Hierarchical Reinforcement Learning for Pedagogical Policy Induction (Extended Abstract)*

Guojing Zhou , Hamoon Azizsoltani , Markel Sanz Ausin , Tiffany Barnes and Min Chi
North Carolina State University
{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. In recent years, there is growing interest in applying data-driven techniques for adaptive decision making that 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 Gaussian Processes based Hierarchical Reinforcement Learning (HRL) framework to induce a hierarchical pedagogical policy that makes decisions at both problem and step levels. An empirical classroom study shows that the HRL policy is significantly more effective than a Deep Q-Network (DQN) induced policy and a random *yet reasonable* baseline policy.

1 Introduction

E-learning environments, such as intelligent tutoring systems (ITSs) has enabled us to provide low-cost and time-flexible education opportunities. It has been shown that such environments can improve students' learning in real classrooms [Koedinger *et al.*, 1997]. During training, the tutor makes decisions on what students should do next based on observations of students' learning state. The tutor's decisions can be viewed as a temporal sequence, each of which affects the student's successive actions and performance. Its impact on student learning cannot often be observed immediately and the effectiveness of one decision also depends on subsequent decisions. Most existing e-learning applications employ *Pedagogical policies* to decide what action to take next in the face of alternatives. Ideally, an effective learning environment should craft and adapt its decisions to users' needs [Anderson *et al.*, 1995]. In recent years, a number of researchers have studied applying Reinforcement Learning (RL) to induce pedagogical policies directly from student-tutor interaction logs (e.g. [Shen *et al.*, 2018; Chi *et al.*, 2011; Mandel *et al.*, 2014]). Despite promising, most prior applications treat all system actions *equally, or independently*, and do not account for the long-term impact that tutor decisions may have across these two levels of granularity.

In ITSs, there are decisions to make at different levels of granularity, from whole problems to single steps. At each of these levels, the system may decide to give a hint, provide a worked example, or give immediate feedback. These actions all have the goal of helping students solve a problem, but some may be more important or impactful than others. Human decision-makers treat these distinct levels of granularity differently and are capable of selecting among them [Evens and Michael, 2006; Lepper *et al.*, 1993]. In this work, we propose and apply an offline hierarchical reinforcement learning (HRL) framework to induce a pedagogical policy that makes decisions at two levels of granularity: *problem* and *step*.

More specifically, the tutor first decides whether the next *problem* should be a Worked Example (WE), Problem Solving (PS), or Collaborative Problem Solving (CPS). In WE, the student observes how the tutor solves a problem; in PS, the student solves the problem themselves; in CPS, the student and the tutor *co-construct* the solution. Based on the problem-level decision, the tutor then makes step-level decisions on whether to elicit the next step from the student or to show it directly. We refer to such decisions as *elicit/tell*. 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 CPS is selected, the tutor will decide whether to elicit or tell a step based on the corresponding step-level policy. A great deal of research has investigated the impacts of the problem-level WE vs. PS and the step-level elicit vs. tell [Renkl *et al.*, 2002; Schwonke *et al.*, 2009; Salden *et al.*, 2010]. However, none of them are well understood, and there is no widespread consensus on how they should be used. Thus, here, we apply data-driven approaches to induce pedagogical policies directly from data. Results from an empirical classroom study showed that the HRL policy was significantly more effective than a DQN induced step-level policy and a random yet reasonable step-level policy. Since both elicit and tell are always considered to be reasonable educational interventions in our learning context, our random policy is random yet reasonable.

*This is an abridged version of the AIED 2019 best paper "*Hierarchical Reinforcement Learning for Pedagogical Policy Induction*" [Zhou *et al.* 2019]

2 Background

In recent years, RL, especially Deep RL has been applied to handle complicated tasks and has achieved superhuman performance in several complex games. However, different from the classic game-play situations where the ultimate goal is to make smart system decisions, our ultimate goal for RL is to make the student-tutor interaction productive and fruitful. This raises additional challenges due to the high cost to collect and the complexity of human learning data.

Generally speaking, RL approaches can be categorized into online and offline. Online approaches learn a policy in real-time by interacting with the environment while offline approaches learn from pre-collected training data. 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. Offline RL approaches, on the other hand, "take advantage of previously collected samples, and generally provide robust convergence guarantees" [Schwab and Ray, 2017]. 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. Researchers have applied a variety of online [Beck *et al.*, 2000; Iglesias *et al.*, 2009; Rafferty *et al.*, 2016] and offline [Shen *et al.*, 2018; Chi *et al.*, 2011; Mandel *et al.*, 2014] RL approaches for pedagogical policy induction. All the models described here were evaluated in classroom studies, yielding improved student learning and/or behaviors as compared to baseline policies.

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 applying both online and offline RL for pedagogical policy induction has not taken the granularity of decisions into account. 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 [Cuayáhuitl *et al.*, 2010; Peng *et al.*, 2017; Wang *et al.*, 2018; Kulkarni *et al.*, 2016]. 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 fewer data to find the optimal policies. For example, Cuayáhuitl et al. induced navigation policies [Cuayáhuitl *et al.*, 2010] at 3 levels: buildings, floors, and corridors, showing that HRL converged to an optimal policy in much fewer iterations. Although promising, the use of hierarchy requires additional information, such as the transitions and rewards at different levels of granularity, to induce a policy. A simple and effective way to collect such information is to explore the environment during learning. Therefore, most existing HRL applications have been online. But here, we propose and apply an offline HRL approach that can induce policies from pre-collected data.

3 Policy Induction

In RL policy induction, immediate rewards are generally more effective than delayed rewards. This is because it is easier to assign appropriate credit or blame when the feedback is tied to a single decision. For HRL policy induction, immediate rewards are needed at different levels, to induce hierarchical policies. 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, in this work, we first propose and apply a Gaussian Processes based (GP-based) approach to infer "immediate rewards" from the delayed rewards and then apply 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 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 $s_i \xrightarrow{a_i, r_i} s_{i+1}$ indicates that at the i_{th} turn in d, the learning environment was in state s_i, the agent executed action a_i and received reward r_i, and then the learning environment transferred into state s_{i+1}. Since our primary interest is to improve students' final learning gain, 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 complete, only terminal states have non-zero rewards. That is for a trajectory d, $r_1 \cdots, r_{n-1}$ are all equal to 0, only $r_n = NLG \times 100$, which is in the range of (-∞, 100].

To infer the immediate rewards, we applied a Gaussian Processes (GP) approach to learn a function that specifies the reward for each state-action pair in the trajectory [Azizsoltani *et al.*, 2019]. This approach distributes immediate rewards inside each trajectory by assuming that they follow Gaussian distributions and that these rewards add up to the delayed reward. Different from standard GP which minimizes the direct output error, this approach minimizes the additive error of the model output. The process starts with assigning a prior probability to each possible function. Then following the Gaussian Process Regression [Rasmussen, 2004; Azizsoltani and Sadeghi, 2018] and using the shared mutual information existed in the feature representation, higher probabilities are given to the functions where the sum of the generated immediate rewards is close to the observed delayed reward. Averaging all the possible functions generates the reward function.

3.2 An Offline 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). An MDP describes a stochastic control process that can be described as a 4-tuple: $< S, A, T, R >$. In pedagogical policy induction, 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 has entered so far and so. In this study, the learning environment is described by 142 features; the actions A are selected from $\{$WE, PS, CPS$\}$ 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 $\mathcal{D}$.

SMDPs extend the existing MDP framework by adding a set of complex activities [Barto and Mahadevan, 2003] or options [Sutton *et al.*, 1999], each of which can invoke other activities recursively, thus allowing the hierarchical policy to function. 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 CPS 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.

Since complex activities can take a variety of time steps to execute, it is necessary to extend the state-transition function to take into account the activity length. The extended transition probability function can be denoted as: $P(s', t'|s, a)$, which specifies the probability of transitioning to state s' after t' time steps if taking action a in state s. Accordingly, the expected reward function is also extended to accumulate over the waiting time t' in s given activity a. Similar to RL, HRL learns the policy through estimating the Q-value function $Q(s, a)$, denoted as the expected cumulative rewards the agent will receive if it takes activity a in state s and follows the policy to the end. The optimal Q-value function Q^* denotes the expected cumulative rewards the agent can receive if it follows the optimal policy and Q^* satisfies the Bellman equation [Sutton *et al.*, 1999]. In SMDPs, the Bellman equation can be rewritten as:

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

where $0 \leq \gamma \leq 1$ is a discount factor. For HRL, learning occurs at multiple levels. The global learning generates a policy for the top-level decisions 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 HRL framework, both problem- and step-level policies were learned by recursively using the Gaussian Processes (GP) to estimate the Q-value function [Rasmussen, 2004] following Equation 1 until the Q-value function and the policy converged. In each iteration, a Q-value was generated for each state-action pair in the training trajectories following Equation 1 based on the inferred immediate rewards and the latest GP model. Then the GP model was updated based on the new Q-value assigned to each state-action pair. Our training corpus contains 1,118 students' interaction logs. To induce the hierarchical policy, we defined a problem-level semi-MDP for determining whether the next problem should be WE, PS or CPS 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 CPS is selected for that training problem.

3.3 DQN for Policy Induction

A Double DQN approach [Van Hasselt *et al.*, 2016] with the prioritized experience replay technique [Schaul *et al.*, 2015] was applied to induce the DQN step-level policy. A multilayer 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 an undergraduate Discrete Mathematics course in Fall 2018 as a regular homework assignment. 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 final exam preparations, 140 students completed the study. 3 students who scored perfectly in the pre-test and 9 students who completed the study in groups were excluded from our subsequent analysis. 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 significantly by condition: $\chi^2(2) = 1.03, p = 0.598$.

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

Condition	Pre	Iso Post	Adj Post	NLG
HRL(44)	66.4(18.8)	85.8(14.6)	77.7(10.3)	14.3(19.2)
DQN(45)	73.9(13.6)	85.2(13.1)	71.2(12.0)	-2.2(29.4)
Random(39)	66.3(18.9)	80.5(19.5)	71.4(13.8)	-0.1(35.0)

Table 1: Learning Performance

the pedagogical policy, 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 four phases: 1) textbook, 2) pre-test, 3) training on the ITS, and 4) post-test. The only difference between 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 practice problems. Then they took a 14-problem **pre-test** where no feedback was given. During **training on the ITS**, all students received the same 12 problems in the same order. Finally, they took a 20-problem **post-test**. 14 of the problems were isomorphic to the pre-test and the remainders were non-isomorphic complicated problems.

Grading. The pre- and post-tests were graded using a partial credit rubric, where each problem score was determined by the percentage of principles correctly applied. For example, a student who correctly applied 4 of 5 possible principles would get a score of 0.8. Experienced graders conducted the grading in a double-blind manner. 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 was no significant difference between HRL and Random: $t(125) = 0.02$, $p = 0.986$, $d = 0.00$. Since students were not perfectly balanced in incoming competence, we took students' pre-test score into account in all following analyses. Table 1 shows the mean and standard deviation (SD) of students' learning performance results, showing (from left to right) the Condition with the number of students in parentheses, pre-test (Pre), isomorphic post-test (Iso Post), adjusted post-test (Adj Post) and Normalized Learning Gain (NLG).

To measure students' learning improvement, we compared their isomorphic post-test scores with their pre-test scores. A repeated measures analysis showed that all students scored significantly higher in the isomorphic post-test than in the pre-test: $F(1, 127) = 158.63$, $p < 0.0001$, $\eta = 0.555$. This is also true for each individual condition including the random baseline. This suggests 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 how the HRL and DQN policies impacted students learning performance, we conducted analysis based on the full post-test score. Comparing to the pre-test, the full post-test contains six additional complicated problems. An ANCOVA analysis on the full post-test using the pre-test score as a covariate showed a significant difference among the three conditions: $F(2, 124) = 3.86$, $p = 0.024$, $\eta = 0.030$. Then, based on the linear model generated in the ANCOVA analysis and the pre-test score, we calculated the adjusted post-test score (Adj Post). Contrast analysis on the adjusted post-test 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 (calculated based on pre- and full post-test) showed that there was a significant difference among the three conditions: $F(2, 125) = 4.39$, $p = 0.014$, $\eta = 0.066$. Subsequent contrast analysis showed that HRL scored significantly higher than DQN: $t(125) = 2.75$, $p = 0.007$, $d = 0.66$ and Random: $t(125) = 2.30$, $p = 0.023$, $d = 0.52$. Again, no significant difference was found between DQN and Random. Note that NLG is the reward we used for policy induction and thus, the results suggest that our HRL policy indeed can improve the desired outcome measure.

6 Conclusion and Discussion

In this study, we proposed and applied an offline GP-based HRL framework to induce a hierarchical pedagogical policy, which makes decisions first at the problem level and then the step level. Results from an empirical classroom study showed that the HRL policy was significantly more effective than a DQN induced step-level policy and a Random step-level policy. 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.

Acknowledgments

This research was supported by the NSF Grants: #1726550, #1651909, and #1660878.

References

[Anderson *et al.*, 1995] John R Anderson, Albert T Corbett, Kenneth R Koedinger, and Ray Pelletier. Cognitive tutors: Lessons learned. *The journal of the learning sciences*, 4(2):167–207, 1995.

[Azizsoltani and Sadeghi, 2018] Hamoon Azizsoltani and Elham Sadeghi. 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.

[Azizsoltani *et al.*, 2019] Hamoon Azizsoltani, Yeo Jin Kim, Markel Sanz Ausin, Tiffany Barnes, and Min Chi. Unobserved is not equal to non-existent: using gaussian processes to infer immediate rewards across contexts. In *IJCAI 2019*, pages 1974–1980. AAAI Press, 2019.

[Barto and Mahadevan, 2003] Andrew G Barto and Sridhar Mahadevan. Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems*, 13(1-2):41–77, 2003.

[Beck *et al.*, 2000] Joseph Beck, Beverly Park Woolf, and Carole R Beal. Advisor: A machine learning architecture for intelligent tutor construction. *AAAI/IAAI*, 2000(552-557):1–2, 2000.

[Chi *et al.*, 2011] Min Chi, Kurt VanLehn, Diane Litman, and Pamela Jordan. 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.

[Cuayáhuitl *et al.*, 2010] Heriberto Cuayáhuitl, Nina Dethlefs, Lutz Frommberger, Kai-Florian Richter, and John Bateman. Generating adaptive route instructions using hierarchical reinforcement learning. In *ICSC*, pages 319–334. Springer, 2010.

[Evens and Michael, 2006] Martha Evens and Joel Michael. *One-on-one tutoring by humans and computers*. Psychology Press, 2006.

[Iglesias *et al.*, 2009] Ana Iglesias, Paloma Martínez, Ricardo Aler, and Fernando Fernández. Reinforcement learning of pedagogical policies in adaptive and intelligent educational systems. *Knowledge-Based Systems*, 22(4):266–270, 2009.

[Koedinger *et al.*, 1997] Kenneth R Koedinger, John R Anderson, William H Hadley, and Mary A Mark. Intelligent tutoring goes to school in the big city. *I. J. Artificial Intelligence in Education*, 1997.

[Kulkarni *et al.*, 2016] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *NIPS*, pages 3675–3683, 2016.

[Lepper *et al.*, 1993] Mark R Lepper, Maria Woolverton, Donna L Mumme, and J Gurtner. Motivational techniques of expert human tutors: Lessons for the design of computer-based tutors. *Computers as cognitive tools*, 1993:75–105, 1993.

[Mandel *et al.*, 2014] Travis Mandel, Yun-En Liu, Sergey Levine, Emma Brunskill, and Zoran Popovic. Offline policy evaluation across representations with applications to educational games. In *AAMAS*, pages 1077–1084, 2014.

[Peng *et al.*, 2017] Xue Bin Peng, Glen Berseth, KangKang Yin, and Michiel Van De Panne. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Transactions on Graphics*, 36(4):41, 2017.

[Rafferty *et al.*, 2016] Anna N Rafferty, Emma Brunskill, Thomas L Griffiths, and Patrick Shafto. Faster teaching via pomdp planning. *Cognitive science*, 40(6):1290–1332, 2016.

[Rasmussen, 2004] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Advanced lectures on machine learning*, pages 63–71. Springer, 2004.

[Renkl *et al.*, 2002] Alexander Renkl, Robert K Atkinson, Uwe H Maier, and Richard Staley. From example study to problem solving: Smooth transitions help learning. *The Journal of Experimental Education*, 70(4):293–315, 2002.

[Salden *et al.*, 2010] Ron JCM Salden, Vincent Aleven, Rolf Schwonke, and Alexander Renkl. The expertise reversal effect and worked examples in tutored problem solving. *Instructional Science*, 38(3):289–307, 2010.

[Schaul *et al.*, 2015] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

[Schwab and Ray, 2017] Devin Schwab and Soumya Ray. Offline reinforcement learning with task hierarchies. *Machine Learning*, 106(9-10):1569–1598, 2017.

[Schwonke *et al.*, 2009] Rolf Schwonke, Alexander Renkl, Carmen Krieg, Jörg Wittwer, Vincent Aleven, and Ron Salden. The worked-example effect: Not an artefact of lousy control conditions. *Computers in Human Behavior*, 25(2):258–266, 2009.

[Shen *et al.*, 2018] Shitian Shen, Markel Sanz Ausin, Behrooz Mostafavi, and Min Chi. Improving learning & reducing time: A constrained action-based reinforcement learning approach. In *UMAP*, pages 43–51. ACM, 2018.

[Sutton *et al.*, 1999] Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

[Van Hasselt *et al.*, 2016] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *AAAI*, volume 2, page 5. Phoenix, AZ, 2016.

[Wang *et al.*, 2018] Xin Wang, Wenhu Chen, Jiawei Wu, Yuan-Fang Wang, and William Yang Wang. Video captioning via hierarchical reinforcement learning. In *CVPR*, pages 4213–4222, 2018.

[Zhou *et al.*, 2019] Guojing Zhou, Hamoon Azizsoltani, Markel Sanz Ausin, Tiffany Barnes, and Min Chi. Hierarchical reinforcement learning for pedagogical policy induction. In *International conference on artificial intelligence in education*, pages 544–556. Springer, 2019.