Hierarchical Reinforcement Learning for Pedagogical Policy Induction

Guojing Zhou, Hamoon Azizsoltani, Markel Sanz Ausin, Tiffany Barnes, and Min Chi

Department of Computer Science
North Carolina State University, Raleigh, NC 27695, USA
{gzhou3,hazizso,msanzau,tmbarnes,mchi@ncsu.edu}

Abstract. In interactive e-learning environments such as Intelligent Tutoring Systems, there are pedagogical decisions to make at two main levels of granularity: whole problems and single steps. Recent years have seen growing interest in data-driven techniques for such pedagogical decision making, which can dynamically tailor students' learning experiences. Most existing data-driven approaches, however, treat these pedagogical decisions *equally, or independently*, disregarding the long-term impact that tutor decisions may have across these two levels of granularity. In this paper, we propose and apply an offline, off-policy Gaussian Processes based Hierarchical Reinforcement Learning (HRL) framework to induce a hierarchical pedagogical policy that makes decisions at both problem and step levels. In an empirical classroom study with 180 students, our results show that the HRL policy is significantly more effective than a Deep Q-Network (DQN) induced policy and a random *yet reasonable* baseline policy.

Keywords: Hierarchical Reinforcement Learning · Pedagogical Policies

1 Introduction

Interactive e-Learning Environments such as Intelligent Tutoring Systems (ITSs) and educational games have become increasingly prevalent in educational settings. In domains like math and science, solving a problem often requires producing one or multiple steps, each of which is the result of applying a domain principle or rule. For example, $2x+5=9$ can be solved for x in two steps: 1) subtract the same term 5 from both sides of the equation; and 2) divide both sides by 2. Tutoring in such domains is thus often structured as a two-loop procedure [35]: the outer loop makes *problem* level decisions, such as problem selection; while the inner loop controls *step* level decisions, such as whether or not to give hints or give a feedback. As a result, there are decisions to make and opportunities to give at *different levels of granularity*, such as hints, worked examples, immediate feedback, or suggested subgoals, and some are more important or impactful than others. Human decision-makers treat these distinct levels of granularity differently and are capable of selecting between them [7, 12].

Data-driven approaches, and especially reinforcement learning (RL), have been shown to improve the effectiveness of ITSs [4, 28, 19, 5, 29, 9, 10, 39]. However, most prior applications of RL for pedagogical policy induction treat all system decisions

equally or independently and do not account for the long-term impact of higher-level actions or the interaction of decisions made at different levels. In this paper, we propose and apply an offline, off-policy Gaussian Processes-based (GP-based) Hierarchical Reinforcement Learning (HRL) framework to induce a hierarchical pedagogical policy at two levels of granularity: *problem* and *step*. More specifically, our HRL policy will first make a problem-level decision and then make step-level decisions based on the problem-level decision. In this study, for example, our HRL policy first decides whether the next *problem* should be a worked example (WE), problem solving (PS), or a faded worked example (FWE). In WEs, students observe how the tutor solves a problem; in PSs students solve the problem themselves; in FWEs, the students and the tutor *co-construct* the solution. Based on the problem-level decision, the HRL policy then makes step-level decisions on whether to elicit the next solution step from the student, or to show it to the student directly. We refer to such decisions as *elicit/tell* decisions. If WE is selected, an all-tell step policy will be carried out; if PS is selected, an all-elicit policy will be executed; finally if FWE is selected, the tutor will decide whether to elicit or to tell a step based on the corresponding induced step-level policy. Both WE and PS can be seen as two extreme ends of FWEs. Therefore, one non-hierarchical way to make decisions would be to focus on step-level decisions alone.

In a classroom study, we compared the HRL induced hierarchical policy (HRL) with two step-level policies: a Deep Q-Network induced policy (DQN) and a *random yet reasonable* (Random) policy because both *elicit* and *tell* are always considered to be *reasonable* educational interventions in our learning context. 180 students were randomly assigned to three conditions and our results showed that the HRL policy was significantly more effective than the DQN and Random policies, and no significant difference was found between the two latter policies. For time on task, no significant difference was found between the HRL condition and Random but the former (HRL) spent significantly more time than DQN. Finally, the induced HRL policy is more likely to select PS and FWE than WE, which confirmed our hypothesis that HRL would provide the right balance to pedagogical decision making, targeting WEs and tells to just those problems and steps that need them.

2 Background & Related Work

2.1 Previous Research on Applying RL to ITSs

Generally speaking, RL approaches can be classified as online, where the agent learns a policy in real time by interacting with the environment, or offline, where the agent learns from pre-collected training data. RL approaches can also be divided into on-policy vs. off-policy, based on the relationship between their behavior and estimation policies [32]. In on-policy RL, the behavior policy used to control how the agent explores the environment (online), or collects training data (offline), is the same as the estimation policy being learned. In off-policy methods, these two policies may be unrelated. Both online and offline RL approaches have been used for pedagogical policy induction in recent years; among them, prior research mainly took an off-policy RL approach [3, 9, 10, 19, 36, 28, 4, 39, 13]. Next, we will describe prior RL work from the online vs. offline perspective.

Online RL research to induce pedagogical policies has often relied on simulations or simulated students. As a consequence, the success of these approaches is heavily dependent on the accuracy of the simulations. Beck et al. [3] applied temporal difference, with off-policy ϵ-greedy exploration, to induce pedagogical policies that would minimize the students' time on task. Iglesias et al. applied another common online, off-policy approach named Q-learning to induce policies for efficient learning [9, 10]. More recently, Rafferty et al. applied POMDP with off-policy tree search to induce policies for faster learning [19]. Wang et al. applied an online, off-policy Deep-RL approach to induce a policy for adaptive narrative generation in educational game [36]. All of the models described above were evaluated via simulations or classroom studies, yielding improved student learning and/or behaviors as compared to baseline policies.

Offline RL approaches, on the other hand, "take advantage of previous collected samples, and generally provide robust convergence guarantees" [25]. The success of offline RL is thus often heavily dependent on the quality of the training data. One common convention is to collect an exploratory corpus by training students on an ITS that makes random yet *reasonable* decisions and then apply RL to induce pedagogical policies from that corpus. Shen et al. applied value iteration and least square policy iteration on a pre-collected training corpus to induce pedagogical policies aimed at improving students' learning performance [28, 27]. Chi et al. applied policy iteration to induce a pedagogical policy aimed at improving students' learning gains [4]. Mandel et al. [13] applied an offline POMDP approach to induce a policy which aims to improve student performance in an educational game. In classroom studies, most models above were found to yield certain improved student learning relative to a baseline policy.

Despite these successes, the necessity for accurate simulations (online) or large training corpora (offline) has limited the wide use of RL for policy induction. Additionally, prior research on both online RL and offline RL has not taken the granularity of decisions into account when applying RL techniques for the induction of pedagogical policies. In the remainder of the paper, we will refer to these approaches as flat RL to differentiate them from our new HRL approach.

It has been widely shown that HRL can be more effective and data-efficient than flat RL approaches [6, 22, 18, 37, 11]. HRL generally breaks down a large decision-making problem into a hierarchy of small sub-problems and induces a policy for each of them. Since the sub-problems are small, they usually require less data to find the optimal policies. For example, Cuayhuitl et al. induced navigation policies [6] at 3 levels: buildings, floors, and corridors, showing that HRL converged to an optimal policy in much fewer iterations. Peng et al. showed success using temporal HRL to induce locomotion control policies for path following and soccer dribbling while flat policies could not complete these tasks [18]. Although promising, the use of hierarchy requires additional information, such as the transitions and rewards at different levels of granularity, to induce a policy, and this may be hard to get from pre-collected data. Therefore, most existing HRL applications have been online, but here, we propose and apply an offline, off-policy HRL approach. To the best of our knowledge, this is the first attempt to apply HRL to induce pedagogical policies.

2.2 WE, PS and FWE

Prior research has investigated the effectiveness of WE, PS, FWE, and their various combinations [31, 16, 15, 14, 33, 21, 26, 17, 23]. When focusing on PS and WE, Mclaren et al. found no significant difference in learning performance between studying WE-PS pairs and doing PS-only, but the former spent significantly less time than the PS-only [16]. In a subsequent study, Mclaren et al. compared three conditions: WE-only, PS-only and WE-PS pairs [15]. Similarly, no significant differences were found among them in terms of learning gains, but the WE condition spent significantly less time than the other two; and no significant time on task difference was found between PS-only and WE-PS pairs.

Several studies were conducted comparing different combinations of WE, PS, and FWE. Renkl et al. compared WE-FWE-PS with WE-PS pairs, and the former significantly outperformed the latter on student learning performance while no significant difference was found between them on time on task [21]. Similarly, Najar et al. compared adaptive WE/FWE/PS with WE-PS pairs [17]. They found that the former significantly outperformed WE-PS pairs in terms of learning outcomes and the former also spent significantly less time on task than the latter. For adaptive WE/FWE/PS, they used expert rules to make decisions based on student learning states. Finally, Salden et al. compared three conditions: WE-FWE-PS, FWE, and PS-only [23]. Their results showed that FWE outperformed WE-FWE-PS, which in turn outperformed PS-only, and no significant time on task difference was found among the three conditions. Note that in their study, the order of WE, FWE, and PS were fixed in WE-FWE-PS; while in FWE, the tutor used an adaptive pedagogical policy, expert rules combined with data-driven student models. In short, previous studies have shown that alternating among WE, PS, and FWEs can be more effective than only alternating between WE and PS; however, it is not clear whether the former can be more effective than only using FWEs. On the other hand, prior research either used a fixed policy (WE-FWE-PS) or hand-coded expert rules combined with data-driven student models to make decisions. In this work, we applied an offline, off-policy HRL framework to derive a hierarchical pedagogical policy directly from empirical data. Its effectiveness is directly compared against another data-driven FWE policy induced by applying one of the state-of-the-art flat RL methods: Deep Q-Network.

3 Policy Induction

In this work, both our proposed HRL framework and DQN are offline, off-policy in that they induce policies from a historical dataset $\mathcal{D}$ collected by training students on the ITS that makes random yet reasonable decisions. RL focuses on inducing effective decision making policies for an agent with the goal of maximizing the agent's cumulative rewards. In many domains, RL is applied with immediate rewards. In an automatic call center system, for example, the agent can receive an immediate reward for every question it asks because the impact of each question can be assessed instantaneously [38]. Immediate rewards are generally more effective than delayed rewards for RL-based policy induction. This is because it is easier to assign appropriate credit or blame when the feedback is tied to a single decision. The more we delay the rewards or punishments,

the harder it becomes to assign credit or blame properly. The availability of immediate rewards is especially important for HRL approaches. On the other hand, the most appropriate reward to use in ITSs is student learning gains, which are typically unavailable until the entire training process is complete. This is due to the complex nature of the learning process which makes it difficult to assess students' learning moment by moment and more importantly, many instructional interventions that boost short-term performance may not be effective over the long-term. Therefore, we first proposed and applied a Gaussian Processes based (GP-based) approach to infer "immediate rewards" from the delayed rewards and then applied HRL and DQN to induce the corresponding hierarchical or step-level policies based on the inferred immediate rewards. In the following, we will briefly describe: 1) our proposed GP-based approach to infer immediate rewards, 2) our offline, off-policy GP-based HRL framework, and 3) DQN. We now present a few critical details of the process, but many have been omitted to save space.

3.1 GP-Based Approach for Immediate Reward Inference

Our historical dataset $\mathcal{D}$ consists of student-ITS interaction trajectories with different lengths. Each trajectory d can be viewed as: $s_1 \xrightarrow{a_1,r_1} s_2 \xrightarrow{a_2,r_2} \cdots s_n \xrightarrow{a_n,r_n}$. Here $s_i \xrightarrow{a_i,r_i} s_{i+1}$ indicated that at the i_{th} turn in d, the learning environment was in state s_i, agent executed action a_i and received reward r_i, and then the learning environment transferred into state s_{i+1}. Because our primary interest is to improve students' final learning, we used Normalized Learning Gain (NLG) as the reward because it measures students' gain *irrespective of their incoming competence*. $NLG = \frac{posttest - pretest}{\sqrt{1-pretest}}$ where *pretest* and *posttest* refer to the students' test scores before and after the ITS training respectively and 1 is the maximum score. Given that a student's NLG will not be available until the entire training is completed, only terminal states have non-zero rewards. Thus for a trajectory d, $r_1 \cdots$, r_{n-1} are all equal to 0, and only the final reward r_n is equal to the student's $NLG \times 100$, which is in the range of (-∞, 100].

To infer the immediate rewards from the final delayed reward for each trajectory, we applied Gaussian Processes (GP) to learn a distribution function f for the expected values and the standard deviations of all of the immediate rewards. More specifically, a prior probability is given to each possible function before observation. Then, higher probabilities are given to the functions where the sum of the generated immediate rewards is close to the observed delayed reward. In other words, the immediate rewards inside each trajectory were inferred by minimizing the mean square error (MMSE) of additive Gaussian distributions [8]. The immediate rewards were distributed inside each trajectory by assuming that they follow Gaussian distributions and that these rewards add up to the delayed reward for each trajectory. Following the Gaussian Process Regression [20, 1] and the shared mutual information existed in the feature representation, we can thus infer the immediate rewards from delayed rewards.

3.2 An Offline, Off-policy GP-based HRL for Policy Induction

Most HRL research is based upon an extension of Markov Decision Processes (MDPs) called Discrete Semi-Markov decision processes (SMDPs) and the central idea behind

the HRL approach is to transform the problem of inducing effective pedagogical policies into one of computing an optimal policy for choosing actions in SMDPs. An MDP describes a stochastic control process and formally corresponds to a 4-tuple: $< S,A,T,R >$. When inducing pedagogical policies, the states S are vector representations composed of relevant learning environment features such as the difficulty level of a problem, percentage of the correct entries a student entered so far and so. In this study, we have a total of 142 state features to describe the learning environment; the actions A are selected from $\{WE,FWE,PS\}$ for problem-level decisions and from{elicit, tell} for steps; the reward function R is calculated from the system's success measures: students NLG. Once the $\{S,A,R\}$ has been defined, the transition probabilities T are estimated from the training corpus, $\mathcal{D}$. Once a complete MDP is constructed, calculation of an optimal policy via policy iteration is straightforward.

SMDPs extend the existing MDP framework with the addition of a set of complex activities [2] or options [30], each of which can invoke other activities recursively, thus allowing for hierarchical policy functions. The *complex* activities are distinct from the primitive actions in that a complex activity may contain multiple *primitive* actions. In our applications, WE, PS and FWE are complex activities while elicit and tell are primitive actions. A complex activity consists of three elements: a policy π that maps states to each available option, a termination condition, and an initiation set. A solution to the SMDP mentioned above is an optimal policy (π^*), a mapping from state to complex activities or primitive actions, that maximizes the expected discounted cumulative reward for each state.

The complex activities in SMDPs can take a variable number of low-level activity (or actions) to execute across multiple time steps. This makes it necessary to extend the state-transition function to take into account the activity length. If an activity a takes t' time steps to be executed in state s, then the state transition probability function given s and a is defined by the joint distribution of the result state s' and the number of time steps t' when action a is performed in the state s: $P(s',t'|s,a)$. The expected reward function is also extended to accumulate over the waiting time in s given action a. More specifically, the Q-value function $Q(s,a)$ represents the expected discounted reward the agent will gain if it takes an action a in a state s and follows the policy to the end and for SMDP, the Bellman equation can be re-written as:

$$Q(s,a) = R(s,a) + \sum_{s',t'} \gamma^{t'} P(s',t'|s,a) \max_{a' \in A} Q(s',a') \tag{1}$$

$0 \leq \gamma \leq 1$ is a discount factor. If γ is less than 1, then it will discount rewards obtained later. For HRL, learning occurs at multiple levels. The global learning generates a policy for the top level decision and local learning generates a policy for each complex activity. This process retains the fundamental assumption of RL: that goals are defined by their association with reward, and thus that the objective is to discover actions that maximize the long-term cumulative reward. Local learning focuses not on learning the best policy for the overall task but the best policy for the corresponding complex activity.

In our offline off-policy HRL framework, both problem- and step-level policies were learned by recursively using the Gaussian Processes to estimate the Q-value

function in equation 1. Using an actor-critic policy iteration framework, we iteratively update the policy. This process continues until the Q-value function and the induced policy converged. We assume that the Q-value function follows a prior distribution and by combining the prior of Q-value function and the inferred immediate rewards, the Gaussian Process Regression can provide the posterior distribution of the Q-value function approximation in a tractable way. In this work, our training corpus contains a total number of 1118 students' interaction logs collected from a series of seven prior studies which followed the identical procedure and learning materials as the students in this study described below. To induce the hierarchical policy, we defined a problem-level semi-MDP for determining whether the next problem should be WE, PS or FWE and for each of the training problems, we defined a step-level semi-MDP for inducing a step-level policy to determine elicit vs. tell if a complex activity FWE is selected for that training problem.

3.3 DQN for Policy Induction

A Double DQN approach [34] with the prioritized experience replay technique [24] was applied to induce the DQN step-level policy. A multi-layer perceptron neural network was used to approximate the Q-function. The inputs to the neural network were the last 3 step observations of a student and the outputs were the Q values for each possible step level action (in our case, elicit and tell). The network consists of two 64-unit layers with the rectified linear unit (ReLU) activation function (except that the output layer has no activation function). As a convention for this algorithm, an experience replay buffer and a target network were used to stabilize the training. The data and immediate rewards used for DQN policy induction were identical to those used for HRL.

4 Empirical Experiment

Participants This study was conducted in the undergraduate Discrete Mathematics course at the Department of Computer Science at North Carolina State University in the Fall of 2018. The study was given as one of the regular homework assignments; students had one week to complete it and were graded based upon their demonstrated effort rather than performance. Students (N=180) were randomly assigned into three conditions (60 in each of HRL, DQN, and Random). Due to preparations for exams and the length of the experiment, 140 students completed the study. 3 students who scored perfectly in the pre-test were excluded from our subsequent analysis. In addition, 9 students who completed the study in groups were excluded. The remaining 128 students were distributed as follows: $N = 44$ for HRL, $N = 45$ for DQN, and $N = 39$ for Random. A χ^2 test shows that the participants' completion rate did not differ by condition: $\chi^2(2) = 1.03, p = 0.598$.

Pyrenees is a web-based ITS that teaches students a general problem solving strategy and 10 major principles of probability, such as the Complement Theorem and Bayes' Rule. It provides students with step-by-step instruction, immediate feedback,

and on-demand help. Specifically, the help is provided via a sequence of increasingly specific hints. The last hint in the sequence, i.e., the bottom-out hint, tells student exactly what to do. Except for the decision granularity, the remaining components of the tutor, including the GUI interface, the training problems, and the tutorial support were identical for all students.

Procedure All three conditions went through the same four phases: 1) textbook, 2) pre-test, 3) training on the ITS, and 4) post-test. The only difference among them was the policy employed by the ITS. During **textbook**, all students read a general description of each principle, reviewed some examples, and solved some training problems. The students then took a **pre-test** which contained a total of 14 single- and multiple-principle problems. Students were not given feedback on their answers, nor were they allowed to go back to earlier questions (this was also true for the post-test). During **training on the ITS**, all three conditions received the same 12 problems in the same order. Each domain principle was applied at least twice. Finally, all students took the 20-problem **post-test**; 14 of the problems were isomorphic to the pre-test. The remainder were non-isomorphic multiple-principle problems.

Grading criteria The pre- and post-test problems required students to derive an answer by writing and solving one or more equations. We used three scoring rubrics: binary, partial credit, and one-point-per-principle. Under the binary rubric, a solution was worth 1 point if it was completely correct or 0 if not. Under the partial credit rubric, each problem score was defined by the proportion of correct principle applications evident in the solution. A student who correctly applied 4 of 5 possible principles would get a score of 0.8. The One-point-per-principle rubric in turn gave a point for each correct principle application. All of the tests were graded in a double-blind manner by a single experienced grader. The results presented below were based upon the partial-credit rubric but the same results hold for the other two. For comparison purposes, all test scores were normalized to the range of [0,100].

5 Results

Despite of random assignment, a one-way ANOVA analysis on the pre-test score showed a marginally significant difference among the three conditions: $F(2,125) = 2.805$, $p = 0.064$, $\eta = 0.043$. Subsequent contrast analysis showed that DQN scored significantly higher than HRL: $t(125) = 2.06$, $p = 0.042$, $d = 0.46$ and Random: $t(125) = 2.01$, $p = 0.046$, $d = 0.46$; but there is no significant difference between HRL and Random: $t(125) = 0.02$, $p = 0.986$, $d = 0.00$. The results suggest that while our random assignment indeed balanced the HRL and Random conditions' incoming competence, it did not do so for the DQN condition. Therefore, we mainly focus on comparing learning performances that consider the pre-test differences, that is, adjusted post-test and NLG especially the latter because it is the reward we used for policy induction.

Table 1 shows the mean and standard deviation (SD) of students' learning performance and total training time results across three conditions. From left to right, it shows the condition with the number of students in parentheses, pre-test (Pre), isomorphic post-test (Iso Post), full post-test (Full Post), adjusted post-test (Adj Post), Normalized Learning Gain (NLG), and the total training time on the ITS in hours (Time).

Table 1. Learning Performance and Time on Task

Condition	Pre	Iso Post	Full Post	Adj Post	NLG	Time (hours)
HRL(44)	66.4(18.8)	85.8(14.6)	75.3(16.9)	77.7(10.3)	14.3(19.2)	2.19(.64)
DQN(45)	73.9(13.6)	85.2(13.1)	74.2(14.6)	71.2(12.0)	-2.2(29.4)	1.81(.58)
Random(39)	66.3(18.9)	80.5(19.5)	69.0(19.6)	71.4(13.8)	-0.1(35.0)	1.97(.52)

Isomorphic Post-test To measure students' learning improvement, we compared their isomorphic post-test scores with their pre-test scores. A repeated measures analysis using test type (pre-test vs. isomorphic post-test) as a factor and test score as the dependent measure showed a main effect for test type: $F(1,127)=158.63$, $p<0.0001$, $\eta=0.555$ in that students scored significantly higher in the isomorphic post-test than in the pre-test. More specifically, all three conditions scored significantly higher in the isomorphic post-test than in the pre-test: $F(1,43)=110.74$, $p<0.0001$, $\eta=0.720$ for HRL, $F(1,44)=34.73$, $p<0.0001$, $\eta=0.441$ for DQN, and $F(1,38)=38.47$, $p<0.0001$, $\eta=0.503$ for Random. This showed that the basic practice and problems, domain exposure, and interactivity of our ITS effectively help students acquire knowledge, even when the decisions are made randomly yet reasonably.

Adjusted Post-test To comprehensively evaluate students' final performance, we performed analysis on the full post-test score which has an additional six multiple-principle problems. An ANCOVA analysis on the post-test using the pre-test score as a covariate showed a significant difference among the three conditions: $F(2,124)=3.86$, $p=0.024$, $\eta=0.030$. Subsequent contrast analysis on the adjusted post-test score showed that the HRL condition scored significantly higher than the DQN condition: $t(125)=2.53$, $p=0.013$, $d=0.57$ and the Random condition: $t(125)=2.36$, $p=0.020$, $d=0.52$. No significant difference was found between DQN and Random. The results suggest that the HRL policy is significantly more effective than the DQN policy and the Random policy.

NLG Similarly, a one-way ANOVA analysis on the NLG showed that there is a significant difference among the three conditions: $F(2,125)=4.39$, $p=0.014$, $\eta=0.066$. Subsequent contrast analysis showed that the HRL condition scored significantly higher than the DQN condition: $t(125)=2.75$, $p=0.007$, $d=0.66$ and the Random condition: $t(125)=2.30$, $p=0.023$, $d=0.52$. Again, no significant difference was found between DQN and Random. The results suggest again that the HRL policy significantly outperformed the DQN policy and the Random policy.

Time on Task A one-way ANOVA analysis on time on task showed a significant difference among the three conditions: $F(2,125)=4.74$, $p=0.010$, $\eta=0.071$. More specifically, the HRL condition spent significantly more time than the DQN condition: $t(125)=3.07$, $p=0.003$, $d=0.62$ and marginally significantly more time than the Random condition: $t(125)=-1.75$, $p=0.082$, $d=0.39$.

Tutor Decisions Our preliminary log analysis revealed that for the HRL condition, the average number of problem-level decisions students received are: .95(1.16) for WE, 5.07(2.58) for PS and 3.98(2.49) for FWE. Thus the HRL policy was more likely to choose PS and FWE than WE. Table 2 shows the number of step-level decisions

Table 2. Step Level Tutor Decisions

Condition	Elicit	Tell	Pct Tell
HRL	309.0(60.4)	88.7(66.1)	22.025(15.870)
DQN	205.8(51.6)	188.9(53.0)	47.794(12.974)
Random	200.5(15.9)	203.5(17.4)	50.354(2.482)

students received across the three conditions. The first column shows the condition followed by the number of elicit and tell and finally the percentage of tell. Our preliminary step-level log analysis results showed that the HRL condition received more elicit than tell; while the other two conditions received a relatively balanced amount. A one-way ANOVA analysis on the percentage of tell revealed a significant difference among the three conditions: $F(2, 125) = 71.47,\ p < 0.0001,\ \eta = 0.533$. Subsequent contrast analysis showed that the HRL condition received significantly less tell than the DQN condition: $t(125) = -10.00,\ p < 0.0001,\ d = 1.78$ and the Random condition: $t(125) = -10.60,\ p < 0.0001,\ d = 2.42$. In addition, the HRL and the DQN condition had a much higher SD on tell percentage. This suggests that the HRL policy and the DQN policy made more personalized decisions than the Random policy.

6 Conclusion and Discussion

In this study, we proposed and applied an offline, off-policy GP-based HRL framework to induce a hierarchical pedagogical policy. The policy makes decisions first at the problem level and then the step level. At the problem level, it decides whether the next problem should be WE, PS or FWE. If FWE is selected, a corresponding step-level policy will be activated to decide whether the next step should be elicit or tell. In an empirical classroom study, we compared the HRL policy with a DQN induced step-level policy and a Random step-level policy. Our results showed that the HRL policy was significantly more effective than the DQN policy and the Random policy and no significant difference was found between the latter two policies. For time on task, there was no significant difference between the HRL condition and the Random condition, but the former spent significant more time than the DQN condition. Finally, the HRL policy was more likely to choose PS and FWE than WE.

The results suggest that HRL can be more effective than flat RL in pedagogical policy induction. One possible explanation is that HRL has an explicit problem-level vision. At the problem level, HRL views a problem as an atomic action, and this abstraction has two potential advantages: 1) it aggregates the effects of all steps in a problem and 2) it converts a long step-level sequence into a short problem-level sequence. The aggregation of steps across a problem may provide HRL with a better estimation of the effect of taking a series of steps; while the problem sequence may give HRL a better view of the long-term effects of each problem. Theoretically, flat RL could learn the impact of a problem by aggregating step-level information, but there is no guarantee that it would. Our results confirm the intuition that HRL should outperform flat RL on pedagogical policy induction because it can simultaneously learn at two levels of granularity - the problem level outer loop and the step level inner loop.

References

1. Azizsoltani, H., Sadeghi, E.: Adaptive sequential strategy for risk estimation of engineering systems using gaussian process regression active learning. Engineering Applications of Artificial Intelligence **74**(July), 146–165 (2018)
2. Barto, A.G., Mahadevan, S.: Recent advances in hierarchical reinforcement learning. Discrete event dynamic systems **13**(1-2), 41–77 (2003)
3. Beck, J., Woolf, B.P., Beal, C.R.: Advisor: A machine learning architecture for intelligent tutor construction. AAAI/IAAI **2000**(552-557), 1–2 (2000)
4. Chi, M., VanLehn, K., Litman, D., Jordan, P.: Empirically evaluating the application of reinforcement learning to the induction of effective and adaptive pedagogical strategies. User Modeling and User-Adapted Interaction **21**(1-2), 137–180 (2011)
5. Clement, B., Oudeyer, P.Y., Lopes, M.: A comparison of automatic teaching strategies for heterogeneous student populations. In: EDM 16-9th International Conference on Educational Data Mining (2016)
6. Cuayáhuitl, H., Dethlefs, N., Frommberger, L., Richter, K.F., Bateman, J.: Generating adaptive route instructions using hierarchical reinforcement learning. In: International Conference on Spatial Cognition. pp. 319–334. Springer (2010)
7. Evens, M., Michael, J.: One-on-one tutoring by humans and computers. Psychology Press (2006)
8. Guo, D., Shamai, S., Verdú, S.: Mutual information and minimum mean-square error in gaussian channels. IEEE Transactions on Information Theory **51**(4), 1261–1282 (2005)
9. Iglesias, A., Martínez, P., Aler, R., Fernández, F.: Learning teaching strategies in an adaptive and intelligent educational system through reinforcement learning. Applied Intelligence **31**(1), 89–106 (2009)
10. Iglesias, A., Martínez, P., Aler, R., Fernández, F.: Reinforcement learning of pedagogical policies in adaptive and intelligent educational systems. Knowledge-Based Systems **22**(4), 266–270 (2009)
11. Kulkarni, T.D., Narasimhan, K., Saeedi, A., Tenenbaum, J.: Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In: Advances in neural information processing systems. pp. 3675–3683 (2016)
12. Lajoie, S.P., Derry, S.J.: Motivational techniques of expert human tutors: Lessons for the design of computer-based tutors. In: Computers as cognitive tools, pp. 83–114. Routledge (2013)
13. Mandel, T., Liu, Y.E., Levine, S., Brunskill, E., Popovic, Z.: Offline policy evaluation across representations with applications to educational games. In: Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems. pp. 1077–1084. International Foundation for Autonomous Agents and Multiagent Systems (2014)
14. McLaren, B.M., van Gog, T., Ganoe, C., Yaron, D., Karabinos, M.: Exploring the assistance dilemma: Comparing instructional support in examples and problems. In: Intelligent Tutoring Systems. pp. 354–361. Springer (2014)
15. McLaren, B.M., Isotani, S.: When is it best to learn with all worked examples? In: AIED. pp. 222–229. Springer (2011)
16. McLaren, B.M., Lim, S.J., Koedinger, K.R.: When and how often should worked examples be given to students? new results and a summary of the current state of research. In: CogSci. pp. 2176–2181 (2008)
17. Najar, A.S., Mitrovic, A., McLaren, B.M.: Adaptive support versus alternating worked examples and tutored problems: Which leads to better learning? In: UMAP, pp. 171–182. Springer (2014)

18. Peng, X.B., Berseth, G., Yin, K., Van De Panne, M.: Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. ACM Transactions on Graphics (TOG) **36**(4), 41 (2017)
19. Rafferty, A.N., Brunskill, E., Griffiths, T.L., Shafto, P.: Faster teaching via pomdp planning. Cognitive science **40**(6), 1290–1332 (2016)
20. Rasmussen, C.E.: Gaussian processes in machine learning. In: Advanced lectures on machine learning, pp. 63–71. Springer (2004)
21. Renkl, A., Atkinson, R.K., Maier, U.H., Staley, R.: From example study to problem solving: Smooth transitions help learning. The Journal of Experimental Education **70**(4), 293–315 (2002)
22. Ryan, M., Reid, M.: Learning to fly: An application of hierarchical reinforcement learning. In: In Proceedings of the 17th International Conference on Machine Learning. Citeseer (2000)
23. Salden, R.J., Aleven, V., Schwonke, R., Renkl, A.: The expertise reversal effect and worked examples in tutored problem solving. Instructional Science **38**(3), 289–307 (2010)
24. Schaul, T., Quan, J., Antonoglou, I., Silver, D.: Prioritized experience replay. arXiv preprint arXiv:1511.05952 (2015)
25. Schwab, D., Ray, S.: Offline reinforcement learning with task hierarchies. Machine Learning **106**(9-10), 1569–1598 (2017)
26. Schwonke, R., Renkl, A., Krieg, C., Wittwer, J., Aleven, V., Salden, R.: The worked-example effect: Not an artefact of lousy control conditions. Computers in Human Behavior **25**(2), 258–266 (2009)
27. Shen, S., Ausin, M.S., Mostafavi, B., Chi, M.: Improving learning & reducing time: A constrained action-based reinforcement learning approach. In: Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization. pp. 43–51. ACM (2018)
28. Shen, S., Chi, M.: Reinforcement learning: the sooner the better, or the later the better? In: Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization. pp. 37–44. ACM (2016)
29. Stamper, J.C., Eagle, M., Barnes, T., Croy, M.: Experimental evaluation of automatic hint generation for a logic tutor. In: International Conference on Artificial Intelligence in Education. pp. 345–352. Springer (2011)
30. Sutton, R.S., Precup, D., Singh, S.: Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. Artificial intelligence **112**(1-2), 181–211 (1999)
31. Sweller, J., Cooper, G.A.: The use of worked examples as a substitute for problem solving in learning algebra. Cognition and Instruction **2**(1), 59–89 (1985)
32. Thomas, P., Brunskill, E.: Data-efficient off-policy policy evaluation for reinforcement learning. In: International Conference on Machine Learning. pp. 2139–2148 (2016)
33. Van Gog, T., Kester, L., Paas, F.: Effects of worked examples, example-problem, and problem-example pairs on novices learning. Contemporary Educational Psychology **36**(3), 212–218 (2011)
34. Van Hasselt, H., Guez, A., Silver, D.: Deep reinforcement learning with double q-learning. In: AAAI. vol. 2, p. 5. Phoenix, AZ (2016)
35. Vanlehn, K.: The behavior of tutoring systems. IJAIED **16**(3), 227–265 (2006)
36. Wang, P., Rowe, J., Min, W., Mott, B., Lester, J.: Interactive narrative personalization with deep reinforcement learning. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (2017)
37. Wang, X., Chen, W., Wu, J., Wang, Y.F., Yang Wang, W.: Video captioning via hierarchical reinforcement learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4213–4222 (2018)

38. Williams, J.D.: The best of both worlds: unifying conventional dialog systems and pomdps. In: INTERSPEECH. pp. 1173–1176 (2008)
39. Zhou, G., Wang, J., Lynch, C., Chi, M.: Towards closing the loop: Bridging machine-induced pedagogical policies to learning theories. In: EDM (2017)