TDM: Trustworthy Decision-Making via Interpretability Enhancement

Daoming Lyu, Fangkai Yang, Hugh Kwon, Wen Dong,
Levent Yilmaz, *Member, IEEE,* and Bo Liu, *Senior Member, IEEE*

***Abstract*—Human-robot interactive decision-making is increasingly becoming ubiquitous, and trust is an influential factor in determining the reliance on autonomy. However, it is not reasonable to trust systems that are beyond our comprehension, and typical machine learning and data-driven decision-making are black-box paradigms that impede interpretability. Therefore, it is critical to establish computational trustworthy decision-making mechanisms enhanced by interpretability-aware strategies. To this end, we propose a Trustworthy Decision-Making (TDM) framework, which integrates symbolic planning into sequential decision-making. The framework learns interpretable subtasks that result in a complex, higher-level composite task that can be formally evaluated using the proposed trust metric. TDM enables the subtask-level interpretability by design and converges to an optimal symbolic plan from the learned subtasks. Moreover, a TDM-based algorithm is introduced to demonstrate the unification of symbolic planning with other sequential-decision making algorithms, reaping the benefits of both. Experimental results validate the effectiveness of trust-score-based planning while improving the interpretability of subtasks.**

***Index Terms*—Symbolic Planning, Sequential Decision-making, Interactive Machine Learning, Trustworthy Machine Learning.**

I. Introduction

FROM self-driving cars to voice assistant on the phone, artificial intelligence (AI) is progressing rapidly. Though AI systems are undeniably powerful and continue to expand their role in our daily lives, emerging issues, such as lack of transparency and uncontrolled risk in decision-making, give rise to a vital concern: ***why should the end-users trust the decision support system?*** Reliance on technology and autonomy is strongly influenced by trust [1]. However, a recent study reveals that a mere belief in interacting with autonomous teammate led to diminished performance and passive behavior among human participants even though they were actually collaborating with a remote human partner [2], indicating that we do not trust the capabilities of AI systems as much as we trust humans.

Building trustworthy AI systems [3, 4, 5, 6] , therefore, becomes an important issue for humans to reap the full spectrum of societal and industrial benefits from AI, as well as for ensuring safety in the use of the system [7]. Indeed, in modern artificial intelligence systems, there are many interactive relations, primarily in the form of *interactions* between humans, between humans and the smart agents (human-computer interaction systems), between agents and the environment (interactive machine learning), and among smart agents (multi-agent systems).

It is also important to note that the trust is mutual. An AI agent must also gauge the trustworthiness of the end-users. However, very few studies exist in agent-to-human trust models. The recent Boeing 737 MAX 8 incidents [8, 9] have tragically shown the dangers of the lack of the automated agent's ability to trust the end-users. The pilots of the crashed airplanes attempted to take control back from a malfunctioning flight control system, but the flight control system could not respond to the change of trust from pilots. Instead of trusting the pilots' decisions, the flight control system continued to override the manual control resulting in fatal crashes.

Besides, AI systems, in general, need to comply with the norms of the social and cultural context if they are expected to operate side by side with humans. Therefore, trust is particularly important for normative agents that are expected to exhibit behavior consistent with the norms of a society or group. From the humans' perspective, we trust things that behave as we expect them to, indicating that trust derives from the process of minimizing the perceived risk [10]. Perceived risk is often defined in terms of uncertainty about the possibility of failure or the likelihood of exhibiting improper behavior.

The typical approach to computationally modeling trust is through beliefs. In multi-agent settings, the agents maintain beliefs on the trustworthiness of other agents influenced by mutual interactions [11]. When end-users are involved, the agent maintains beliefs about the *beliefs* of humans', concerning the trustworthiness of the agent [12]. It is difficult to design an explicit measure of trust without considering different perspectives. Recent work of Hind et al. [13] suggested several elements, such as fairness, robustness, and explainability, to increase trust in AI systems.

To open up the AI black-box and facilitate trust, it is critical to enable the AI system to be reliable, transparent, and explainable so that the system can achieve reproducible results, justify the decisions it makes, and understand what is important in the decision-making process. And this combination of features can be called "interpretability". Deep learning algorithms have been successfully applied to sequential decision-making problems involving high-dimensional sensory inputs such as Atari games [14]. However, deep learning algorithms have limited interpretability and transparency, which gives rise to doubt by the human end-users due to a lack of trust and confidence in the AI systems. Prior research efforts [15] on interpretability often focus on explaining the results of

D. Lyu, H. Kwon, L. Yilmaz, and B. Liu are with the Department of Computer Science and Software Engineering, Auburn University, Auburn, AL, 36849 USA. F. Yang is with NVIDIA Corporation, Bellevue, WA, 36849, USA. W. Dong is with the Department of Computer Science and Engineering, University at Buffalo, Buffalo, NY, 14260 USA.

Corresponding author: Bo Liu <boliu@auburn.edu>.

learning when the underlying problem and architecture are complex. Symbolic planning [16, 17, 18, 19, 20] has been used to break down the complex task into simpler interpretable subtasks. However, existing studies [21, 22] focus on the performance gain of symbolic planning when it is applied to general sequential decision-making rather than interpretability.

Motivated by understanding the task-level behavior of the algorithm and elevating the trust between human end-users and the smart agents, we propose a unified framework of Trustworthy Decision-Making. Symbolic planning is utilized to perform reasoning and planning on explicitly represented knowledge, which results in task-level interpretability. We also introduce a computational trust metric based on the success ratio of symbolic tasks. The key insight is that the success is better measured when the tasks are interpretable. Achieving a reliable success rate of the tasks is equivalent to minimizing perceived risks of the agent's behavior, thereby enhancing its trustworthiness.

The rest of the paper is structured as follows. Section II presents related research in trust, interpretability, followed by a review of prior trust evaluation studies in the extant literature from different perspectives. Section III introduces the background on symbolic planning along with an overview of the interactive sequential decision problem. Section IV provides a detailed description of the TDM framework. We introduce a learning algorithm facilitated by our formal framework in Section V. The optimality analysis for the converged plan generated by the algorithm is presented to substantiate the utility of our strategy. The experimental results of Section VI validate the interpretability of the high-level behavior of the system and the effectiveness of the trust metric while maintaining improved data efficiency. Section VII concludes by summarizing our results and delineating potential avenues of future research.

II. Backgrounds

This section identifies and elaborates several key components of a computational framework for trustworthy decision-making, i.e., the definition of trust, interpretability and its relation to trust, and computational models of trust metrics.

Trust Trust is a subjective perception affected by many factors. One essential impact factor is risk [10]. Following this direction, Marsh [3, 23] presented a formalization of trust and its uses in distributed AI; Liu et al. [24] proposed a stochastic block coordinate ascent policy search algorithm to address the risk management in dynamic decision-making problems. From the perspective of human social interactions, Lee and See [1] studied trust in automation. Specifically, in the context of AI, O'Donovan and Smyth [4] presented two computational models of trust for the recommender systems; Ribeiro et al. [5] proposed the Local Interpretable Model-agnostic Explanations framework to explain the predictions of any classifier so that the model can be transformed into a trustworthy one. In addition, Hind et al. [13] proposed the supplier's declaration of conformity to increase trust in AI services. Another important factor of trust is performance. Prior research in robotics suggests performance and risk are related factors. Desai et al. [25] demonstrated how a robot's failure influences human trust in real-time. Chen et al. [12] proposed to use deliberate failures to calibrate the trust level. In this work, the human users perceive potential failures as risks and then respond according to the level of trust they place in the robot's ability.

Interpretability A key component of an AI system is the ability to explain its decisions, recommendations, predictions, or actions along with the process through which they are made. Studies on interpretability focus on describing the internals of a system in an understandable way to humans [26, 27, 28]. Biran and McKeown [15] proposed an approach to justify the prediction to the user rather than only explaining how the prediction is reached automatically. Another line of research attempts to explain the Markov Decision Processes (MDPs). Elizalde et al. [29] devised an intelligent assistant to assist the power plant operator, which can explain commands generated by an MDP-based planning system. Khan et al. [30] explored the minimal sufficient explanation of policies for factored MDP by populating a set of domain-independent templates. Symbolic planning has been integrated with reinforcement learning in the previous attempts [21, 22] when the underlying MDP is relatively simple. Deep reinforcement learning is often used for larger MDP, and in this direction, the program induction approach is used instead to enable policy interpretability [31]. Our work admits the symbolic knowledge to enable task-level interpretability for sequential decision-making with function approximation, such as deep reinforcement learning, and it is interpretable by construction.

Computational Models of Trust Metric Studies have been conducted on how to evaluate trust quantitatively. Schmidt et al. [32] proposed a customized trust evaluation model based on fuzzy logic for multi-agent systems. Huynh et al. [33] presented a decentralized model to evaluate trust in open systems. Castelfranchi et al. [11] devised a socio-cognitive model of trust by using the fuzzy cognitive maps and introduced a degree of trust derived from the credibility of the trust beliefs. Also, there exists research about trust evaluation in the context of computer networks. Theodorakopoulos and Baras [34] focused on the evaluation processes of trust evidence in Ad Hoc Networks. Sun et al. [35] presented an information-theoretic framework to quantitatively measure trust and model trust propagation in Ad Hoc Networks. Yan et al. [36] and Sun et al. [37] both proposed trust evaluation frameworks concerning security in computer networks. In the context of HRI, one of the earliest trust evaluations is proposed by Lee and Moray [38] and expressed as a regression series. Xu and Dudek [39] also used a regression series to evaluate trust in autonomous vehicles. In another work, Xu and Dudek [40] proposed a Bayesian model. Trust has also been evaluated as states of MDP [12].

To the best of our knowledge, there exists limited and inconclusive research on establishing a computational model of trust to measure the quality of a proposed task. In a quality-centric perspective, the quality of behavior should correspond to the plan's trust score that indicates the level of trustworthiness. Such metrics need to be general enough to make evaluations across different tasks. A relevant research area in psychology is called intrinsic motivation, which is defined as the level of

inherent satisfaction by accomplishing an activity. Interpersonal trust is strongly correlated to the trustor's perceived intrinsic motivation of trustee [41, 42]. Combined with the role of interpretability in trustworthiness, this means a human is more likely to develop feelings of trust towards a smart agent if the human understands that the agent is intrinsically motivated to improve its abilities.

Different from extrinsic motivation, the inherent satisfaction is driven by an internal utility function [43]. This makes it a suitable general metric to make evaluations of different tasks. Intrinsically-motivated learning [44] used the framework of options. An option, or a subtask, can be initiated in some states to control the agent's behavior in a subset of the environment with a terminating condition. Motivated by these factors, we propose a trust metric evaluation framework based on the internal utility function, which can be used to compute the trustworthiness values for the trust score and evaluate the high-level behavior of the system. One of the problems of interpretability is that it can induce unwarranted trust of unreliable agents [45, 46]. We hypothesize that the quality-based metric will alleviate this problem by justifying why the learned plans are good and trustworthy, whereas the abandoned plans are not, thus ensuring the agent's learning process is reliable.

III. Preliminaries

In this section, we introduce the motivation for developing a trust metric and the formulation of a symbolic planning strategy with respect to sequential decision making.

Interactive Sequential Decision-Making. In sequential decision-making problems, an agent takes a sequence of actions to determine its utility. However, the utility is often difficult to model in practical settings when the environment is complex or changes dynamically. In such cases, the utility is determined through feedback from human interaction. We will demonstrate an environment that changes over time in our experiments but will simplify it to model the utility changes to allow agents to train quickly without interaction. One approach is to design multiple similar tasks in differing complexities, learn simple tasks first, and transfer the knowledge to learn more complex tasks [47]. Instead, we will break down complex tasks into simpler sub-tasks. And since it is expensive to consider a complete sequence of decisions, we will model our problem with Markovian property.

Consider a Markov Decision Process (MDP) [1] defined by a tuple $(\mathcal{S}, \mathcal{A}, P^a_{ss'}, r, \gamma)$. We will use the MDP to model the sequential decision-making problem in symbolic representation, where $\mathcal{S}$ is the set of symbolic states and $\mathcal{A}$ is the set of actions. Then $P^a_{ss'}$ is the transition kernel that, given a state $s \in \mathcal{S}$ and an action $a \in \mathcal{A}$, defines the probability the next state will be $s' \in \mathcal{S}$. Moreover, $r(s,a) : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is a reward function bounded by $r_{\max}$, and $0 \leq \gamma < 1$ is a discount factor. A solution to an MDP is a policy $\pi : \mathcal{S} \mapsto \mathcal{A}$ that maps a state to an action.

[1]We follow the style of notation in the 1st edition of reinforcement learning book by Sutton and Barto [48].

To evaluate a policy π, there are two types of performance measures: the expected discounted sum of reward for infinite-horizon problems and the expected un-discounted sum of rewards for finite horizon problems. In this paper we adopt the latter metric defined as $J^\pi_{\text{avg}}(s) = \mathbb{E}[\sum_{t=0}^{T} r_t | s_0 = s]$. We define the *gain reward* $\rho^\pi(s)$ reaped by policy π from s as $\rho^\pi(s) = \lim_{T\to\infty} \frac{J^\pi_{\text{avg}}(s)}{T} = \lim_{T\to\infty} \frac{1}{T}\mathbb{E}[\sum_{t=0}^{T} r_t]$. Gain reward $\rho^\pi(s)$ is often used to measure the lower bound of the expected cumulative rewards of a task w.r.t a given policy. For example, $\rho^\pi(s)$ is instrumental in deciding whether a specific task is rewarding enough to make the "stay-or-leave" decision [49]. More mathematical details of gain reward can be referred to [50, 51, 52, 53].

Symbolic Planning (SP) has been used in tasks that must be highly interpretable to human users, such as interacting and cooperating with mobile robots [16, 17, 18, 19, 20]. By abstracting the planning problem as symbols, SP can solve the problem based on logical reasoning. A symbolic representation is constructed so that it contains the knowledge of objects, properties, and the effects of executing actions in a dynamic system. Such representation can be implemented via a formal, logic-based language such as the Planning Domain Definition Language (PDDL) [54] or an action language [55] that relates to logic programming under answer set semantics (answer set programming) [56].

SP algorithms are also white-box algorithms. With the symbolic knowledge designed to be human-readable, the behavior of an SP agent that plans and reasons based on it is naturally interpretable. Prior works combine symbolic planning with other sequential-decision making algorithms to bring stability to plan-based learning [21, 22]. However, these works only highlight the performance improvement from the prior domain knowledge represented by SP.

Action Language $\mathcal{BC}$. An *action description* D in the language $\mathcal{BC}$ [57] includes two kinds of symbols on signature σ, *fluent constants* that represent the properties of the world, and *action constants* , representing actions that influences the world. A *fluent atom* associates a fluent constant f to a value v and is expressed as $f = v$. In particular, we consider a boolean domain for f. *Causal laws* can now be defined on the fluent atoms and action constants, describing the relationship among fluent atoms and the effects of actions on the value of fluent atoms. A *static law* may state that a fluent atom A is true at a given time step when $A_1, \ldots, A_m$ are true (A **if** $A_1, \ldots, A_m$) or the value of f equals v by default (**default** $f = v$). On the other hand, a *dynamic law* describes an action a (**nonexecutable** a **if** $A_1, \ldots, A_m$) or the effect of a on the fluent atom A (a **causes** A **if** $A_1, \ldots, A_m$). An inertia is a special case of dynamic law that states the value of fluent constant f does not change with time (**inertial** f). An action description is a finite set of causal laws, capturing the domain dynamics as transitions.

Therefore, given action description D, here is an example of how to perform symbolic planning with action language $\mathcal{BC}$. Let $\langle s, a, s' \rangle$ denote a transition from a symbolic state s to a symbolic state s' by a set of action a. Then a planning problem

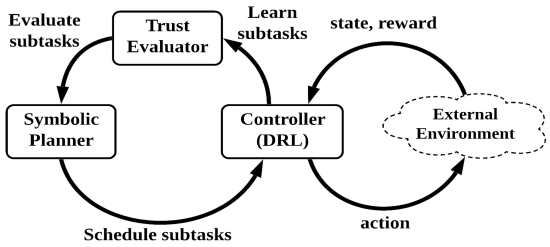


Fig. 1: Architecture illustration

can be formulated as a tuple (I, G, D). The planning problem has a plan of length $l-1$ if and only if there exists a transition path of length l such that $I = s_1$ and $G = s_l$. In the rest of the paper, Π denotes both the plan and the transition path by following that plan. Given the above semantics represented in $\mathcal{BC}$, automated planning can be achieved by an answer set solver such as CLINGO [58], and provide the solution to the planning problem.

IV. THE TDM FRAMEWORK

We will also use the MDP to model the underlying sequential decision-making problem, define by a tuple $(\widetilde{\mathcal{S}}, \widetilde{\mathcal{A}}, \widetilde{P^a_{ss'}}, r, \tilde{\gamma})$, where $\widetilde{\mathcal{S}}$ consists of states of high-dimensional sensory inputs such as pixel images, $\widetilde{\mathcal{A}}$ is the set of primitive actions, and others are similarly defined as before. The goal is to learn both a sequence of subtasks and the corresponding sub-policies, so that executing the sub-policy for each subtask one by one can achieve maximal cumulative reward. It should be noted that our framework is not restricted to high-dimensional sensory inputs. However, we choose to focus on them since learning from such inputs is more challenging and relevant to real-world applications.

We assume human experts can provide a symbolic structure of the problem. The symbolic structure consists of the knowledge of objects, properties, and the preconditions and effects of executing subtasks in a given problem domain. Although this appears to be a lot of effort, it is possible to develop general-purpose action modules [59, 60, 61] as it has been shown that dynamic domains share many actions in common. Therefore, the symbolic formulation for one problem can be adapted to another with a little effort by instantiating a different set of objects or adding a few more rules. Our framework's domain dynamics have a coarse granularity and high-level abstraction, enabling the decision-making process to be robust and flexible when facing uncertainty and domain changes.

With a symbolic representation given by the human expert, the TDM architecture is shown in Fig. 1. A symbolic planner generates a high-level plan, i.e., a sequence of subtasks, based on the trust evaluator's evaluation feedback. We implement a mapping function to perform symbol grounding in our experiments, translating the problem domain to a symbolic representation. However, we assume such functions are generally available in different domains. After symbol grounding, the subtasks in the plan will be sent to the controller to execute so that the sub-policies for the corresponding subtasks are learned. Since it is possible that the controller cannot successfully learn the sub-policies, a metric is introduced to evaluate the competence of learned sub-policies, such as the success ratio over several episodes. After all subtasks in the plan have been executed, the trust evaluator computes the trust score by utilizing the trustworthiness values for subtasks. The evaluation for subtasks is returned to the symbolic planner and is used to generate new plans by either exploring new subtasks or sequencing learned subtasks that supposedly can achieve a higher trust score in the next iteration.

A. A Computational Framework for Trust Evaluation

We formulate the trust evaluation with a success ratio of task execution, a simple measure of quality. Intuitively, it is easier to measure the success of interpretable tasks than non-interpretable tasks. By design, our plans and subtasks are interpretable, and therefore it is easier to devise the corresponding trust score in our problem.

Trust has been evaluated by using the internal utility function in our case to measure the plan's trust score. By doing so, trust evaluation can establish the connection between the trust metric and observation (trust evidence), and justify the decision made upon trust score. Therefore, the subtasks in a plan will be evaluated quantitatively, and the plan with a low trust score will have less ability to be considered for the final solution.

The internal utility score based on the success ratio alone is not a convincing measure of trust. So we tie the trust evaluation to the interpretability perspective and promote the subtasks that are both successful and interpretable as trustworthy. In essence, we try to solve an optimization problem to maximize the explicit trust score subject to the constraint of implicit interpretability:

$$\max_{\Pi} \frac{1}{|\Pi|} \sum_{o \in \Pi} \epsilon(o) \text{ s.t. } I(\Pi) > \delta,$$

where ϵ is the success ratio of a symbolic task o, Π is a symbolic plan, I is an oracle able to numerically qualify the interpretability of a symbolic plan, and δ is the interpretability threshold.

The success ratio with respect to interpretability is the key to our trust evaluation mechanism. But interpretability is a qualitative measure, and it is difficult to formulate a traditional optimization approach to satisfy this constraint. Instead, we use the symbolic representation to naturally induce interpretability in our problem. We discuss the details of our approach in the subsequent sections.

B. Interpretability Enhancement via Symbolic Representation

Let us consider a planning problem (I, G, D). Although a symbolic formulation is possible with various planning or action languages, we will use $\mathcal{BC}$ to represent D as a demonstration. Specifically, we add the following causal laws to D to formulate gain rewards of executing actions and their effects on cumulative plan quality:

- For any symbolic state that contains atoms $\{A_1, \ldots, A_n\}$, D contains static laws of the form:

$$s \textbf{ if } A_1, \ldots, A_n, \text{ for state } s \in \mathcal{S}.$$

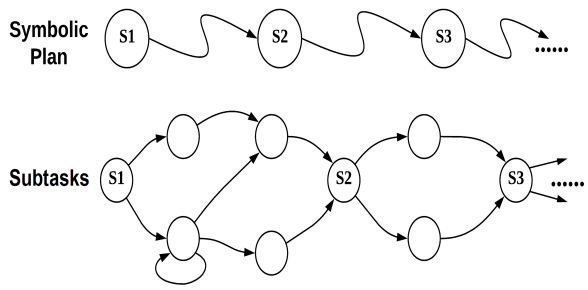


Fig. 2: The mapping from a symbolic transition path to subtasks

- We introduce new fluent symbols of the form $\rho(s, a)$ to denote the gain reward at state s following action a. D contains a static law stating that the gain reward is initialized optimistically by default to promote exploration and is denoted as *INF*:

$$\textbf{default } \rho(s, a) = \mathit{INF}, \text{ for } s \in \mathcal{S}, a \in \sigma_A(D).$$

- We use fluent symbol *quality* to denote the total trustworthiness value of a plan, termed as *plan quality* or *trust score*. D contains dynamic laws of the form

$$a \textbf{ causes } \mathit{quality} = C + Z \textbf{ if } s, \rho(s, a) = Z, \mathit{quality} = C.$$

- D contains a set P of facts of the form $\rho(s, a) = z$.

In our case, I is still the initial symbolic planning state, but the goal state G is also a linear constraint of the form

$$\mathit{quality} > \mathit{quality}(\Pi), \tag{1}$$

for a symbolic plan Π measured by the internal utility function *quality* defined as

$$\mathit{quality}(\Pi) = \sum_{\langle s_i, a_i, s_{i+1}\rangle \in \Pi} \rho(s_i, a_i). \tag{2}$$

It should be noted that (1) in TDM doesn't have the logical constraint part and enables "model-based exploration by planning", which is more suitable for the problems where the trust score drives the agent's behavior.

With the help of declarative paradigms for modeling, symbolic representation has good interpretability in its nature as the hypotheses are understandable and interpretable.

C. Implicit Interpretability Constraint Satisfaction: From Symbolic Transitions to Subtask Options

Our discussion of trust evaluation and interpretability enhancement has been restricted to general machine learning. Symbolic planning can be used in many problem domains, e.g., image classification, and success ratio may be replaced with a suitable metric, such as accuracy, in the respective domain. Here, we narrow our focus to a sequential decision-making problem, which will allow us to devise a specific algorithm based on the TDM framework.

Let us define a symbolic mapping function as $\mathbb{F} : \mathcal{S} \times \widetilde{\mathcal{S}} \mapsto \{\texttt{True}, \texttt{False}\}$. If a symbolic state $s \in \mathcal{S}$ corresponds to $\tilde{s} \in \widetilde{\mathcal{S}}$, then $\mathbb{F} = \texttt{True}$. $\mathbb{F} = \texttt{False}$ if otherwise. We assume an oracle exists to determine whether the symbolic properties specified as fluent atoms of the form $f = v$ in s are true in $\tilde{s}$. Due to advances in computer vision, an oracle such as a perception module for object recognition in images is not uncommon.

An MDP can be considered as a flat decision-making system where the decision is made at each time step. On the contrary, humans make decisions by incorporating temporal abstractions. A *subtask policy* (I, π, β) is temporally extended course of action where a policy $\pi : \mathcal{S} \times \mathcal{A} \mapsto [0, 1]$, a termination condition $\beta : \mathcal{S} \mapsto [0, 1]$, and an initiation set $I \subseteq \mathcal{S}$. A subtask policy (I, π, β) is available in state s_t if and only if $s_t \in I$. When a subtask policy is initiated, the subtask is executed through actions stochastically selected according to π until a terminating symbolic state is reached according to β.

Given $\mathbb{F}$ and a pair of symbolic states $s, s' \in \mathcal{S}$, we can induce a semi-Markov subtask policy, as a triple (I, π, β) where the initiation set $I = \{\tilde{s} \in \widetilde{\mathcal{S}} : \mathbb{F}(s, \tilde{s}) = \texttt{True}\}$, $\pi : \widetilde{\mathcal{S}} \mapsto \widetilde{\mathcal{A}}$ is the intra-subtask policy, and β is the termination condition such that

$$\beta(\tilde{s'}) = \begin{cases} 1 & \mathbb{F}(s', \tilde{s'}) = \texttt{True}, \text{for } \tilde{s'} \in \widetilde{\mathcal{S}} \\ 0 & \text{otherwise} \end{cases}$$

The formulation above maps symbolic transition to a similar structure of subtask policies. It implies that the execution of the subtask will be feasible if the termination condition is 1, while it will be infeasible if the termination condition is 0. Therefore, the interpretability of the subtasks depends on how they contribute to the plan's symbolic transitions. The interpretability of the feasible subtasks is guaranteed by the interpretability of the corresponding symbolic transition. In contrast, the infeasible subtasks are abandoned as uninterpretable as they do not contribute to any interpretable symbolic plan.

By mapping the tasks of the symbolic plan to the subtask policies, we incorporate the hierarchical structure to the decision-making where the symbolic task controls the higher-level planning, and the corresponding subtask policy controls the primitive actions required to accomplish the task. As illustrated in Fig. 2, the subtask policies exhibit Markovian property at the symbolic task level and thus executed sequentially according to the plan.

D. Explicit Success Ratio Maximization

We first assign $t_e(\tilde{s'})$ with some trustworthiness values in a binary form, which is used for computing the trust score. It is defined as:

$$t_e(\tilde{s'}) = \begin{cases} +1, & \beta(\tilde{s'}) = 1 \\ -1, & \text{otherwise} \end{cases} \tag{3}$$

which means that the trustworthiness value will be $+1$ if the subtask terminated at $\tilde{s'}$ can be achieved, otherwise it will be -1. Other more sophisticated distribution (e.g., Bernoulli distribution) can also be modeled other than this 0-1 distribution model in practice.

We further define $r_e(s, o)$ as $r_e(s, o) = f(\epsilon)$, where f is a function of ϵ, and ϵ is the success ratio that denotes the average rate of completing the subtask successfully over the previous 100 episodes. This metric is used to measure whether

the subtask o at symbolic state s is learnable or not. This is different from the definition of $t_e(\tilde{s'})$ since $r_e(s,o)$ requires reliably achieving the subtask after the training by episodes and keeping the success ratio above a certain threshold. Therefore, f is defined as

$$f(\epsilon) = \begin{cases} -\psi & \epsilon < \mathrm{E} \\ r(s,o) & \epsilon \geq \mathrm{E} \end{cases} \tag{4}$$

where ψ is a large positive numerical value, $r(s,o)$ is the reward obtained from the MDP environment by following the subtask o, and the hyper-parameterE is a predefined threshold of success ratio. In our experiments, we set $\mathrm{E} = 0.9$ according to empirical performance observations. Intuitively, it means if the sub-policy can reliably achieve the subtask after the training by episodes, then the trustworthiness value of $r_e(s,o)$ at s' reflects true cumulative reward from the MDP environment by following the subtask; otherwise, the trustworthiness value will be very low (as a large negative number), indicating that the sub-policy performs badly and is probably not learnable. A plan Π of (I,G,D) is considered to be *optimal* iff $\sum_{\langle s,a,s'\rangle} r_e(s,o)$ is maximal among all plans.

V. Algorithm Design

We present a learning algorithm for the TDM framework in Algorithm 1 and show TDM's planning and learning process. Each episode starts with the symbolic planner generating a symbolic plan Π_t for the problem (I,G,D) (Line 4). The symbolic transitions of Π_t are mapped to subtasks (Line 10) to be learned by a controller (Line 11). The controller performs deep Q-learning with t_e using experience replay (Lines 12–15) to estimate the Q value $Q(\tilde{s},\tilde{a};o) \approx Q(\tilde{s},\tilde{a};\theta,o)$, where θ is the parameter of the non-linear function approximator. The observed transition $(\tilde{s}_t,\tilde{a}_t,r_e(\tilde{s}_{t+1},g),\tilde{s}_{t+1})$ is stored as an experience in $\mathcal{D}_o$. The loss at i^{th} iteration is calculated as an expectation over the collected experience and is given by

$$L(\theta;o) = \mathrm{E}_{(\tilde{s},\tilde{a},g,t_e,\tilde{s'})\sim D_o}[r_e + \gamma \max_{\tilde{a'}} Q(\tilde{s},\tilde{a'};\theta_{i-1},o) - Q(\tilde{s},\tilde{a};\theta_i,o)]^2. \tag{5}$$

When the inner loop terminates, a symbolic transition $\langle s_t,a_t,s_{t+1}\rangle$ for the subtask o_t is completed, and we are ready to update the trust score of the subtask (Line 18). Although various learning methods, such as Q-learning, may be used here, we use R-learning [62] to evaluate the trust score as an average reward case rather than a discounted reward case. With R-learning, the update rule becomes

$$\begin{aligned} R_{t+1}(s_t,o_t) &\xleftarrow{\alpha} r_e - \rho_t^{o_t}(s_t) + \max_o R(s_t,o) \\ \rho_{t+1}^{o_t}(s_t) &\xleftarrow{\beta} r_e + \max_o R_t(s_{t+1},o) - \max_o R_t(s_t,o) \end{aligned} \tag{6}$$

The *quality* of Π_t is measured by (2) (Line 20) and the trust score of the plan is updated according to *quality*(Π_t) (Line 21). The symbolic formulation is then updated with the learned ρ values (Line 22). With the symbolic formulation changed, an improved plan may be generated in the next episode. The loop terminates when no such improvement can be made. The algorithm guarantees symbolic level optimality conditioned on R-learning convergence.

ALGORITHM 1: TDM Planning and Learning Loop

Input: $(I,G,D,\mathbb{F})$ where $G = (\textit{quality} > 0)$, and an exploration probability ϵ
Output: An optimal symbolic plan Π^*
$P_0 \Leftarrow \emptyset$, $\Pi \Leftarrow \emptyset$
while True **do**
  $\Pi^* \Leftarrow \Pi$
  Take ϵ probability to solve planning problem and obtain a plan
  $\Pi \Leftarrow$ CLINGO.*solve*$(I,G,D \cup P_t)$
  if $\Pi = \emptyset$ **then**
    return Π^*
  end if
  for symbolic transition $\langle s,a,s'\rangle \in \Pi$ **do**
    Obtain current state $\tilde{s}$
    Correspond to subtask o by using $\mathbb{F}$ to obtain initiation set and terminate condition
    while $\beta(\tilde{s}) \neq 1$ and maximal step is not reached **do**
      Pick up an action $\tilde{a}$ and obtain transition $(\tilde{s},\tilde{a},\tilde{s'},t_e(\tilde{s'}))$
      Store transition in experience replay buffer $\mathcal{D}_o$
      Estimate $Q(\tilde{s},\tilde{a};\theta,o)$ by minimizing loss function (5) when there are sufficient samples in $\mathcal{D}_o$
      Update current state $\tilde{s} \Leftarrow \tilde{s'}$
    end while
    Calculate $r_e(s,o)$ with trustworthiness values
    Update $R(s,o)$ and $\rho^o(s)$ using (6).
  end for
  Calculate quality of Π by (2).
  Update planning goal $G \Leftarrow (\textit{quality} > \textit{quality}_t(\Pi))$.
  Update facts $P_t \Leftarrow \{\rho(s,a) = z : \langle s,a,s'\rangle \in \Pi, \rho_t^a(s) = z\}$
end while

Theorem 1 (Termination)**.** *If the trust evaluator's R-learning converges, Algorithm 1 terminates iff an optimal symbolic plan exists.*

Proof. When R-learning converges, for any transition $\langle s,a,t\rangle$, the increment terms in (6) diminish to 0, which implies

$$\begin{aligned} R(s,o) &= \max_{o'} R(s,o'), \\ \rho^a(s) &= r_e(s,o) - \max_{o'} R(s,o') + \max_{a'} R(t,o') \end{aligned} \tag{7}$$

Algorithm 1 terminates iff there exists an upper bound of plan quality iff there does not exist a plan with a loop L such that $\sum_{\langle s,a,t\rangle\in L} \rho^a(s) > 0$. By (7), it is equivalent to $\sum_{\langle s,o,t\rangle\in L}(r_e(s,o) - R(s,o) + R(t,o)) \leq 0$ iff $\sum_{\langle s,o,t\rangle\in L} r_e(s,o) - R(s_{|L|},o) + R(s_0,o) \leq 0$. Since L is a loop, $s_{|L|} = s_0$, so $\sum_{\langle s,a,t\rangle\in L} r_e(s,o) \leq 0$ iff any plan Π does not have a positive loop of cumulative reward iff optimal plan exists, which completes the proof.

Theorem 2 (Optimality)**.** *If the trust evaluator's R-learning converges, when Algorithm 1 terminates,* Π^* *is an optimal symbolic plan.*

Proof. By Lee et al. [57, Theorem 2], Π^* is a plan for planning problem (I,G,D). For Π_o returned by Algorithm 1 when it terminates, *quality*$(\Pi) \leq$ *quality*(Π^*) for any Π iff

$$\sum_{\langle s,a,t\rangle\in\Pi} \rho^a(s) \leq \sum_{\langle s,a,t\rangle\in\Pi_o} \rho^a(s).$$

By (7), the inequality is equivalent to

$$\sum_{\langle s,a,t\rangle\in\Pi} r_e(s,o) + R(s_{|\Pi|},o) \leq \sum_{\langle s,a,t\rangle\in\Pi_o} r_e(s,o) + R(s_{|\Pi^*|},o).$$

Since $s_{|\Pi|}$ and $s_{|\Pi^*|}$ are terminal states of each symbolic plan with no subtask policies available, we have $\sum_{\langle s,a,t\rangle\in\Pi} r_e(s,o) \leq \sum_{\langle s,a,t\rangle\in\Pi_o} r_e(s,o)$. This completes the proof.

VI. Experiment

We use the Taxi domain [63] to demonstrate the behavior of learning and planning based on the trust score, Grid World [21] to show the ability to learn unmodeled domain knowledge, and on Montezuma's Revenge [14] for interpretability and data efficiency. We use 1M to denote 1 million and 1k to denote 1000.

A. Taxi Domain

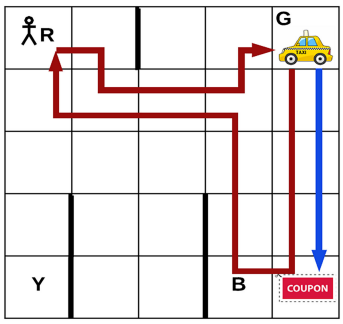
The objective of the Taxi domain is to maximize reward by successfully picking-up and dropping-off a passenger at a specified destination on a grid map (Fig. 3a). A taxi agent moves from a cell to an adjacent cell with -1 reward collected at each time step. The agent receives a reward of 50 for dropping the passenger off at the destination while receiving a reward of -10 when attempting to pick-up or drop-off at an incorrect location. Additionally, there is a one-time reward coupon the taxi can collect at $(4, 4)$. With a reward of 10, we will observe the behavior of learning agents when the environment changes.

Setup. For each episode, the taxi starts at $(0, 4)$, and we will call every 2000 episodes a task. The experiment consists of 10 tasks where the reward for the successful drop-off decreases in each task by 5, i.e., 50 in Task 1, 45 in Task 2, and so forth. Other reward settings stay constant throughout the tasks. We compare TDM with Q-learning.

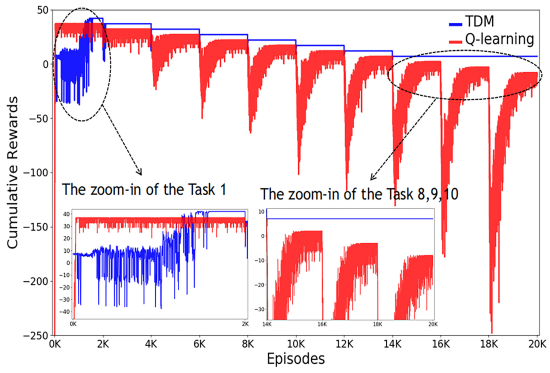
Experimental Results. First, let us consider the optimal policies as shown in Fig. 3a. The dark red colored route that collects the coupon, picking up, and then dropping off the passenger is the optimal policy from Task 1 to Task 7. From Task 8 and onward, the reward for the passenger drop-off (≤ 15) is no longer worth the effort, and hence, the optimal policy changes to simply collecting the coupon as indicated by the blue line. As shown in the learning curve in Fig. 3b, averaged over 10 runs, TDM successfully learns the optimal policy for all Tasks. As noted, the optimal policy changes at Task 8, and TDM quickly abandons the sub-tasks that lead to a sub-optimal policy and learns the new optimal policy. All these are in contrast to Q-learning. While Q-learning converges faster than TDM (see the zoom-in of Task 1 in Fig. 3b) because its model-free exploration is more aggressive than TDM's exploration guided by the trust-score-based planning, Q-learning often fails to learn the optimal policy. Moreover, once Q-learning converges to a policy, it is unable to learn a new one when the environment changes. On the other hand, the trust evaluation mechanism in TDM allows the agent to be flexible and abandon the plan with lower trust score. In practice, the reward is often collected from interaction and changes over time in a dynamic environment. This experiment shows TDM may adapt well in such situations.

B. Grid World

TDM can learn domain details that are not modeled into symbolic knowledge. We demonstrate this behavior with the



(a) Taxi Domain



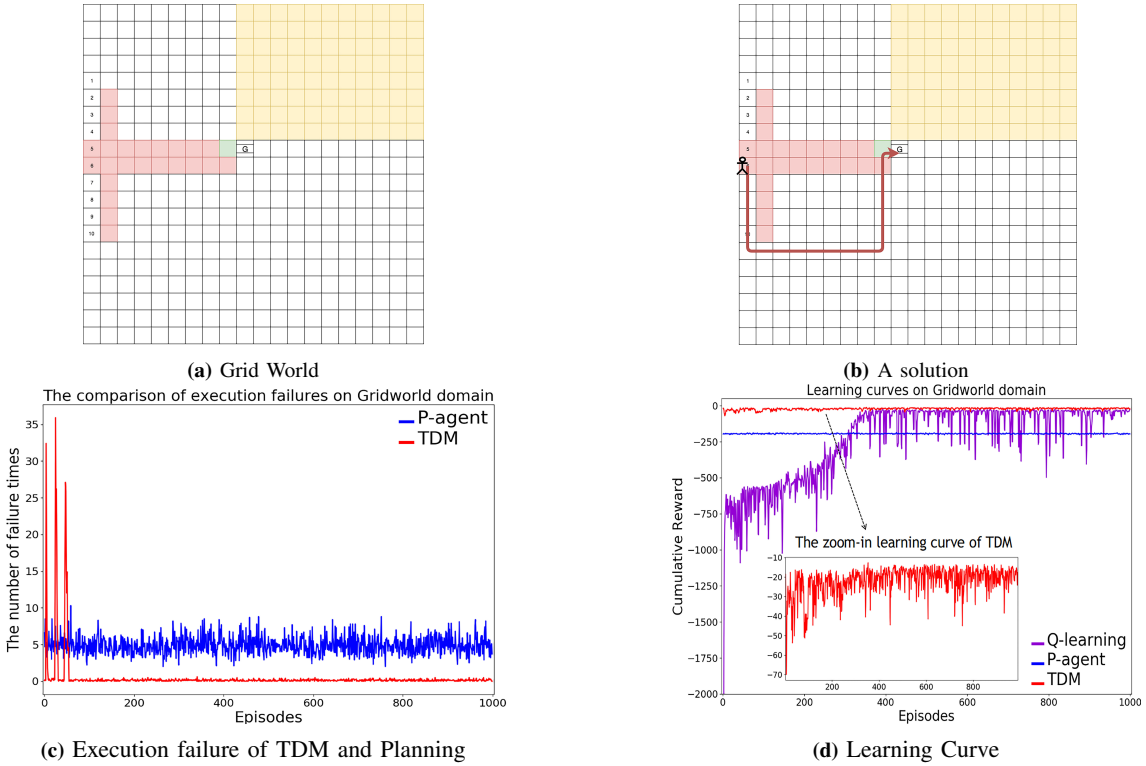
(b) Learning Curve

Fig. 3: Experimental Results on Taxi Domain

Grid World adapted from Leonetti et al. [21], shown in Fig. 4a. An agent must navigate to (9,10), which can only be entered through a door at (9, 9). At the door, the agent must grab a doorknob, turn it, and then push the door to reach the goal, all of which may fail and incur a -10 penalty. As in the Taxi domain, every movement has a reward of -1.

Setup. Horizontal and vertical bumpers represent the domain details that the agent needs to learn. The agent receives an additional penalty for a movement into a bumper. Penalties are -30 and -15 for red and yellow bumpers, respectively. The agent starts at random on one of the numbered grids in the first column. We use Q-learning and a standard planning agent (P-agent) as baselines for this domain. P-agent generates plans using CLINGO and executes the plans without any learning capability.

Experimental Results. Learning curves of cumulative reward are shown in Fig. 4d. TDM achieves the optimal behavior: it avoids the bumpers and learns to reliably open and push through the door (e.g., Fig. 4b), surpassing Q-learning in both the rate of learning and variance of the cumulative reward. P-agent prefers shortest plans which are not ideal plans in this case. P-agent relies on re-planning upon encountering failures without learning. As such, the number of execution failures to enter the door remains high throughout episodes, as shown in Fig. 4c. On the other hand, TDM abandons unsuccessful plans due to their low trust scores and quickly learns to correctly open the door and reach the goal.



(a) Grid World **(b)** A solution

(c) Execution failure of TDM and Planning **(d)** Learning Curve

Fig. 4: Grid World

No.	Layer	Details
1	Convolutional	32 filters, kernel size=8, stride=4, activation='relu'
2	Convolutional	64 filters, kernel size=4, stride=2, activation='relu'
3	Convolutional	64 filters, kernel size=3, stride=1, activation='relu'
4	Fully Connected	512 nodes, activation='relu'
5	Output	activation='linear'

TABLE I: Neural Network Architecture for Montezuma's Revenge

C. Montezuma's Revenge

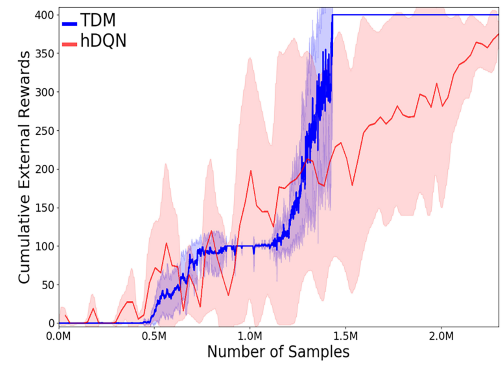
In "Montezuma's Revenge," the player controls a game character through a labyrinth avoiding dangerous enemies and collecting items that are helpful to the player. We focus on the first room of the labyrinth. Here, the player has to collect a key in the room to unlock the door to the next room. In a typical setting for DRL, an agent receives +100 reward for collecting the key and +300 for opening the door with the key. This is a challenging domain for DRL as it involves a long sequence of high-level tasks and many primitive actions to achieve those tasks. This is a challenging domain where the vanilla DQN often fails to learn [14].

Setup. Our experiment setup follows the DQN controller architecture [64] with double-Q learning [65] and prioritized experience replay [66]. The architecture of the deep neural networks is shown in Table I. The experiment is conducted using Arcade Learning Environment (ALE) [67]. We have implemented the symbolic mapping function $\mathbb{F}$ based on ALE API. The binary trustworthiness value follows (3) for when a subtask successfully completes or the agent loses its life. The subtask trust score follows (4) where $\psi = 100$ and define $r(s, o) = -10$ for $\epsilon > 0.9$ to encourage shorter plan. We use hierarchical DQN (hDQN) [64] as the baseline.

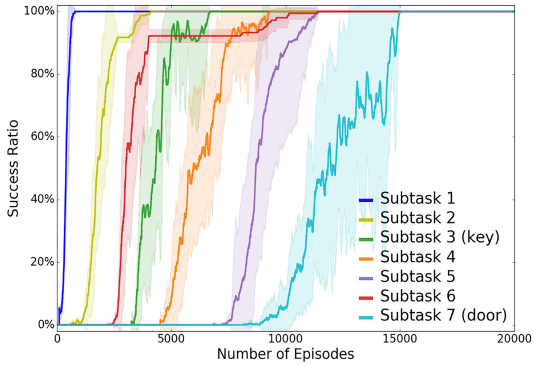
Symbolic Representation. We represent the transition dynamics and domain knowledge in action language $\mathcal{BC}$. There are 6 pre-defined locations or objects: middle platform (`mp`), right door (`rd`), left of rotating skull (`ls`), lower left ladder (`lll`), lower right ladder (`lrl`), and key (`key`). Note that the number of predefined locations or objects depends on the users and their domain knowledge. We then formulate 13 subtasks based on the symbolic locations. The corresponding symbolic transitions from the mapping function $\mathbb{F}$ are shown in Table II. The difference between our and hDQN's subtask definitions should be noted. A subtask is only associated with an object in hDQN; However, in our work, it is defined as a symbolic transition with an initiation set and termination condition based on the states satisfying symbolic properties. With this informative symbolic representation, our approach is more general, descriptive, and interpretable. It also makes sub-policy for each subtask to be more easily learned and subtasks more easily sequenced.

Experimental Results. It is easy to see that TDM is more data-efficient than the baseline hDQN from the learning curve (Fig. 5a). Interpretability requires a qualitative analysis. We will emphasize how trust score guides TDM on planning, learning, and sequencing of the subtasks, and therefore learns the optimal behavior, all of which are visualized in our figures.

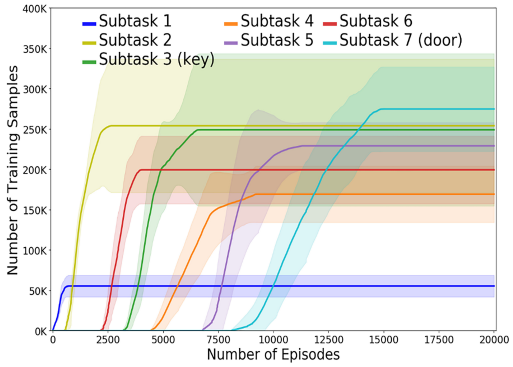
The results have been averaged over 10 runs and we shown mean-variance plots in the Fig. 5a, 5b, 5c. The description of subtasks can refer to both Fig. 5d and Table II. The environment rewards are only given for completing Subtask 3 (picking up the key, $+100$) and Subtask 7 (opening the right door, $+300$). Since other subtasks do not receive any reward from the environment, we can infer from Fig. 5a that TDM first



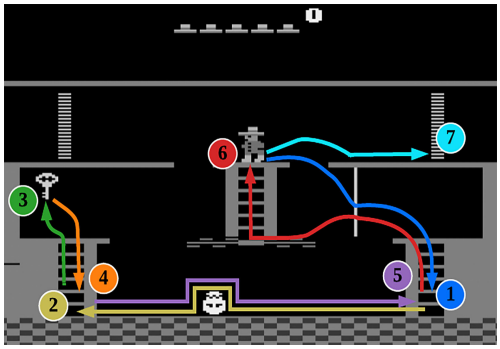
(a) Learning Curve



(b) Success Ratio



(c) Samples Required



(d) Final Solution

Fig. 5: Experimental Results on Montezuma's Revenge

No.	Subtask	Policy learned	In the optimal plan
1	MP to LRL, no key	✓	✓
2	LRL to LLL, no key	✓	✓
3	LLL to key, no key	✓	✓
4	key to LLL, with key	✓	✓
5	LLL to LRL, with or without key	✓	✓
6	LRL to MP, with or without key	✓	✓
7	MP to RD, with key	✓	✓
8	LRL to LS, with or without key	✓	
9	LS to key, with or without key	✓	
10	MP to RD, no key	✓	
11	LRL to key, with or without key		
12	key to LRL, with key		
13	LRL to RD, with key		

TABLE II: Subtasks for Montezuma's Revenge

learns the plan to sequence Subtasks 1–3. The agent learns other subtasks through exploration as the trust-score-based planning encourages executing untried subtasks and try different locations until it reaches the door. At that point, we can see in the learning curve that the agent quickly converges to the optimal plan that sequences through Subtasks 1–7 (Fig. 5d), resulting in the maximal reward of 400. In contrast, hDQN does not reliably achieve the maximal reward even after TDM stably converges. Also, it's difficult to see how hDQN sequenced through its subtasks due to high variance. TDM achieves a smaller variance than hDQN partially because our definition of the subtask is easier to learn than the one defined in hDQN, leading to more robust and stable learning.

The symbolic planning of TDM allows flexibility in reusing the learned subtasks in various plans. Fig. 5b shows that TDM learns Subtask 6 before Subtask 3, but in the optimal plan, Subtask 3 must be sequenced before Subtask 6. This means Subtask 6 has been learned as a part of a different plan but re-selected in the optimal plan, which is possible because subgoals of the subtasks are associated by their starting states and only activated by the planner when the agent satisfies the starting states.

During the experiment, Subtasks 1–10 are successfully learned by controllers (or DQNs), with 7 of them being selected in the final solution with achieving a success ratio of 100%, shown in Fig. 5b. TDM prunes other Subtasks 8–13 based on the low trust score achieved during training. Subtask 8, from the lower right ladder to the left of the rotating skull, reaches a success ratio of 0.9 but later quickly drops back to 0, due to the instability of DQN. Subtasks 9 and 10 reach the required success ratio but are discarded as they do not contribute to the optimal plan, whereas the success ratio on subtasks 11–13 are poor, suggesting they are difficult to learn.

Interpretability and Trust In our proposed framework, the evaluation criteria are designed through human involvement, based on the human expert's interpretation of the agent's execution. The symbolic representation can then be refined to improve the agent's performance. Therefore, the agent's decision-making's trustworthiness is rooted in the trust we place in the underlying symbolic representation. Although many existing interpretable frameworks focus on simplifying the underlying model to generate human-understandable explanations [5, 26, 68], we show that the trade-off between the interpretability and performance of the model is not always necessary. In our case, a performance-based trust metric is used

		TDM	hDQN
Interpretability	Interpretable	Yes	Yes
	Descriptive	Strong	Weak
	Persuasive	Strong	Weak
Trustworthiness	Robustness	Strong	Weak

TABLE III: Comparison of TDM and hDQN

to guide the learning agent to maximize its performance and justify the subtasks it learns. This is, in essence, a separation of descriptive and persuasive explanation tasks [69], where we handle the descriptive task through symbolic representation and perform persuasive explanation through the use of the trust score..

We now relate these concepts to the comparison of our framework to the baseline. In both hDQN and TDM, the bounding box defined by a human expert is utilized, considering the locations of objects only. As a result, the bounding box can be meaningful and interpretable with the human expert's domain knowledge. The subtasks of hDQN are only associated with the bounding box. However, TDM utilizes symbolic representation to derive subtasks except for the bounding box and can provide a relational perspective, such as objects/entities and their relations. This makes the subtasks of TDM more descriptive since a symbolic state may contain more rich information than hDQN. According to the trust evaluation on subtasks, trust scores would motivate the TDM agent to maximize its performance by executing the learnable subtasks while discarding the unlearnable ones. The experiment results show TDM's persuasive explanation is more convincing than hDQN. Also, the variance about the curves (Fig. 5a) demonstrates the degree of robustness when the agent faces uncertainties. The summarization is shown in Table III.

VII. Conclusions

Since interpretability of the high-level behavior is critical to enhancing trust in a hierarchical sequential decision-making problem, we proposed the TDM framework in this paper by integrating symbolic planning with sequential decision-making operating on high-dimensional sensory input. We also presented a TDM-based deep learning algorithm. Deep learning architecture is used to learn low-level control policies for subtasks managed by high-level symbolic planning based on explicit symbolic knowledge. Both theoretical analysis and empirical studies on benchmark problems validate that our trust-score based algorithmic framework brings *task-level interpretability* to deep reinforcement learning and *improved data efficiency* induced by the symbolic-planning-learning framework of the agent.

There are several promising future work potentials in this research direction. For example, it would be intriguing to apply TDM in the human-computer interaction systems, allowing layperson to understand the system behavior and provide meaningful evaluation feedback to the machine. The second interesting direction is to apply the framework of TDM in multi-agent systems. Trust is important in interactions among different agents where the integrity of some agents is not always guaranteed. Interpretability may be used to help improve the behavior of the agents and lead to a better equilibrium.

References

[1] J. D. Lee and K. A. See, "Trust in automation: Designing for appropriate reliance," *Human factors*, vol. 46, no. 1, pp. 50–80, 2004.

[2] M. Demir, N. J. McNeese, and N. J. Cooke, "The impact of perceived autonomous agents on dynamic team behaviors," *IEEE TETCI*, vol. 2, no. 4, pp. 258–267, 2018.

[3] S. Marsh, "Trust in distributed artificial intelligence," in *European Workshop on Modelling Autonomous Agents in a Multi-Agent World*. Springer, 1992, pp. 94–112.

[4] J. O'Donovan and B. Smyth, "Trust in recommender systems," in *Proceedings of the 10th international conference on Intelligent user interfaces*. ACM, 2005, pp. 167–174.

[5] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?: Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 2016, pp. 1135–1144.

[6] Y.-S. Ong and A. Gupta, "Air 5: Five pillars of artificial intelligence research," *IEEE TETCI*, vol. 3, no. 5, pp. 411–415, 2019.

[7] K. A. Hoff and M. Bashir, "Trust in automation: Integrating empirical evidence on factors that influence trust," *Human Factors*, vol. 57, no. 3, pp. 407–434, 2015.

[8] L. Josephs, "Black box data from doomed ethiopian airlines jet show 'clear similarities' between both boeing 737 max crashes," *CNBC*, March 18, 2019. [Online]. Available: https://www.cnbc.com/2019/03/18/french-investigator-clear-similarities-between-boeing-737-max-crashes.html

[9] J. Ortiz, "Fatal flights: What we know about boeing max 8 crashes in ethiopia and indonesia," *USA Today*, April 4, 2019. [Online]. Available: https://www.usatoday.com/story/news/nation/2019/04/04/boeing-max-8-plane-crashes-what-we-know-two-fatal-flights/3371347002/

[10] D. M. Rousseau, S. B. Sitkin, R. S. Burt, and C. Camerer, "Not so different after all: A cross-discipline view of trust," *Academy of management review*, vol. 23, no. 3, pp. 393–404, 1998.

[11] C. Castelfranchi, R. Falcone, and G. Pezzulo, "Trust in information sources as a source for trust: a fuzzy approach," in *AAMAS*. ACM, 2003, pp. 89–96.

[12] M. Chen, S. Nikolaidis, H. Soh, D. Hsu, and S. Srinivasa, "Trust-aware decision making for human-robot collaboration: Model learning and planning," *arXiv preprint arXiv:1801.04099*, 2018.

[13] M. Hind, S. Mehta, A. Mojsilovic, R. Nair, K. N. Ramamurthy, A. Olteanu, and K. R. Varshney, "Increasing trust in ai services through supplier's declarations of conformity," *arXiv preprint arXiv:1808.07261*, 2018.

[14] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[15] O. Biran and K. R. McKeown, "Human-centric justification of machine learning predictions." in *IJCAI*, 2017, pp.

1461–1467.

[16] M. Hanheide, M. Göbelbecker, G. S. Horn *et al.*, "Robot task planning and explanation in open and uncertain worlds," *Artificial Intelligence*, 2015.

[17] K. Chen, F. Yang, and X. Chen, "Planning with task-oriented knowledge acquisition for a service robot." in *IJCAI*, 2016, pp. 812–818.

[18] P. Khandelwal, S. Zhang, J. Sinapov, M. Leonetti, J. Thomason, F. Yang, I. Gori, M. Svetlik, P. Khante, V. Lifschitz, and P. Stone, "Bwibots: A platform for bridging the gap between ai and human–robot interaction research," *The International Journal of Robotics Research*, vol. 36, no. 5-7, pp. 635–659, 2017.

[19] I.-B. Jeong, W.-R. Ko, G.-M. Park, D.-H. Kim, Y.-H. Yoo, and J.-H. Kim, "Task intelligence of robots: Neural model-based mechanism of thought and online motion planning," *IEEE TETCI*, vol. 1, no. 1, pp. 41–50, 2016.

[20] D. Lyu, F. Yang, B. Liu, and S. Gustafson, "Sdrl: interpretable and data-efficient deep reinforcement learning leveraging symbolic planning," in *AAAI*, vol. 33, no. 01, 2019, pp. 2970–2977.

[21] M. Leonetti, L. Iocchi, and P. Stone, "A synthesis of automated planning and reinforcement learning for efficient, robust decision-making," *Artificial Intelligence*, vol. 241, pp. 103–130, 2016.

[22] F. Yang, D. Lyu, B. Liu, and S. Gustafson, "Peorl: Integrating symbolic planning and hierarchical reinforcement learning for robust decision-making," in *IJCAI*, 2018.

[23] S. P. Marsh, "Formalising trust as a computational concept," Ph.D. dissertation, University of Stirling, 1994.

[24] B. Liu, T. Xie, Y. Xu, M. Ghavamzadeh, Y. Chow, D. Lyu, and D. Yoon, "A block coordinate ascent algorithm for mean-variance optimization," *arXiv preprint arXiv:1809.02292*, 2018.

[25] M. Desai, P. Kaniarasu, M. Medvedev, A. Steinfeld, and H. Yanco, "Impact of robot failures and feedback on real-time trust," in *Proceedings of the 8th ACM/IEEE international conference on Human-robot interaction*. IEEE Press, 2013, pp. 251–258.

[26] Z. C. Lipton, "The mythos of model interpretability," *arXiv preprint arXiv:1606.03490*, 2016.

[27] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *arXiv preprint arXiv:1702.08608*, 2017.

[28] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. Specter, and L. Kagal, "Explaining explanations: An approach to evaluating interpretability of machine learning," *arXiv preprint arXiv:1806.00069*, 2018.

[29] F. Elizalde, L. E. Sucar, A. Reyes, and P. d., "An mdp approach for explanation generation." in *ExaCt*, 2007, pp. 28–33.

[30] O. Z. Khan, P. Poupart, and J. P. Black, "Minimal sufficient explanations for factored markov decision processes." in *ICAPS*, 2009.

[31] A. Verma, V. Murali, R. Singh, P. Kohli, and S. Chaudhuri, "Programmatically interpretable reinforcement learning," *arXiv preprint arXiv:1804.02477*, 2018.

[32] S. Schmidt, R. Steele, T. S. Dillon, and E. Chang, "Fuzzy trust evaluation and credibility development in multi-agent systems," *Applied Soft Computing*, vol. 7, no. 2, pp. 492–505, 2007.

[33] T. D. Huynh, N. R. Jennings, and N. R. Shadbolt, "An integrated trust and reputation model for open multi-agent systems," *Autonomous Agents and Multi-Agent Systems*, vol. 13, no. 2, pp. 119–154, 2006.

[34] G. Theodorakopoulos and J. S. Baras, "Trust evaluation in ad-hoc networks," in *Proceedings of the 3rd ACM workshop on Wireless security*. ACM, 2004, pp. 1–10.

[35] Y. Sun, W. Yu, Z. Han, and K. R. Liu, "Trust modeling and evaluation in ad hoc networks," in *GLOBECOM'05. IEEE Global Telecommunications Conference, 2005.*, vol. 3. IEEE, 2005, pp. 6–pp.

[36] Z. Yan, P. Zhang, and T. Virtanen, "Trust evaluation based security solution in ad hoc networks," in *Proceedings of the Seventh Nordic Workshop on Secure IT Systems*, vol. 14, 2003.

[37] Y. L. Sun, Z. Han, W. Yu, and K. R. Liu, "A trust evaluation framework in distributed networks: Vulnerability analysis and defense against attacks." in *INFOCOM*, vol. 6, 2006, pp. 1–13.

[38] J. Lee and N. Moray, "Trust, control strategies and allocation of function in human-machine systems," *Ergonomics*, vol. 35, no. 10, pp. 1243–1270, 1992.

[39] A. Xu and G. Dudek, "Towards modeling real-time trust in asymmetric human–robot collaborations," in *Robotics Research*. Springer, 2016, pp. 113–129.

[40] ——, "Optimo: Online probabilistic trust inference model for asymmetric human-robot collaborations," in *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 2015, pp. 221–228.

[41] J. Holmes and J. Rempel, "Trust in close relationships," *Journal of Personality and Social Psychology*, vol. 49, pp. 95–112, 07 1985.

[42] C. Moorman, R. Deshpandé, and G. Zaltman, "Factors affecting trust in market research relationships," *Journal of Marketing*, vol. 57, no. 1, pp. 81–101, 1993.

[43] P. Oudeyer and F. Kaplan, "What is intrinsic motivation? a typology of computational approaches," *Frontiers in neurorobotics*, vol. 1, p. 6, 2009.

[44] N. Chentanez, A. G. Barto, and S. P. Singh, "Intrinsically motivated reinforcement learning," in *NIPS*, 2005, pp. 1281–1288.

[45] N. Wang, D. V. Pynadath, and S. G. Hill, "Trust Calibration within a Human-Robot Team: Comparing Automatically Generated Explanations," in *Proceddings of the 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, Mar. 2016, pp. 109–116.

[46] X. J. Yang, V. V. Unhelkar, K. Li, and J. A. Shah, "Evaluating effects of user experience and system transparency on trust in automation," in *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, ser. HRI, 2017, pp. 408–416.

[47] B. Peng, J. MacGlashan, R. Loftin, M. L. Littman, D. L. Roberts, and M. E. Taylor, "Curriculum design for machine learners in sequential decision tasks," *IEEE*

TETCI, vol. 2, no. 4, pp. 268–277, 2018.

[48] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT press Cambridge, 1998.

[49] S. Shuvaev, S. Starosta, D. Kvitsiani, A. Kepecs, and A. Koulakov, "R-learning in actor-critic model offers a biologically relevant mechanism for sequential decision-making," *NeurIPS*, vol. 33, 2020.

[50] M. L. Puterman, *Markov Decision Processes*. New York, USA: Wiley Interscience, 1994.

[51] S. Mahadevan, "Average reward reinforcement learning: Foundations, algorithms, and empirical results," *Machine learning*, vol. 22, no. 1, pp. 159–195, 1996.

[52] P. Tadepalli, D. Ok *et al.*, "H-learning: A reinforcement learning method to optimize undiscounted average reward," Oregon State University, Tech. Rep. 94–30–1, 1994.

[53] S. Mahadevan and B. Liu, "Basis construction from power series expansions of value functions," *NIPS*, vol. 23, pp. 1540–1548, 2010.

[54] D. McDermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins, *PDDL-the planning domain definition language*, 1998.

[55] M. Gelfond and V. Lifschitz, "Action languages," *ETAI*, 1998.

[56] V. Lifschitz, "What is answer set programming?" in *AAAI*. MIT Press, 2008, pp. 1594–1597.

[57] J. Lee, V. Lifschitz, and F. Yang, "Action Language $\mathcal{BC}$: A Preliminary Report," in *IJCAI*, 2013.

[58] M. Gebser, B. Kaufmann, and T. Schaub, "Conflict-driven answer set solving: From theory to practice," *Artificial Intelligence*, vol. 187-188, pp. 52–89, 2012.

[59] S. T. Erdoğan and V. Lifschitz, "Actions as special cases," in *KR*, 2006, pp. 377–387.

[60] S. T. Erdoğan, "A library of general-purpose action descriptions," Ph.D. dissertation, University of Texas at Austin, 2008.

[61] D. Inclezan and M. Gelfond, "Modular action language alm," *TPLP*, vol. 16, no. 2, pp. 189–235, 2016.

[62] S. Mahadevan, "Average reward reinforcement learning: Foundations, algorithms, and empirical results," *Machine Learning*, vol. 22, pp. 159–195, 1996.

[63] A. Barto and S. Mahadevan, "Recent advances in hierarchical reinforcement learning," *Discrete Event Systems Journal*, vol. 13, pp. 41–77, 2003.

[64] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum, "Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation," in *NIPS*, 2016, pp. 3675–3683.

[65] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning." in *AAAI*, vol. 16, 2016, pp. 2094–2100.

[66] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.

[67] M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, "The arcade learning environment: An evaluation platform for general agents," *Journal of Artificial Intelligence Research*, vol. 47, pp. 253–279, 2013.

[68] D. A. Wood, "A transparent open-box learning network provides insight to complex systems and a performance benchmark for more-opaque machine learning algorithms," *Advances in Geo-Energy Research*, vol. 2, no. 2, pp. 148–162, 2018.

[69] B. Herman, "The promise and peril of human evaluation for model interpretability," *arXiv preprint arXiv:1711.07414*, 2017.

Daoming Lyu received the B.S. degree in Electrical Engineering from Southwest University, Chongqing, China, in 2011, and the M.E. degree in Biomedical Engineering from Zhejiang University, Hangzhou, China, in 2015. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Software Engineering, Auburn University, Auburn, AL, USA. His current research interests include Reinforcement Learning, Neuro-Symbolic AI, Trustworthy Decision-making and Healthcare Informatics.

Fangkai Yang obtained his Ph.D. from Department of Computer Science, University of Texas at Austin in 2014. His research focus is logic-based artificial intelligence, in particular, knowledge representation and reasoning, non-monotonic reasoning and answer set programming. He is also interested in application of logic-based AI in autonomous systems, mobile robots and self-driving vehicles. Dr. Yang is a senior scientist in NVIDIA focusing on behavior planning and prediction in self-driving vehicles.

Hugh Kwon received the B.S. and M.S. degrees in Computer Science from Columbus State University in 2009 and 2018, respectively. He is currently pursuing the Ph.D. with the Department of Computer Science and Software Engineering at Auburn University. His research interests include reinforcement learning, distributed learning, and safe learning.

Wen Dong is an Assistant Professor of Computer Science and Engineering with a joint appointment at the Institute of Sustainable Transportation and Logics at the State University of New York at Buffalo. His research focuses on developing machine learning and signal processing tools to study the dynamics of large social systems in vivo. He has a PhD degree from the MIT Media Laboratory.

Levent Yilmaz is the Alumni Distinguished Professor of Computer Science and Software Engineering at Auburn University with a courtesy appointment in Industrial and Systems Engineering. He holds M.S. and Ph.D. degrees in Computer Science from Virginia Tech. His research interests are in theory and methodology of modeling and simulation, agent-directed simulation, cognitive systems, and model-driven science and engineering for complex adaptive systems.

Bo Liu is a Tenure-Track Assistant Professor with the Department of Computer Science and Software Engineering, Auburn University, Auburn, AL, USA. He received the Ph.D. degree from the University of Massachusetts Amherst, Amherst, MA, USA, in 2015. He has over 30 publications on several notable venues. He is the recipient of the UAI'2015 Facebook best student paper award and the Amazon research award in 2018. He is an Associate Editor of IEEE Transactions on Neural Networks and Learning Systems (IEEE-TNN), a senior member of IEEE, and a member of AAAI, ACM, and INFORMS.