descriptions abstractions or strengths, and are typically not interactive. While some extensions of association rule learning consider multiple abstraction levels (e.g., [Srikant and Agrawal, 1995; Han and Fu, 1999]), they are still pedagogical and non-interactive. Further, they only describe subsets of the data they analyze[11] and only understand abstractness syntactically, requiring complete taxonomies be provided explicitly and up-front. Even though it is a work of notable vision—attempting to provide flexible, multi-granular explanations inside a query-response loop for use in data-driven robotics—the work in [Perera *et al.*, 2016] produces explanations using similar principles as generalized association rules[12] and is thus subject to much the same criticism.

Decision support systems (e.g., [Wasylewicz and Scheepers-Hoeks, 2019]) typically allow users to interactively investigate data, with operations such as drill-ups in OLAP (OnLine Analytical Processing) cubes analogous to a simple form of abstraction, Their typical notions of analysis, however, largely operate by truncating portions of data distributions and running analytics packages on selected subregions at user's requests, failing to leverage access to the data-generation mechanism when present, and failing to

[9] [Cousot and Cousot, 1977] features abstraction in verification, [Tennent, 1976] features abstraction at play in interpretating programs, and [Standardization, 1996] is an excellent example of interfaces providing a notion of abstractness in network communications.

[10] We have also found this to be referred to as "introspective" explanations versus "rationalizations", such as in [Kim *et al.*, 2018].

[11] Attempts to ensure all data is described run into the "rare itemset problem" as discussed in [Liu *et al.*, 1999]).

[12] In fact, the explanation mechanism appears less flexible than, e.g., [Srikant and Agrawal, 1995], since [Perera *et al.*, 2016] require separate, unmixable vocabularies for each abstraction level, whereas other approaches—including Fanoos—allow verbiage of different granularity to appear together in descriptions.

provide explicit abstractions or guarantees about the material it presents.

More closely related to our work are approaches to formally analyze neural networks to extract rules, ensure safety, or determine decision stability, which we discuss in more detail below. Techniques related to our inner-loop reachability analysis have been used for stability or reachability analysis in systems that are otherwise hard to analyze analytically. Reachability analysis for FFNNs based on abstract interpretation domains, interval arithmetic, or set inversion has been used in rule extraction and neural net stability analysis [Andrews *et al.*, 1995; Driescher and Korn, 1997; Thrun, 1995; Wen and Callahan, 1996] and continues to be relevant, e.g., [Pulina and Tacchella, 2010; Wang *et al.*, 2018]. While these works provide methods to extract descriptions that faithfully reflect behavior of the network, they neither generally ensure descriptions are comprehensible by users, explore strengthening descriptions by ignoring the effects of measure-zero sets, nor consider varying description abstraction.

Closest in spirit to our work are the planning-related explanations of [Sreedharan *et al.*, 2018],[13] providing multiple levels of abstraction with a user-in-the-loop refinement process, but with a focus on markedly different search spaces, models of human interaction, algorithms for description generation and extraction, and experiments. Further, we attempt to tackle the difficult problem of extracting high-level symbolic knowledge from systems where such concepts are not natively embedded, in contrast to [Sreedharan *et al.*, 2018], who consider purely symbolic systems.

4 Experiments and Results

We analyze learned systems from robotics control and more traditional ML predictions to demonstrate the applicability to diverse domains.

Inverted Double Pendulum (IDP)

The control policy for an inverted double-pendulum is tasked to keep a pole steady and upright; the pole consists of two under-actuated segments attached end-to-end, rotationally free in the same plane; the only actuated component is a cart with the pivot point of the lower segment attached. While similar to the basic inverted single pendulum example in control, this setting is substantially more complicated, since multi-pendulum systems are known to exhibit chaotic behavior [Kellert, 1993; Levien and Tan, 1993]. The trained policy was taken from reinforcement learning literature.[14] The seven-dimensional observation space includes the segment's angles, the cart x-position, their time derivatives, and the y-coordinate of the second pole. The output is a torque in $[-1, 1]$Nm and a state-value estimate, which is not a priori bounded. The values chosen for the input space bounding box were inspired by the 5% and 95% quantile values over simulated runs. We expanded the input box beyond this range to consider rare inputs and observations the model was not necessarily trained on;[15] whether the analysis stays in the region trained-for depends on the user's question.

CPU Usage (CPU)

We also analyze a more traditional ML algorithm for a polynomial kernel regression for modeling CPU usage. Specifically, we use a three-degree fully polynomial basis over a 5-dimensional input space[16] to linearly regress-out a three-dimensional vector. We trained our model using the publicly available data from [Vanschoren *et al.*, 2013].[17] The observations are $[\mathit{lread}, \mathit{scall}, \mathit{sread}, \mathit{freemem}, \mathit{freeswap}]$, which are normalized in respect to the training set min and max prior to featurization, and the response variables we predict are $[\mathit{lwrite}, \mathit{swrite}, \mathit{usr}]$. While the kernel weights may be interpreted in some sense (e.g., indicating which individual feature is, by itself, most influential), the joint correlation between the features and non-linear transformations of the input values makes it far from clear how the model behaves over the original input space. For Fanoos, the input space bounding box was determined from the 5% and 95% quantiles for each input-variable over the full, normalized dataset.

4.1 Experiment Design

Tests were conducted using synthetically generated interactions, with the goal of determining whether our approach properly changes the description abstractness in response to the user request. The domain and question type were randomly chosen, the latter selected among the options listed in Table 1. The questions themselves were randomly generated to have up to four disjuncts, each with conjuncts of length no more than four; conjuncts were ensured to be distinct, and only predicates respecting the constraints of the question-type were used. Interaction with Fanoos post-question-asking was randomly selected from four alternatives (MA means "more abstract" and LA means "less abstract"): Initial refinement of 0.25 or $0.20 \rightarrow$ make LA $\rightarrow$ make MA $\rightarrow$ exit; Initial refinement of 0.125 or $0.10 \rightarrow$ make MA $\rightarrow$ make LA $\rightarrow$ exit. For the results presented here, over 130 interactions were held, resulting in several hundred question-answer-descriptions.

4.2 Metrics

We evaluated the abstractness of each response of Fanoos using metrics across the following categories: reachability analysis, structural description, and expert labeling.

Reachability Analysis

We compare the reachability analysis results over different levels of refinement: we record statistics about distribution of volumes of input-boxes generated during the CEGAR-like analysis, normalized to the input space bounding box so that each axis is in $[0, 1]$ to yield comparable results across domains. The values provide a rough sense of the abstractness

[13] We note that [Sreedharan *et al.*, 2018] was published after the core of our approach was developed; both of our thinkings developed independantly.

[14] https://github.com/araffin/rl-baselines-zoo trained using PPO2 [Schulman *et al.*, 2017]

[15] For instance, the train and test environments exit whenever the end of the second segment is below a certain height. In real applications, users may want to ensure recovery is attempted.

[16] The input space includes cross-terms and the zero-degree element—e.g., x^2y and 1 are members.

[17] Dataset at https://www.openml.org/api/v1/json/data/562

notion implicit in the size of boxes and how they relate to descriptions. For brevity, we only report volume, but note that the distribution of sum-of-side-lengths show similar trends.

Description Structure

Fanoos responds to users with a multi-weighted DNF description. This structure is summarized as follows to give a rough sense of how specific each description is by itself: number of disjuncts, including atomic predicates; number of conjuncts, excluding[18] atomic predicates; number of named predicates (atomic user-defined predicates that occur anywhere in the description, i.e., excluding box-range predicates or conjuncts of atomic predicates); number of box-range predicates that occur anywhere (i.e., in conjuncts as well as stand-alone).

The Jaccard score and overlap coefficients—classic text analysis measures—are calculated over the set of atomic predicates in the descriptions to measure verbage similarity.

Expert Labeling

As humans, we understand which atomic predicates comparatively are more abstract notions in the world, and as such can evaluate the responses based on usage of more vs. less abstract verbage. For simplicity we choose two classes, more abstract (MA) vs. less abstract (LA), and count the number of predicates either (a) accounting for multiplicity, (b) accounting for uniqueness; if an atomic predicate q has label MA (resp., LA) and occurs twice in a sentence, it contributes twice to the (a) score, while contributing only once to (b).

4.3 Results

Summary statistics of our results are shown in Table 2. We are chiefly interested in how a description changes in response to a user-requested abstraction change. Specifically, for pre-interaction state S_t and post-interaction state S_{t+1}, we collect metrics $m(S_{t+1}) - m(S_t)$ that describe *relative* change for each domain-response combination (for the Jaccard and overlap coefficients, the computation is simply $m(S_{t+1}, S_t)$). The medians of these distributions are reported in Table 2.

In summary, the reachability and structural metrics follow the desired trends: when the user requests greater abstraction (MA), the boxes become larger, and the sentences become structurally less complex—namely, they become shorter (fewer disjuncts), have disjuncts that are less complicated (fewer explicit conjuncts, hence more atomic predicates), use fewer unique terms overall (reduction in named predicates) and resort less often to referring to the exact values of a box (reduction in box-range predicates). Symmetric statements can be made for when requests for less abstraction (LA) are issued. From the overlap and Jaccard scores, we can see that the changes in response complexity are not simply due to increased verbosity—simply adding or removing phrases to the descriptions from the prior steps—but also the result of changes in the verbage used.

Trends for the expert labels are similar, though more subtle. We see that use of LA-related terms follows the trend of user requests with respect to multiplicity and uniqueness counts (increases for LA-requests, decreases for MA-requests). We see that the MA counts, when taken relative to the same measures for LA terms, are correlated with user requests in the expected fashion. Specifically, when a user requests greater abstraction (MA), the counts for LA terms decrease far more than those of MA terms, and the symmetric situation occurs for requests of lower abstraction (LA), as expected. These results—labelings coupled with the structural trends—lend solid support that Fanoos can recover substantial elements of an expert's notions about abstractness by leveraging the grounded semantics of the predicates.

[18] By excluding atomic predicates, this provides some rough measure of the number of "complex" terms.

Table 2: Median *relative* change in description before and after Fanoos adjusts the abstraction in the requested direction

			CPU	CPU	IDP	IDP
		Request	LA	MA	LA	MA
Reachability	Boxes	Number	8417.5	-8678.0	2.0	-16.0
	Volume	Max	-0.015	0.015	-0.004	0.004
		Median	-0.003	0.003	-0.004	0.004
		Min	-0.001	0.001	-0.003	0.003
		Sum	-0.03	0.03	-0.168	0.166
Structural	Jaccard		0.106	0.211	0.056	0.056
	Overlap coeff.		0.5	0.714	0.25	0.25
	Conjuncts		1.0	-2.0	0.5	-2.5
	Disjuncts		7.0	-7.5	2.0	-2.5
	Named preds.		1.0	-1.0	1.0	-4.5
	Box-Range preds.		2.0	-2.0	1.5	-1.5
Expert	MA term	Multiplicity	3.0	-3.0	24.0	-20.0
		Uniqueness	0.0	0.0	1.0	-1.5
	LA term	Multiplicity	20.0	-21.5	68.5	-86.0
		Uniqueness	2.0	-2.0	12.0	-14.0

5 Future Work and Conclusions

Fanoos is an explanitory framework for ML systems that mixes technologies ranging from heuristic search to classic verification. Our experiments lend solid support that Fanoos can produce and navigate explanations at multiple granularities and strengths. We are investigating operator-selection learning and further data-driven predicate generation to accelerate knowledge base construction - the latter focusing on representational power, extrapolation intuitiveness, behavioral certainty, and data efficiency. Finally, this work can adopt engineering improvements to ML-specific reachability computations. Further discussion is in Appendix A. We will continue to explore Fanoos's potential, and hope that the community finds inspiration in both the methodology and philosophical underpinnings presented here.

Acknowledgments

This material is based upon work supported by the United States Air Force and DARPA under Contract No. FA8750-18-C-0092. We thank: Nicholay Topic for supporting our spirits at some key junctures of this work; David Held for pointing us to the rl-baselines-zoo repository; David Eckhardt for his proof-reading of earlier versions of this document; the anonymous reviewers for their thoughtful feedback.

References

[Adadi and Berrada, 2018] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (XAI). *IEEE Access*, 6:52138–52160, 2018.

[Agrawal *et al.*, 1993] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *Acm sigmod record*, volume 22, pages 207–216. ACM, 1993.

[Aha *et al.*, 2017] David W Aha, Trevor Darrell, Michael Pazzani, Darryn Reid, Claude Sammut, and Peter Stone. IJCAI 2017 workshop on explainable artificial intelligence (XAI). *Melbourne, Australia, August*, 2017.

[Althoff, 2015] Matthias Althoff. An introduction to CORA 2015. In Goran Frehse and Matthias Althoff, editors, *1st and 2nd International Workshop on Applied veRification for Continuous and Hybrid Systems, ARCH@CPSWeek 2014, Berlin, Germany, April 14, 2014 / ARCH@CPSWeek 2015, Seattle, WA, USA, April 13, 2015*, volume 34 of *EPiC Series in Computing*, pages 120–151. EasyChair, 2015.

[Andrews *et al.*, 1995] Robert Andrews, Joachim Diederich, and Alan Tickle. Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-Based Systems*, 6:373–389, 12 1995.

[Anjomshoae *et al.*, 2019] Sule Anjomshoae, Amro Najjar, Davide Calvaresi, and Kary Främling. Explainable agents and robots: Results from a systematic literature review. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1078–1088. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

[Ballard, 1981] DH Ballard. Generalising the Hough transform to detect arbitary patterns. *Pattern Recognition*, 13, 1981.

[Biere *et al.*, 2003] Armin Biere, Alessandro Cimatti, Edmund M Clarke, Ofer Strichman, Yunshan Zhu, et al. Bounded model checking. *Advances in computers*, 58(11):117–148, 2003.

[Chakraborti *et al.*, 2019] Tathagata Chakraborti, Anagha Kulkarni, Sarath Sreedharan, David E Smith, and Subbarao Kambhampati. Explicability? legibility? predictability? transparency? privacy? security? the emerging landscape of interpretable agent behavior. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 29, pages 86–96, 2019.

[Clarke *et al.*, 2000] Edmund Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. Counterexample-guided abstraction refinement. In *International Conference on Computer Aided Verification*, pages 154–169. Springer, 2000.

[Clarke *et al.*, 2003] Edmund Clarke, Ansgar Fehnker, Zhi Han, Bruce Krogh, Olaf Stursberg, and Michael Theobald. Verification of hybrid systems based on counterexample-guided abstraction refinement. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 192–207. Springer, 2003.

[Cousot and Cousot, 1977] Patrick Cousot and Radhia Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proceedings of the 4th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 238–252, 1977.

[Cropper and Muggleton, 2016] Andrew Cropper and Stephen H. Muggleton. Metagol system. https://github.com/metagol/metagol, 2016.

[David, 1988] Q David. Design issues in adaptive control. *IEEE Transactions on Automatic Control*, 33(1), 1988.

[De Moura and Bjørner, 2008] Leonardo De Moura and Nikolaj Bjørner. Z3: An efficient SMT solver. In *Proceedings of the Theory and Practice of Software, 14th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, TACAS'08/ETAPS'08, pages 337–340, Berlin, Heidelberg, 2008. Springer-Verlag.

[Driescher and Korn, 1997] A Driescher and U Korn. Checking stability of neural narx models: An interval approach. *IFAC Proceedings Volumes*, 30(6):1005–1010, 1997.

[Europ. Parliament, Council of the EU, 2016] Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (General Data Protection Regulation), 2016.

[Fayyad *et al.*, 1996] Usama M Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, Ramasamy Uthurusamy, et al. *Advances in knowledge discovery and data mining*, volume 21. AAAI press Menlo Park, 1996.

[Floridi, 2008] Luciano Floridi. The method of levels of abstraction. *Minds and machines*, 18(3):303–329, 2008.

[Frehse *et al.*, 2011] Goran Frehse, Colas Le Guernic, Alexandre Donzé, Scott Cotton, Rajarshi Ray, Olivier Lebeltel, Rodolfo Ripado, Antoine Girard, Thao Dang, and Oded Maler. Spaceex: Scalable verification of hybrid systems. In *International Conference on Computer Aided Verification*, pages 379–395. Springer, 2011.

[Fridovich-Keil *et al.*, 2018] David Fridovich-Keil, Sylvia L Herbert, Jaime F Fisac, Sampada Deglurkar, and Claire J Tomlin. Planning, fast and slow: A framework for adaptive real-time safe trajectory planning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 387–394. IEEE, 2018.

[Friedrich and Zanker, 2011] Gerhard Friedrich and Markus Zanker. A taxonomy for generating explanations in recommender systems. *AI Magazine*, 32(3):90–98, 2011.

[Garcıa and Fernández, 2015] Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[Guidotti *et al.*, 2019] Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi. A survey of methods for explaining black box models. *ACM computing surveys (CSUR)*, 51(5):93, 2019.

[Gunning, 2017] David Gunning. Explainable artificial intelligence (XAI). *Defense Advanced Research Projects Agency (DARPA), nd Web*, 2, 2017.

[Han and Fu, 1999] Jiawei Han and Yongjian Fu. Mining multiple-level association rules in large databases. *IEEE Transactions on knowledge and data engineering*, 11(5):798–805, 1999.

[Hayes and Scassellati, 2016] Bradley Hayes and Brian Scassellati. Autonomously constructing hierarchical task networks for planning and human-robot collaboration. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5469–5476. IEEE, 2016.

[Hayes and Shah, 2017] Bradley Hayes and Julie A Shah. Improving robot controller transparency through autonomous policy explanation. In *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI*, pages 303–312. IEEE, 2017.

[Herbert *et al.*, 2017] Sylvia L Herbert, Mo Chen, SooJean Han, Somil Bansal, Jaime F Fisac, and Claire J Tomlin. FaSTrack: A modular framework for fast and guaranteed safe motion planning. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 1517–1522. IEEE, 2017.

[Huang *et al.*, 2019] Sandy H. Huang, David Held, Pieter Abbeel, and Anca D. Dragan. Enabling robots to communicate their objectives. *Autonomous Robots*, 43(2):309–326, Feb 2019.

[Katz *et al.*, 2017] Guy Katz, Clark W. Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks, 2017.

[Kearfott, 1996] R Baker Kearfott. Interval computations: Introduction, uses, and resources. *Euromath Bulletin*, 2(1):95–112, 1996.

[Kellert, 1993] Stephen H Kellert. *In the wake of chaos: Unpredictable order in dynamical systems*. University of Chicago press, 1993.

[Kim *et al.*, 2018] Jinkyu Kim, Anna Rohrbach, Trevor Darrell, John F. Canny, and Zeynep Akata. Textual explanations for self-driving vehicles, 2018.

[Koul *et al.*, 2019] Anurag Koul, Alan Fern, and Sam Greydanus. Learning finite state representations of recurrent policy networks, 2019.

[Levien and Tan, 1993] RB Levien and SM Tan. Double pendulum: An experiment in chaos. *American Journal of Physics*, 61(11):1038–1044, 1993.

[Liu *et al.*, 1999] Bing Liu, Wynne Hsu, and Yiming Ma. Mining association rules with multiple minimum supports. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 337–341, 1999.

[Maloney *et al.*, 2010] John Maloney, Mitchel Resnick, Natalie Rusk, Brian Silverman, and Evelyn Eastmond. The Scratch programming language and environment. *ACM Trans. Comput. Educ.*, 10(4), November 2010.

[Miller *et al.*, 2017] Tim Miller, Piers Howe, and Liz Sonenberg. Explainable AI: Beware of inmates running the asylum or: How i learnt to stop worrying and love the social and behavioural sciences. *arXiv preprint arXiv:1712.00547*, 2017.

[Mohseni-Kabir *et al.*, 2015] Anahita Mohseni-Kabir, Charles Rich, Sonia Chernova, Candace L Sidner, and Daniel Miller. Interactive hierarchical task learning from a single demonstration. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*, pages 205–212. ACM, 2015.

[Moore, 1966] Ramon E Moore. *Interval analysis*, volume 4. Prentice-Hall Englewood Cliffs, NJ, 1966.

[Muggleton *et al.*, 2014] Stephen H Muggleton, Dianhuan Lin, Niels Pahlavi, and Alireza Tamaddoni-Nezhad. Meta-interpretive learning: application to grammatical inference. *Machine learning*, 94(1):25–49, 2014.

[Muggleton, 1999] Stephen Muggleton. Inductive logic programming: issues, results and the challenge of learning language in logic. *Artificial Intelligence*, 114(1-2):283–296, 1999.

[Neema, 2017] Sandeep Neema. Assured autonomy, 2017.

[Perera *et al.*, 2016] Vittorio Perera, Sai P Selveraj, Stephanie Rosenthal, and Manuela Veloso. Dynamic generation and refinement of robot verbalization. In *2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 212–218. IEEE, 2016.

[Piatetsky-Shapiro and Frawley, 1991] G Piatetsky-Shapiro and WJ Frawley. Knowledge discovery in databases, 1991.

[Pulina and Tacchella, 2010] Luca Pulina and Armando Tacchella. An abstraction-refinement approach to verification of artificial neural networks. In *International Conference on Computer Aided Verification*, pages 243–257. Springer, 2010.

[Ribeiro *et al.*, 2016] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should I trust you?": Explaining the predictions of any classifier, 2016.

[Rosenthal *et al.*, 2010] Stephanie Rosenthal, Joydeep Biswas, and Manuela Veloso. An effective personal mobile robot agent through symbiotic human-robot interaction. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pages 915–922. International Foundation for Autonomous Agents and Multiagent Systems, 2010.

[Schulman *et al.*, 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[Sreedharan *et al.*, 2018] Sarath Sreedharan, Midhun Pookkottil Madhusoodanan, Siddharth Srivastava, and Subbarao Kambhampati. Plan explanation through search in an abstract model space. pages 67–75, 2018.

[Srikant and Agrawal, 1995] Ramakrishnan Srikant and Rakesh Agrawal. Mining generalized association rules, 1995.

[Standardization, 1996] I Standardization. ISO/IEC 7498-1: 1994 information technology–open systems interconnection–basic reference model: The basic model. *International Standard ISOIEC*, 74981:59, 1996.

[Tennent, 1976] R. D. Tennent. The denotational semantics of programming languages. *Commun. ACM*, 19(8):437–453, August 1976.

[Thrun, 1995] Sebastian Thrun. Extracting rules from artificial neural networks with distributed representations. In *Advances in neural information processing systems*, pages 505–512, 1995.

[Vanschoren *et al.*, 2013] Joaquin Vanschoren, Jan N. van Rijn, Bernd Bischl, and Luis Torgo. OpenML: Networked science in machine learning. *SIGKDD Explorations*, 15(2):49–60, 2013.

[Veloso *et al.*, 2015] Manuela M Veloso, Joydeep Biswas, Brian Coltin, and Stephanie Rosenthal. CoBots: Robust symbiotic autonomous mobile service robots. In *IJCAI*, page 4423, 2015.

[Walter and Jaulin, 1994] E. Walter and L. Jaulin. Guaranteed characterization of stability domains via set inversion. *IEEE Transactions on Automatic Control*, 39(4):886–889, April 1994.

[Wang *et al.*, 2018] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. Formal security analysis of neural networks using symbolic intervals. In *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018*, pages 1599–1614, 2018.

[Wasylewicz and Scheepers-Hoeks, 2019] A. T. M. Wasylewicz and A. M. J. W. Scheepers-Hoeks. *Clinical Decision Support Systems*, pages 153–169. Springer International Publishing, Cham, 2019.

[Wen and Callahan, 1996] Wu Wen and John Callahan. Neuralware engineering: develop verifiable ANN-based systems. In *Proceedings IEEE International Joint Symposia on Intelligence and Systems*, pages 60–66. IEEE, 1996.

[Wen *et al.*, 1996] Wu Wen, John Callahan, and Marcello Napolitano. Towards developing verifiable neural network controller. *Department of Aerospace Engineering, NASA/WVU Software Research Laboratory*, 1996.

[Wen *et al.*, 1997] Wu Wen, Marcello Napolitano, and John Callahan. Verifying stability of dynamic soft-computing systems. 1997.

[XAI, 2018] XAIP. *XAIP 2018: Proceedings of the 1st Workshop on Explainable Planning*, 2018.

[Xiang and Johnson, 2018] Weiming Xiang and Taylor T. Johnson. Reachability analysis and safety verification for neural network control systems. *CoRR*, abs/1805.09944, 2018.

[Yasmin *et al.*, 2013] Mussarat Yasmin, Muhammad Sharif, and Sajjad Mohsin. Neural networks in medical imaging applications: A survey. *World Applied Sciences Journal*, 22(1):85–96, 2013.

A Fanoos Extensions

Here, we elaborate further on-going work for extensions of Fanoos in three thrusts: operator-selection learning (A.1), more advanced data-driven predicate generation (A.2), and engineering improvements (A.3, A.4, and A.5).

A.1 Learning to Select Operators

Fanoos lends itself to the use of active learning to improve the operator-selection procedure, utilizing a proxy-user to help bootstrap the process. Initial developments include an approach that attempts to balance exploration and exploitation, use a priori knowledge of the process and potentially the domain, and leverage structure of the states.[19] We are in the process of carrying out and refining experiments; we hope to report on this work in the near future.

A.2 Predicate Generation

We view the predicate generation problem as important, but distinct from the challenge Fanoos attempts to tackle and one that is amenable to many solutions in context (for instance, the methods we adopted in Section 4 backed by elementary statistics). Importantly, we do not believe requiring predicates simply pushes our core problem to a different arena. Our belief is that Fanoos addresses a large swath of the desiderata and resulting pipeline while allowing improvements to be plugged in for specific subproblems - subproblems with reasonably mature literature, existing methods, and communities working there-in to provide enhancements. Grounded predicate generation is one such subproblem. To elaborate with illustrative examples: databases are considered sensible solutions despite the fact that query optimizers can often be improved, while it is not sensible to say that the traveling salesman problem has been solved on the basis of it reducing to the knapsack problem; we view Fanoos's relationship to the predicate generation problem as similar to the former, not the latter. That noted, we ultimately want an entire system that produces well-tailored explanations for ML users while requiring as little human effort as possible - particularly effort that is unintuitive or requires uncommon expertise. Achieving this greater ease of deployment in more circumstances requires making the subproblems - such as grounded predicate generation - more effortless, more often.

On this front, we are further investigating learning techniques for the generation of domain predicates. Among these, Generalized Hough Transforms [Ballard, 1981] were an early candidate, while more recently our attention has been drawn to advancements in inductive logic programming, Metagol [Muggleton *et al.*, 2014; Cropper and Muggleton, 2016] chief among them. These methods have the potential to provide representational flexibility, predicates amenable to human review, intuitiveness of the extrapolations that may be necessary in the process of generating predicates, and data-efficiency. With Metagol there also is potential for incorporating human-desirable invariants into predicates automatically, and handle input data with a spectrum of structure; in our experiments thus far, however, non-trivial effort has been required to handle numerical values as desired.

It is particularly desirable to enable the generation of predicates that are invariant under certain types of user-defined transformations. For example, one might learn the concept of closing a left-gripper and generalize it to closing at least one gripper by making the truth of the predicate invariant to renaming of hands. We might learn the concept of a sharp turn by only observing left turns and ensure that the hypothesis produced did not become curtailed to leftness by enabling invariance under rotation and reflection (possibly as a wrapper function that converts turns, using these transforms, to left turns). To keep such an approach both practical and meaningful, we must restraint its scope to not venture into more general AI challenges, such as attempting to solve the whole of transfer learning. Implementing invariance relating to naming, copying, and basic geometric transformations seems naturally desirable and not unmanageable in common use cases, and is thus one of the sub-focuses of our research.

The space of possible solutions in the modern ML landscape to engineer predicate generation is vast, particularly if one is willing to admit predicates that work reliably in practice but are not necessarily perfect in all circumstances.[20] For instance, one might utilize an oft-deployed vision-based recognition technique to pick-out objects reliably, despite it not being adversarially robust; indeed, this is not dissimilar to the approach adopted by [Kim *et al.*, 2018], for instance. There might not be a panacea to the predicate generation challenge when it is conceived too broadly, especially if one insists on retaining certainty, transparency, and data-efficiency while attempting to capture sophisticated concepts with minimal human effort. Even in the face of this, we maintain that it is reasonable to suppose that understanding of a target system can most often (in practice) be improved through analysis/ decomposition utilizing these potentially imperfect materials.[21]

Given this overview of how predicate generation may be supported, it is worth taking a moment to reflect on whether predicates of this style are fundamentally necessary to the project of ML explainability. Not all XAI systems may require *explicit* semantic-grounding mechanisms that are *separate* from the learned system being analyzed—for instance, ranking features by weight,[22] generic salience maps, LIME [Ribeiro *et al.*, 2016], and finding exemplar datum [Huang *et al.*, 2019] do not seem to utilize such mechanisms under casual examination. Further, the non-necessity of these ele-

[19] While these three desiderata have tangled interrelations, they do not seem necessarily redundant: the first two seem possible without the third in a problem possessing bandit-esque labeled levers and an opaque oracle, while the first and third seem able to exclude the second if one simply has rewards and distance measures between states.

[20] This might involve, for instance, relaxing one's confidence in a predicate's reliability and/or asking Fanoos questions about usual behavior instead of worst-case behavior.

[21] It may be necessary to conduct the analysis recursively on components.

[22] Whether this counts as an explanation, particularly if the features fail to have clear meaning to the consumer, is debatable. Further, one could rank the weights of the polynomial regression we experimented on, but that would fail to address all of the difficulties we highlighted in Section 4.

ments seems intuitively true[23] if we momentarily indulge in considering how a person explains their behavior—though of course people produce explanations of questionable truthfulness, reliability, and accuracy when it comes to both daily activity and deeper psychological phenomena. Necessity aside, explicit grounding approaches in the vein of this subsection are not uncommon in XAI efforts, and for the foreseeable future will most likely have fundamental benefits and drawbacks compared to the alternative.

A.3 Reachability Analysis Performance

The reachability analyzer of Fanoos is designed in a generic fashion and amenable to having its implementation swapped out without fundamentally altering the overall approach. While reachability analysis is in principle computationally expensive, there are many algorithms that have undesirable worse-case bounds in theory—for example SAT-solvers and the simplex algorithm—but routinely demonstrate useful performance in practice. Methods to potentially draw upon for engineering improvements include the reachability toolboxes CORA [Althoff, 2015] and SpaceEx [Frehse *et al.*, 2011], as well as FaSTrack [Herbert *et al.*, 2017; Fridovich-Keil *et al.*, 2018], a safe planning framework that addresses a related family of problems; all have pushed forward the frontier of practical applications.

A.4 The Sampling Process Backing "Usually True" Queries

Our sampling method can be modified to user-provided distributions, such as estimates of the typical input distribution. Currently, we use a uniform distribution over the hypercubes where the condition is potentially true; this can help examine the range of behaviors/circumstances to a fuller extent and enable counter-factual reasoning. We can envision both sampling approaches as providing distinct values in practice, and thus both potentially useful over the course of dealing with a learned system.

Note that areas of the input space that are logically impossible and have non-zero measure under the uniform measure can be ignored by instructing Fanoos to add a predicate to "what do you usually do when ..." queries and subsequently filtering by that predicate when forming boxes in response to "when do you usually ..." queries.[24] In the output space, Fanoos attempts to deal with the pushforward of the provided learned component under the input distribution. It is worth noting, then, that for some applications, Fanoos in a sense characterizes what the learned component "attempts to do" or "signals" as opposed to what occurs down-stream, which would not atypically involve additional contributors to the system outside of Fanoos's purview[25]. For instance, Fanoos may be able to say that the controller of an autonomous vehicle "attempts a hard-stop", but it may be the case that the vehicle as a whole exhibits a non-identical behavior due to ice on the road, unmodeled performance of low-level controllers, or other conditions either internal or external to the machine.

A.5 Fanoos's User Interface

Our focus while developing Fanoos has been to ensure that the desired information can be generated. In application, a thin front-end can be developed to provide a more aesthetically pleasing presentation, using the vast array of infographics and related tools available. Various input-output wrapper packages from the extensive literature on decision support systems may provide useful guidelines, as well as particularly promising works in interactive robotics (for instance, [Mohseni-Kabir *et al.*, 2015]). Presenting results as English-like sentences using templates (similar to [Hayes and Shah, 2017]), as bar-graphs sorted on height, or word-clouds emphasizing relative importance are all easily facilitated from Fanoos's output. For input, template-supported English-like phrases, drag-and-drop flowcharts, [26] or even basic HTML drop-down menus with sub-categories for option filtering[27] are all possibilities. Although certainly important in practical application, in this paper we rather focus on presenting the underlying algorithmic contributions than the user-facing presentation.

B Extended Example User Interactions

We present here a typical user interaction with our system, Fig. 1. The interested reader can find the predicate definitions with the code at https://github.com/DBay-ani/Fanoos. In practice, if users want to know more about the operational meaning of predicates (e.g., the exact conditions that each predicate tests for), these are revealed in the domain specification[28]—a large part of the point of this system is to provide functionality beyond just cross-referencing code.

Notice that our code uses a Unix-style interaction in the spirit of the `more` command, so not to flood the screen beyond a preset line limit. We also support auto-complete, finishing tokens when unambiguous and listing options available in the context whenever the user hits tab. Whenever we insert a comment in the interface trace that was not originally there, we put `//` at the beginning of the line.

We show the user posing two questions in the inverted double pendulum domain after providing the model they wished to analyze. The response to the first question by the user was a request for less abstraction, while the user response to the second was for greater abstraction. We see that in both cases the explanations adjusted as one would expect, both in respect to the verbosity of the descriptions returned and the verbiage used.

[23] We note that "intuitive" here is not cause for added certainty, since people's intuitions about the organization of cognition, etc., are not always spot-on.

[24] Both of these operations could be supported in a push-button fashion under the hood of a UI.

[25] We do not rule-out a user developing a model for down-stream behavior and incorporating it into something visible to Fanoos—arguably our experiments show signs of that by having actuator limits for the inverted double-pendulum—but naturally this is not something to assume is always present in sufficient fidelity.

[26] A particularly user-friendly and open-source example being [Maloney *et al.*, 2010].

[27] Either grouped by variables of interest or some richer semantic notion.

[28] This is easily facilitated by open-on-click hyperlinks and/or hover text.

```
(Fanoos) when_do_you_usually and(outputtorque_low ,
    statevalueestimate_high )?
Enter a fraction of the universe box length to limit refinement to at
    the beginning.
 Value must be a positive real number less than or equal to one.
```

User requests box length ↓ `0.125` 4

```
Description:
(0.45789160, 0.61440409, 'x Near Normal Levels')
(0.31030792, 0.51991449, 'pole2Angle_rateOfChange Near Normal Levels')
(0.12008841, 0.37943400, 'pole1Angle_rateOfChange High')
(0.06128723, 0.22426058, 'pole2Angle Low')
(0.02395519, 0.13633780, 'vx Low')a
(0.01147175, 0.01359231, 'pole1Angle Low')
type letter followed by enter key: b - break and ask a different
    question,
 l - less abstract , m - more abstract , h - history travel
```

User requests less abstract, continue at (b) ↓ `l` 14

(a) Initial question response, followed by request for less abstract explanation

```
Description:
(0.16153820, 0.31093854, 'And(endOfPole2_x Near Normal Levels,
    pole1Angle Low, pole1Angle_rateOfChange High, pole2Angle Near
    Normal Levels, pole2Angle_rateOfChange High, x High)')
(0.14268581, 0.18653883, 'And(endOfPole2_x Near Normal Levels,
    pole1Angle Low, pole1Angle_rateOfChange High, pole2Angle Near
    Normal Levels, pole2Angle_rateOfChange Near Normal Levels, x
    High)')
(0.11771033, 0.12043966, 'And(pole1Angle Near Normal Levels,
    pole1Angle_rateOfChange Near Normal Levels, pole2Angle High,
    pole2Angle_rateOfChange Low, vx Low)')
(0.06948142, 0.07269412, 'And(pole1Angle High, pole1Angle_rateOfChange
     Near Normal Levels, pole2Angle_rateOfChange High, vx Low, x
    Near Normal Levels)')q
type letter followed by enter key: b - break and ask a different
    question,
 l - less abstract , m - more abstract , h - history travel
```

User break, continue at (c) ↓ `b` 22

(b) Less abstract explanation, user satisfied and continues with different question

```
(Fanoos) what_are_the_circumstances_in_which and(
    pole1angle_rateofchange_low__magnitude ,
    outputtorque_high__magnitude )?
```

Fanoos answers ↓

```
(0.10147897, 0.17831770, pole1angle_on_the_left,
    pole2angle_on_the_left, pole2angle_rateofchange_low__magnitude)
(0.09885232, 0.16335186, pole1angle_on_the_left,
    pole2angle_on_the_left, pole2angle_turning_counterclockwise)
(0.07900125, 0.14467123, pole1angle_on_the_right,
    pole2angle_on_the_right, pole2angle_turning_clockwise)
(0.06693577, 0.12822191, pole1angle_down, pole2angle_to_right,
    statevalueestimate_very_low )q
```

User requests more abstract ↓ `m` 28

```
(0.44378316, 0.48588134, pole2 not near target position)
(0.33605014, 0.36551887, pole2angle_rateofchange_high__magnitude)
(0.22016670, 0.23739381, pole2angle_to_right,
    statevalueestimate_very_low)
```

(c) Next question, initial response, and user request to make more abstract

Figure 1: A sample user session with Fanoos on the inverted double pendulum example