ABSTRACT

JU, SONG. A Critical Reinforcement Learning Framework for Human-Machine Mixed-Initiative Decisionmaking. (Under the direction of Dr. Min Chi.)

People make many decisions every day, some are minor and others can be life-altering. While making decisions often gives people a sense of control which enhances the expectation of performing behavior for the desired outcome, people may not be always good at making decisions due to the lack of self-control or decision fatigue. Reinforcement Learning (RL) is one of the most effective machine learning methods for decision-making under uncertainty. Much of the prior RL work focuses on *agent-centric tasks* on which the goal is for the agent to make effective decisions. On *human-centric tasks* where human is the subject to interact with the environment, however, applying RL is often challenging because the goal is for the RL agent to promote the effectiveness of *human decision making*. To balance the trade-off between giving the human a feeling of control over their progress and keeping their decisions effective, the key problem is to determine *when* the agent should intervene. This raises our primary research question: *Given a long trajectory of decisions, how can we identify the critical decisions that lead to the desired outcomes?*

In this thesis, we have explored three approaches to identify critical decisions in the fields of education and healthcare. In education, we leverage an Intelligent Tutoring System (ITS) that teaches student probability while in healthcare, we utilize two Electronic Health Records (EHR) datasets regarding septic treatment. ***First***, we apply *offline off-policy evaluation (OPE)* on an ITS historical dataset to determine the existence of critical decisions. Our results show that certain decisions are indeed highly correlated with students' post-test scores while others are less so. ***Second***, we propose a novel Adversarial Deep RL (ADRL) framework to identify critical decisions and induce a Critical policy that makes optimal actions in the identified critical states while random actions in others. Its effectiveness is empirically compared against a baseline policy. Our results show that the Critical policy can lead to higher learning performance than the baseline policy but only for students who experienced more critical states. ***Third***, we propose a novel Long-Short Term Rewards (LSTR) RL framework and evaluate its effectiveness on three testbeds: an ideal GridWorld Game, an ITS tutor, and a healthcare dataset. More specifically, a classroom study is conducted on the ITS tutor to evaluate the LSTR framework empirically. Our results show that for the RL policy to be effective, it must carry out the optimal actions in the LSTR-identified critical states, and more importantly, such critical RL policy can be as effective as a fully executed policy. For healthcare, our LSTR framework shows that it can effectively identify the critical medical interventions in septic treatment. More importantly, when combining our proposed LSTR framework with the expert decision making, our results show that for one healthcare system, our framework needs to nudge the physicians 45% of the time while for the other healthcare system, it only needs to nudge 20%.

A Critical Reinforcement Learning Framework for Human-Machine Mixed-Initiative Decisionmaking

by
Song Ju

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Computer Science

Raleigh, North Carolina

2021

APPROVED BY:

Tiffany Barnes	Noboru Matsuda
Maria Mayorga	Dr. Min Chi Chair of Advisory Committee

BIOGRAPHY

Song Ju was born in Jinan, China. He obtained his bachelor's and master's degree in Computer Science from Beijing Jiaotong University in 2010 and 2012, respectively. The undergraduate study helped him build a good foundation in basic knowledge of computer science, and the graduate study provided him a deep understanding of computer science and scientific research. During the period of the master's program, he was interested in the Location-Based Service and believed that computer science should be combined with other disciplines for sustainability and future development. Thus, he started working at the Institute of Remote Sensing and Digital Earth (RADI), Chinese Academy of Sciences (CAS) after graduation from Beijing Jiaotong University.

As a research assistant in RADI, his main responsibility was to develop Geographic information system (GIS) and research remote sensing image classification algorithms. He took part in several projects as a developer and also led as the project manager in a project named "ZY-3" from beginning to end, which including bidding, requirements analysis, high-level design, detailed design, and acceptance review. Although the work in RADI provided Song valuable experiences and an opportunity to pursue a Ph.D. degree in Remote Sensing, he still felt his inner interest is Computer Science and the strong desire to study abroad to explore the world and broaden his horizons.

In 2015, Song left RADI and started his Ph.D. study in Computer Science at North Carolina State University (NCSU). With an interest in machine learning, Song joined Dr. Min Chi at the Center for Educational Informatics (CEI). Most of Song's work was focused on the field of education, especially on inducing effective pedagogical policies for intelligent tutoring systems (ITSs). He was the first person to apply the Reinforcement Learning (RL) based approach to identify critical decisions in student learning and also generalized it to a healthcare task: sepsis treatment. Besides his academic school life, Song has also obtained summer internships for companies such as Hirtual, InsightFinder and Facebook, which gave him valuable industrial experience and the opportunity to apply his knowledge to solve real-world problems.

ACKNOWLEDGEMENTS

I would like to gratefully acknowledge everyone who played a role in my Ph.D. journey. First and foremost, I would like to give my sincere thanks to my advisor, Dr. Min Chi, for her constant guidance, patience as well as for providing the necessary direction. When I started my Ph.D. studies, I thought doing research would be smooth sailing with hard work. But the academic path is much harder due to the lack of experience and often no clue of where to start. With extensive knowledge and a keen sense of research in education, Dr. Chi could always give me insightful advice to help me overcome obstacles one after another. Furthermore, she is warm, caring, and always supporting her students.

I would like to thank my committee members, Dr. Tiffany Barnes, Dr. Noboru Matsuda, and Dr. Maria Mayorga. They gave me effective and constructive feedback on my dissertation, which inspired me a deeper thought of my work and made my research process more smooth.

I would like to thank my great lab mates Yuan Zhang, Guojing Zhou, Shitian Shen, Ye Mao, Xi Yang, Chen Lin, Yeojim Kim, Markel Sanz Ausin, and Farzaneh Khoshnevisan in CEI. I'm grateful to work with all of you and appreciate the supports and encouragement among us. I would also like to thank my friends Yiqiao Xu, Ming Dai, and Weijie Zhou who have made my personal life colorful. I cherish the time we spend together and appreciate the fortune of meeting you all in my Ph.D. life.

Finally, I would like to thank my family. My parents' constant support and unconditional love gave me confidence and energy to pursue my Ph.D. degree abroad. I'm also grateful to meet my wife Fengyi Gao in my Ph.D. life. Meeting you was the best thing that's ever happened to me and you lighted up my life. The smell of your cooking and the sound of your laughter gives me the feel of the warmth of home. Thank you for being such a caring and supportive wife.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Research Overview

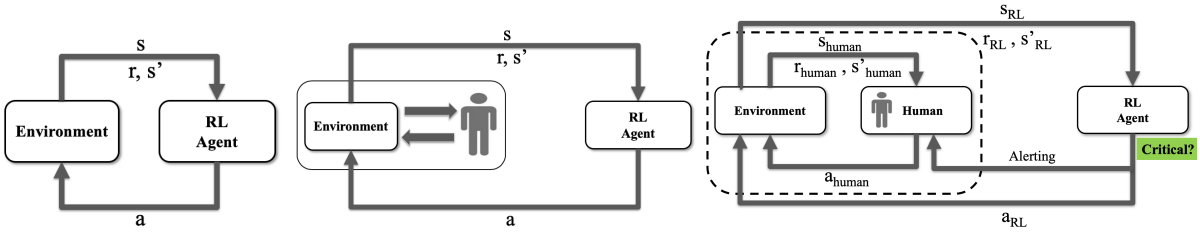
People make many decisions every day, from decisions such as what to eat for lunch, to decisions about which college to enroll. It is estimated that on average people make 35,000 decisions per day [Sol16]. For example, each day people can make an average of 200 decisions related to food alone[Wan07]. Accordingly, we can model people's daily activities (of interest) as sequential decision-making under uncertainty.

Making decisions is vital and determines the trajectory of life. It's in the moments of decision that people's destiny is shaped [Rob92]. In the same way compounding interest increases wealth, making decisions can enhance people to move forward to a positive outcome such as a successful career or a healthy body. Specifically, making decisions often give people a sense of control which entails specific expectations to perform behaviors for obtaining desired outcomes. Thus, people are likely to persist at doing constructive things, like learning, exercising, or quitting smoking, when they are given the choice and when they can make decisions. For example, letting students make decisions during the tutorial process could make them feel that they are actively directing their learning process and not just passively following it. Cordova and Lepper [Cor96] found that offering student choices over their learning could lead to significantly better learning outcomes than those who were not offered. Also, patients' participation in healthcare decision-making could reduce their anxiety and increase the trust of the healthcare professionals which leads to positive and lasting effects on fitness. A study of patients with breast cancer showed that the involvement in decision-making about treatment and follow-up care could significantly improve their health-related quality of life (HRQOL) [MR09].

People are not always good at making decisions. There are many reasons why people make poor decisions. One of them is the *lack of self-control*. For example, in 2018 the Center

for Disease Control and Prevention (CDC) reported that 37% of US adults could be classified as obese, i.e., with Body Mass Index at or above 25 and results show that obese people often exhibit a lower level of self-control than their normal-weight peers, and such lack of self-control can be associated with poor decisions in food and health [Fan14]. Another common reason for making poor decisions is *decision fatigue*. Decision fatigue refers to the phenomena that after a long series of decision-making, people often tend to choose the easiest option without rational thinking, become more procrastinating and less persistent, or fail to recognize decision opportunity [Pig20].

On the other hand, a large number of real-world tasks in science and engineering, from robotics to game playing, tutoring systems, medical treatment design, and beyond, can be characterized as sequential decision-making under uncertainty. ***Reinforcement Learning (RL)*** is one of the most effective machine learning methods for decision making under uncertainty. RL algorithms are designed to induce effective policies that determine the best action for an agent to take in any given situation to maximize a cumulative reward. Figure 1.1 (a) shows how policy induction can be represented as a classic RL. At any given time t, the RL agent observes the environment state s, then chooses an action a from a set of options, and receives a reward r, and the environment transitions into state s'. The RL agent learns the policy for decision-making by estimating the action-value function for each action and choosing the action with the largest value. In recent years, RL, especially Deep RL, has achieved superhuman performance in several complex games [Sil16b; Sil18; And18].



(a) Classic RL (b) Human-centeric RL (c) Human-Machine Mixed-Initiative RL

Figure 1.1: Three RL Frameworks

In this dissertation work, we focus on a different type of task named ***human-centric*** task. In the classic game-play situations, the ultimate goal of the RL agent is to collect as much long-term reward as possible shown in Figure 1.1 (a). In the human-centric task, however, the goal is for the RL agent to improve the human interaction with the environment and as a result, we referred to such framework as *Human-centric RL*, shown in Figure 1.1 (b). Compare the classic RL, in human-centric RL at any given time t, the RL agent observes the *human-environment interaction state* s, then chooses an action a, and receives a reward r based

on *the human's experience*, and the human-environment interaction transitions into the next state s'. In other words, in classic RL, the RL agent interacts with the environment to learn the optimal policy for decision making while in human-centric tasks, the obligation of the RL agent is to make the human-system interactions productive and fruitful: increasing their learning outcomes or decreasing their risk of disease.

Like many machine learning methods, RL is originated from the field of animal behavior, psychology, and neuroscience, in which it is shown that RL is one of the most promising approaches to model and understand the decision-making process in animals and humans [Niv09]. When compared with human decision-making, the RL agent has many advantages including extensive computation power and never getting tired. On the other hand, making humans a decisionmaker gives them a sense of control which can result in great outcomes. Therefore, to balance the pros and cons of the two, we propose a human-machine mixed-initiative decision-making framework.

Figure 1.1 (c) shows the human-machine mixed-initiative decisionmaking framework which involves two loops with two agents: a user agent (human) and an RL agent. In the inner loop, the user interacts with the environment in that he/she will make decisions a_{human} based on his/her perspective of the environment s_{human} and reward function r_{human}; whereas in the outer loop, the RL agent will make decisions a_{RL} based on the various features of the environment s_{RL} defined by the domain experts and a reward function r_{RL}. Based on s_{RL} and r_{RL}, our RL agent will determine how to interact with the inner loop. In this framework, the human is the front-end decisionmaker while the RL agent is the backend to support. The RL agent can be employed to identify the more successful human decisions while alerting when their decisions are suboptimal. Specifically, our RL agent will determine what is the optimal action for the human to take at any moment; then by monitoring the human's actual behavior and matching it to the optimal action, the suboptimal behavior can be identified. The goal is to utilize RL to advise and support human to make effective decisions lead to the desired outcomes and avoid the pitfalls. To balance the trade-off between giving the human a feeling of control over their own progress and keeping the decisions effective, the key problem is to determine *when* the agent should intervene. So, this raises our primary research question: ***Given a long trajectory of decisions, how to identify the critical decisions that lead to the desired outcomes?***

To investigate this research question, we have explored three approaches. First, we apply offline off-policy evaluation (OPE) on historical data to determine the existence of critical decisions in the context of Intelligent Tutoring Systems (ITSs). In a historical student-system interaction log dataset, four groups of critical decisions are identified by four RL-induced policies based upon their Q-value differences. We find that only the group of decisions identified by the effective policy is highly correlated with the students' post-test scores while the others are less so. In other words, the more optimal critical decisions students experienced, the better the learning performance they have.

Second, we propose a novel Adversarial Deep RL (ADRL) framework to identify critical decisions. In this framework, a pair of adversarial policies is induced based upon Deep Q-Network (DQN) with opposite goals: one is to improve student learning while the other is to hinder; critical decisions are identified by comparing the two adversarial policies and using their corresponding Q-value differences; finally, a Critical policy is induced by giving optimal action in critical states but random actions in others. The Critical policy is deployed on a tutor and compared with a random yet reasonable (Random) policy in a classroom study. We find that critical decisions are always occurring in groups (defined as critical phase) and the Critical policy is effective for the subgroup of students who experienced more critical phases. However, there's no significant difference between the Critical policy and the Random policy for the entire population. One potential explanation is that the Random policy is not a weak baseline given that there are only two types of instructional interventions and both of them are reasonable for any given state.

Third, we propose a novel Long-Short Term Rewards (LSTR) framework for critical decision identification. In the LSTR framework, the *long-term* rewards are defined as *Q-value difference* and the *short-term* rewards are determined by the *reward function*. For LSTR, the critical decisions are the union from two sets of critical decisions identified by long-term and short-term rewards separately. Furthermore, we propose a Critical-DQN algorithm to consider the critical decisions in the policy induction process. Experiments on an ideal GridWorld game show that the proposed LSTR framework indeed identifies the critical decisions in the sequences, and the Critical-DQN policy is better than the original DQN policy in identifying critical states. However, the Critical-DQN algorithm is not data-efficiency that it needs more data to converge to an optimal policy. Overall, the result suggests that carrying out the critical decisions alone can be as effective as a fully-executed policy and outperform the random policy.

To investigate whether our LSTR framework is effective and generalizable in real-world applications, we conduct experiments to identify critical decisions in the fields of education and healthcare. In education, we leverage an Intelligent Tutoring System (ITS) that teaches student probability. More specifically, we first evaluate the performance of different critical policies on a historical dataset and then conduct an empirical classroom study to evaluate our LSTR framework. In offline evaluation on the historical dataset, our results reveal that the best critical policy is a combination of Critical-DQN and original DQN, in which the Critical-DQN is used to *identify critical states* while the original DQN is used to select optimal actions in critical states. In our empirical evaluation, we compare our critical policy with two baselines: a critical-suboptimal policy and a fully executed RL policy. Our critical policy would carry out optimal actions in the critical states and random ones in the rest while the critical-suboptimal policy takes *suboptimal actions in the critical states* and random actions in the non-critical states. The fully-executed RL policy would always carry out the optimal actions in all states. Our empirical classroom study shows that 1) the critical policy significantly

outperforms the critical-suboptimal policy and, 2) the critical policy performs similar to the fully executed policy. It suggests that the critical policy indeed identifies the critical decisions in students' learning. For healthcare, we apply the LSTR framework to identify the critical medical interventions for sepsis patients on two healthcare datasets: CCHS and Mayo. The effectiveness of the critical policy is evaluated from two aspects: policy performance and percentage of nudges. Here, the nudge is the moment where it is critical and the policy disagrees with the physician's decision. Our results on sepsis treatment show that the induced critical policy could reduce the percentage of nudges while keeping the septic shock rate as low as a fully executed policy. The results suggest that the LSTR framework could identify the critical decisions in the septic treatment and enhance the physicians' decision-making with minimum interfere.

1.2 Contributions

A large number of prior research has investigated how to make effective decisions in interactive environments. However, most of them rarely consider the importance of decisions. To the best of our knowledge, we are the first one attempting to explicitly investigate the open question: *how to identify the critical decisions that lead to the desired outcomes in a long trajectory of decisions?* and valid the existence of critical decisions across different testbeds (simulation and empirical study) and different domains (education and healthcare).

Overall, our main contributions are summarized as follows:

- Through an offline off-policy evaluation approach, we found evidence of the existence of critical decisions that some decisions are correlated with student learning outcomes while others are not.
- We proposed an ADRL framework to identify critical decisions and found that critical decisions always appear in a consecutive sequence of steps. In other words, in sequential decision-making, critical decisions are not likely to exist individually.
- We proposed a Long Short Term Rewards (LSTR) framework to identify critical decisions and a Critical DQN algorithm to improve the long-term rewards. Evaluation on a synthetic GridWorld game showed that to identify critical decisions, we need to separate critical states from critical decisions that in critical states, optimal actions should be taken while in non-critical states, any action can be taken.
- In the education domain, the effectiveness of our LSTR framework was empirically evaluated from two perspectives: whether optimal actions *must* be carried out in critical states (*necessary hypothesis*) and whether only carrying out optimal actions in critical states is as effective as a fully-executed RL policy (*sufficient hypothesis*). Our results confirmed both hypotheses.

- We generalized our LSTR framework to a healthcare task: sepsis treatment, and validated that the critical policy can be as effective as a fully-executed policy in preventing patients from septic shock and can greatly reduce healthcare workload.

1.3 Organization

The organization of this dissertation is as follows: Chapter 2 provides a review of work that is related to the decision-making in humans and RL. Chapter 3 describes how to apply OPE to identify critical decisions on historical data and what we find. Chapter 4 presents the ADRL framework and a classroom study for identifying the critical decisions. Chapter 5 discusses the LSTR framework and its experiment on an ideal GridWorld game and a real-world ITS dataset. Chapter 6 empirically evaluates the effectiveness of the LSTR framework in a classroom setting. Chapter 7 applies the LSTR framework to two healthcare datasets to identify the critical decisions. Chapter 8 concludes the dissertation and discusses the limitation and future work.

CHAPTER 2

RELATED WORK

In this chapter, we first present the animal and human decision-making behavior studies which inspired our work especially our Long-Short Term Rewards framework. Then, we describe some related work on the decision fatigue phenomenon; this is especially important in the healthcare domain which motivated our work on identifying critical decisions to reduce healthcare workload. Next, we briefly present related work on Reinforcement Learning (RL) with a focus on deep RL; followed by previous work on applying RL and DRL in the fields of Intelligent Tutoring Systems (ITSs) and Healthcare. Finally, we present the related work of utilizing Q-value to identify critical decisions.

2.1 Animal and Human Decision Making

There has been a lot of research on animal and human decision-making [Bud19; San15; Kal11; Rea91]. In the following, we mainly focus on modeling animal and human decision-making processes using the RL framework.

A wealth of neuroscience research focuses on applying RL to understanding the learning and decision-making process in animals and humans. To figure out how the brain generates behavior, the most dominant approach is the RL-based computational method which models the brain as a computing device. Since the brain is an extremely complex dynamic biological system, it is impossible to understand what networks of neurons in the brain represent. The idea of the computational method is to look at what the structure of the brain computes rather than looking at how they function. Recent neuroscience research has revealed in the brain the existence of many key RL signals in the learning and decision-making process.

Morris et al. [Mor06] trained monkeys in a binary-choice instrumental task in which image cues predict rewards with different probabilities. In the experiment, the monkeys faced a computer screen with three keys. The trial was started when the monkey pressed the central

key. There are two kinds of trials: reference and decision trials. In reference trials, one image cue was presented on either the left or the right side of the screen. The monkeys were required to press either the left or the right key, corresponding to the location of the cue. The monkeys could receive liquid rewards only if they touched the correct key. In the decision trials, the monkeys saw two image cues on both sides of the screen and had to choose between them. During the experiment, decision trials were embedded sparsely among a set of reference trials. The electrophysiological recordings showed that the activity of dopamine neurons reflects the value of the option that the monkey is going to select.

Roesch et al. [Roe07] trained rats performing an odor-discrimination task that different odors indicate different rewards and the size of the reward is changing dynamically. In the experiment, one odor indicated one reward on the left side, the second odor indicated one reward on the right side and the third odor signaled two rewards on either side. In forced-choice trials, only one reward was available on one side and the rats needed to choose the correct direction to get the rewards. In free-choice trials, the rats could choose to go to either side to get the rewards. The free-choice trials were sparsely interleaved with the forced-choice trials. The results showed that the activity of dopamine neurons reflects the value of the best option even it is not ultimately selected.

Samuel et al. [McC04; McC07] studied how the human brain responds to the immediate reward and the discounted delayed reward. In the experiment, human participants made a series of binary choices between (smaller, earlier) and (larger, later) money amounts while their brains were scanned using functional magnetic resonance imaging (fMRI). The early option always had a lower value reward than the latter option. At the end of the choosing, only one of the participant's choices was randomly selected to count. The fMRI records showed that the limbic system responds preferentially to the immediate reward while in the contrast, the lateral prefrontal cortex is activated to the choices irrespective of the delay. This means that there are two separate systems in our brain deal with immediate and delayed rewards when making a decision.

The decision-making in animals is stochastic rather than deterministic. It means that even an option has a higher reward than the other, animals still have a chance to select the option with a lower reward. In RL, the Q-value difference defines the probability (desirability) of choosing one of two choices. Multiple research revealed that this kind of signal is involved in the process of action selection. Seo & Lee [Seo08] let monkeys play against a computer opponent in a two-player zero-sum game (matching pennies game) and record the neuron activity during playing. When applying linear regression to model the neuronal activity in different regions of the brain, the results showed that the neurons in the posterior parietal cortex modulated their activity according to the difference in the action-value functions. Sul et al. [Sul11] trained rats to choose freely between two goals that deliver a fixed amount of water reward with different dynamic probabilities. The probability was fixed within 35-45 trials and the water is delivered stochastically. From the analysis of the electrophysiological recordings,

the results indicated that the median agranular cortex conveys significant neural signals for the difference between the action-value functions for two alternative actions.

In summary, current research has shown that neural activity in the brain encodes many key signals in the RL framework such as TD error, Q-value, Q-value difference, immediate and delayed rewards. Insofar, RL is one of the most promising frameworks to model the decision-making process in animals and humans.

2.2 Decision Fatigue

Decision fatigue refers to the deterioration of people's ability to make good decisions after a long session of decision-making. In psychology, a common understanding of decision fatigue is a symptom of ego depletion. The idea is that self-control utilizes a limited pool of mental resources which can be used up by making a decision. When the level of mental resources runs low, people become less persistent and easy to make poor choices without deliberating thinking.

Vohs et al. [Voh08] conducted several empirical experiments to test the hypothesis that making many choices impairs self-control and results in less persistence, more procrastination, and poor performance in a math test. In the first experiment, participants were randomly split into two groups: choice vs. no-choice. In the choice group, participants were required to choose between two versions of each product (white or black t-shirt; red or purple pen) based on their preference. In the no-choice group, participants rated the same products based on the usage in the past (on a scale from 1=never to 5=very often). After the first stage of choosing and rating, all participants attended a self-control task that the level of self-control is measured by how much of a bad-tasting beverage people drink. The result showed that the choice group drinks significantly fewer beverages than the no-choice group. In the second experiment, the two conditions are the same as the first experiment but in the self-control task, participants were tested by how long they could keep their hands in cold water (pain tolerance). This experiment provided the same results that the choice group has lower persistence with less time. In the third experiment, the no-choice group read the college requirement and description of courses while the choice group chose college courses. After the stage of choosing and reading, participants were given 15 minutes to prepare for a test. They were allowed to read magazines and play video games but they didn't know the test won't happen until the last second. The result showed that the choice group was procrastinating and spent more time on time-wasting temptations. In the fourth experiment, two groups were asked to solve an unsolvable math problem and do a solvable math test. In the results, when dealing with unsolvable problems, the no-choice group persistent longer time, when doing the solvable test, the choice group performed significantly worse than the no-choice group. In the last experiment, three conditions were involved: no-choice vs. 4-mins choice vs. 12 mins choice, to test whether a longer period of choice would impair self-control more. In the choosing

phase, participants chose or rated wedding gifts. In the self-control task, they were required to watch a malfunctioned video and measure how long they could sit there before asking for a fix. The results showed that the more choices one makes, the more passive to wait longer.

Overall, these bunch of experiments showed that making too many decisions indeed depletes one's energy and impair self-control. The consequence of losing self-control can be less persistence, more procrastination, more passivity, and even hurt learning performance.

2.3 Reinforcement Learning

Reinforcement Learning (RL) is a type of machine learning framework, which is different from either supervised learning or unsupervised learning. As a framework to solve sequential decision-making problems, it learns from the agent-environment interaction over time. The RL algorithms are designed to induce optimal policies that determine the best action for an agent to take in any given situation to maximize a cumulative reward. Prior research of Reinforcement Learning can be roughly divided into classic RL vs. Deep RL (DRL) approaches. The latter is highly motivated by the fact that the combination of deep learning (neural networks) and novel RL algorithms has made solving complex problems possible in the last decade. As an example of DRL, the Deep Q-Network (DQN) algorithm [Mni15b] takes advantage of convolutional neural networks to learn to play Atari games observing the pixels directly. Since then, DRL has achieved great success in various complex tasks such as game playing [Mni13; Mni15a; Wan16; Mni16; Sil16a], robotic control [Lev16; Zhu17; Zha15], making recommendations [Zhe18; Zha18a; Zha18b], chemical reactions optimization [Zho17b] and also ITS control [Wan17; Nar15].

While promising, relatively little work has been done to analyze, interpret, explain, or generalize RL-induced policies. While traditional hypothesis-driven, cause-and-effect approaches offer clear conceptual and causal insights that can be evaluated and interpreted, RL-induced policies are often large, cumbersome, and difficult to understand. The space of possible policies is exponential in the number of domain features. It is therefore difficult to identify the system decisions that critical to desirable outcomes.

2.4 RL For Intelligent Tutoring System & Healthcare

A lot of research has applied RL to ITS such as inferring students' knowledge [Raf15] and hint generation [Bar08]. For this dissertation, we mainly focus on Pedagogical Policy, which is used to decide what action to take next in the face of alternatives.

Prior research using RL approaches has applied both online and batch/offline approaches to induce pedagogical policies for ITSs. Beck, et al. [Bec00] applied temporal difference learning to induce pedagogical policies that would minimize the students' time on task. Similarly, Iglesias et al. applied Q-learning to induce policies for efficient learning [Igl09a; Igl09b].

More recently, Rafferty et al. applied an online partially observable Markov decision process (POMDP) to induce policies for faster learning [Raf16]. All of the models described above were evaluated via simulations or classroom studies, yielding improved student learning and/or behaviors as compared to some baseline policies. Offline or batch RL approaches, on the other hand, "take advantage of previously collected samples, and generally provide robust convergence guarantees" [Sch17]. Thus, the success of these approaches depends heavily on the quality of the training data. One common convention for collecting an exploratory corpus is to train students on ITSs using *random yet reasonable* policies. Shen et al. applied value iteration and least square policy iteration on a pre-collected exploratory corpus to induce a pedagogical policy that improved students' learning performance [She16b; She16a]. Chi et al. applied policy iteration to induce a pedagogical policy aimed at improving students' learning gain [Chi11]. Mandel et al. [Man14] applied an offline POMDP to induce a policy that aims to improve student performance in an educational game. All the models described above were evaluated in classroom studies and were found to yield certain improved student learning or performance relative to a baseline policy. Wang et al. applied an online DRL approach to induce a policy for adaptive narrative generation in an educational game using simulations [Wan17]; the resulting DRL-induced policies were evaluated via simulations only. In this work, based on the characteristics of our task domain, we focus on batch RL with neural networks, also known as batch Deep Reinforcement Learning (batch DRL) [Jaq19; Fuj19].

Similar to ITS, sepsis treatment can be characterized as a temporal sequential decision-making process, where the outcome of the selected treatment is delayed. In hospitals, physicians often make a larger number of clinical decisions from defining problems, evaluating test results to treatment. Despite the severity of the disease and the challenges faced by practitioners and researchers, it is notoriously difficult to reach an agreement for the optimal treatment due to the complex nature of sepsis and different patients' constitution. Moreover, continuous updates in the sepsis guidelines often lead to inconsistency among clinical practices [Bac17]. On the other hand, RL offers an effective data-driven solution based on a mathematically grounded framework that learns an optimal policy from data to maximize the expected reward [Sut18]. Particularly, Deep RL (DRL) effectively models high-dimensional data and has broadened its coverage to septic treatment [Rag17].

2.5 Critical Decisions in Simulation

One of the closest works to our human-machine mixed-intuitive decisionmaking framework is the so-called "Student-Teacher" framework, which is originally proposed by Jeffery Clouse [Clo96]. In this framework, a "student" agent learns from the interaction with the environment, while a "teacher" agent provides action suggestions to accelerate the learning process. Their research question is not what to advise but when to advise, especially with a limited budget of advice.

Clouse [Clo96] was the first one to study the student-teacher framework in a student-initiated advising mode. They applied the Q-value difference to measure the student's confidence in a state and used it to decide when the student should ask for help. The results showed that compared with random asking, their approach could improve the learning speed significantly. Furthermore, the experiment demonstrated that not all the teacher's advice is equally helpful. The same amount of advice can cause the student agent to take widely varying amounts of steps to find the optimal policy.

Torrey et al. [Tor13] considered the student-teacher framework in a teacher-initiated advising way. They considered an environment with a limited budget of advice and the teacher decided when to give advice. They proposed several heuristic methods to determine when to give advice such as early advising, importance advising, mistake correcting, and predictive advising. The results showed that mistake correcting has the best performance which indicates that advice can have the greatest impact when students make mistakes in important states.

Zimmer et al. [Zim13] modeled the when to advise problem as an RL problem. They learned a teaching policy with two actions: $A = \{advice, noadvice\}$ to decide when to advise the student. Compared with heuristic methods, the result showed that the teacher policy is effective because it can learn not only when to give advice, but also distinguish good and bad student agent that good agent chooses a lot of good actions and doesn't need advice while bad agent needs more.

Amir et al. [Ami16] studied the jointly-initiated strategies for the student-teacher learning framework. In their model, both student and teacher can initiate advising based on heuristic functions. The motivation of their work is to reduce the pressure of the teacher agent on monitoring the student constantly and make the framework closer to the real-life student-agent scenario. The result showed that the joint decision-making approach could reduce the attentions required from the teacher but still keep the student learning effectively.

Fachantidis et al. [Fac17] explored the impact of advice quality in the student-teacher framework. They distinguished teacher agents to be an expert or a good teacher who provide optimal or sub-optimal advice. Also, a Q-teaching method was proposed to learn a teaching policy to decide when to give advice. Their results showed that the best performers are not always the best teachers and the Q-teaching approach is significantly more efficient than others.

In summary, prior works investigated the problem of when to advise in simulated environments. They showed that Q-value difference is a robust and accurate heuristic function to estimate the importance of the decision in interactive environments. However, prior works only considered RL-based student agents but not human students.

CHAPTER 3

OFFLINE OFF-POLICY EVALUATION TO IDENTIFY CRITICAL DECISION

Song Ju, Guojing Zhou, Hamoon Azizsoltani, Tiffany Barnes, Min Chi, Importance Sampling to Identify Empirically Valid Policies and their Critical Decisions. EDM (Workshop) 2019: 69-78

This chapter describes our first attempt to investigate critical decisions. First, we explored the use of three common Importance Sampling-based metrics to evaluate four RL-induced policies on a historical dataset from a logic ITS. Our result showed that Per Decision Importance Sampling (PDIS) is the best metric to evaluate RL-induced policies offline. Then, we used Q-value difference to identify critical decisions and four groups of decisions were identified by the four RL-induced policies. We found that only the group of decisions identified by the effective policy (evaluated by PDIS) is highly correlated with students' learning performance while the others are less so. In other words, the students who experienced more optimal critical decisions significantly outperformed those who received less. Overall, we found the existence of critical decisions in students' learning, and these critical decisions can be identified by using the theoretically "effective" policies that are identified by using PDIS.

3.1 Introduction

Intelligent Tutoring Systems (ITSs) are a type of highly interactive e-learning environment that facilitates learning by providing step-by-step support and contextualized feedback to individual students [Koe97; Van06]. These step-by-step behaviors can be viewed as a sequential decision process where at each step the system chooses an action (e.g. give a hint, show an example) from a set of options, in which *pedagogical strategies* are policies that are used to decide what action to take next in the face of alternatives. Reinforcement Learning (RL) offers

one of the most promising approaches to data-driven decision-making applications and RL algorithms are designed to induce effective policies that determine the best action for an agent to take in any given situation to maximize some predefined cumulative reward. A number of researchers have studied the application of existing RL algorithms to improve the effectiveness of ITSs [Chi11; She16b; Raf16; Cle16; Sta11; Igl09a; Igl09b; Zho17a]. While promising, such RL work faces at least two major challenges discussed below.

One challenge is a lack of reliable yet robust evaluation metrics for RL policy evaluation. Generally speaking, there are two major categories of RL: online and offline. In the former category, the agent learns while interacting with the environment; in the latter case, the agent learns the policy from pre-collected data. Online RL algorithms are generally appropriate for domains where interacting with simulations and actual environments are computationally cheap and feasible. On the other hand, for domains such as e-learning, building accurate simulations or simulated students is especially challenging because human learning is a rather complex, poorly understood process. Moreover, learning policies while interacting with students may not be feasible, and more importantly, may not be ethical. Therefore, to improve student learning, much prior work applied offline RL approaches to induce effective pedagogical strategies. This is done by first collecting a training corpus and the success of offline RL is often heavily dependent on the quality of the training corpus. One common convention is to collect an exploratory corpus by training a group of students on an ITS that makes random yet *reasonable* decisions and then apply RL to induce pedagogical policies from that training corpus. An empirical study is then conducted from a new group of human subjects interacting with different versions of the system. The only difference among the system versions is the policy employed by the ITS. The students' performance is then statistically compared. Due to cost limitations, typically, only the *best* RL-induced policy is deployed and compared against some baseline policies. On the other hand, we often have a large number of RL algorithms (and associated hyperparameter settings), and it is unclear which will work *best* in our setting. In these high-stakes situations, one needs confidence in the RL-induced policy before risking deployment. Therefore, we need to develop reliable yet robust evaluation metrics to evaluate these RL-induced policies without collecting new data before being tested in the real world. This type of evaluation is called *off-policy evaluation* (OPE) because the policy used to collect the training data, also referred to as the *behavior* policy, is different from the RL-induced policy, referred to as the *target* policy to be evaluated. To find reliable yet robust OPE metrics, we explored three Importance Sampling based off-policy evaluation metrics.

The second RL challenge is a lack of interpretability of the RL-induced policies. Compared with the amount of research done on applying RL to induce policies, relatively little work has been done to analyze, interpret, or explain RL-induced policies. While traditional hypothesis-driven, cause-and-effect approaches offer clear conceptual and causal insights that can be evaluated and interpreted, RL-induced policies are often large, cumbersome, and difficult to understand. The space of possible policies is exponential in the number of domain features. It

is therefore difficult to draw general conclusions from them to advance our understanding of the domain. This raises a major open question: *How can we identify the critical system interactive decisions that are linked to student learning?* In this work, we tried to identify key decisions by taking advantage of the reliable OPE metrics we discovered and the properties of the policies we induced.

3.2 Motivation

Just like the fact that assessment sits at the epicenter of educational research [Bra99], policy evaluation is indeed the central concern among the many stakeholders in applying offline RL to ITSs. As educational assessment should reflect and reinforce the educational goals that society deems valuable, our policy evaluation metrics should reflect the effectiveness of the induced policies. While various RL approaches such as policy iteration and policy search have shown great promise, existing RL approaches tend to perform poorly when they are actually implemented and evaluated in the real world.

In a series of prior studies on a logic ITS, RL and Markov Decision Processes (MDPs) were applied to induce *four* different pedagogical policies, named MDP1-MDP4 respectively, on one type of tutorial decision: whether to provide students with a *Worked Example* (WE) or to ask them to engage in *Problem Solving* (PS). In WEs, the tutor presents an expert solution to a problem step by step, while in PSs, students are required to complete the problem with the tutor's support. When inducing each of four policies, we explored different feature selection methods and used *Expected Cumulative Reward (ECR)* to evaluate the RL-induced policies. ECR of a policy is calculated by average over the value function of initial states and generally speaking, the higher the ECR value of a policy, the better the policy is supposed to perform.

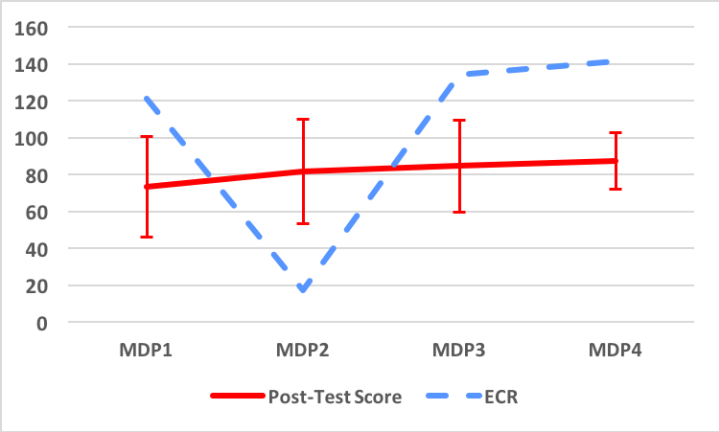


Figure 3.1: Post-test Score vs. ECR, showing no seeming direct relationship.

Figure 3.1 shows the ECRs (blue dashed line) of the four RL-induced policies, MDP1-MDP4

(x-axis), and the *empirical* results of student learning performance (the solid red line) of the corresponding policies. Here the learning performance is measured by an in-class post-test after students were trained on the tutor with corresponding policies (the mean and standard errors of post-test scores are shown with a red solid line). Figure 3.1 shows that our *theoretical* evaluation (ECR) does not match the *empirical* results (post-test) evaluation in that there is no clear relationship between the ECRs' blue line and the corresponding post-test in the red line across the four policies. This result shows that ECR is not a reliable OPE metric for evaluating RL-induced policy in ITSs. Indeed, Mandel et al. [Man14] pointed out that ECR tends to be biased, statistically inconsistent, and thus it may not be the appropriate OPE metric in high stakes domains. In recent years, many state-of-the-art OPE metrics have been proposed and many of them are based on Importance Sampling.

Importance Sampling (IS) is a classic OPE method for evaluating a *target* policy on existing data obtained from an alternate *behavior* policy and thus can be handily applied to the task of evaluating the effectiveness of an offline RL-induced policy using pre-existing historical training datasets. Many IS-based OPE metrics are proposed and explored and it was shown that they gain significant performance in simulation environments like Grid World or Bandit [Tho15; Dud11]. Among them, three IS-based OPE metrics, the original IS, Weighted IS (WIS), and Per-Decision IS (PDIS), are the most widely used. However, real-world human-agent interactive applications such as ITSs are much more complicated due to 1) individual differences, noise, and randomness during the interaction processes, 2) the large state space that can impact student learning, and 3) long trajectories due to the nature of the learning process.

In this work, we investigated the three IS-based offline OPE metrics on MDP1-MDP4 to investigate whether the three IS-based evaluation metrics are indeed effective OPE metrics for evaluating the four RL-induced policies mentioned beforehand. We believe an OPE is effective *if and only if* the *theoretical* results from the OPE evaluations are *completely aligned* with the *empirical* results from the classroom studies. Therefore, we explored different deployment settings for the IS-based metrics from two aspects: one is the transformation function used to convert the RL-induced deterministic policy to a stochastic policy used in IS-based metrics and the other is reward functions: the original reward function vs. the normalized reward function; the latter is supposed to reduce the variance. Our results showed that the theoretical and empirical evaluation results are aligned more or less for different deployment settings using different IS-based metrics. Only when using a soft-max transformation function and original reward function, the theoretical results of PDIS can reach 100% agreement with the empirical results. Based on results from the OPE metrics, we further explored using the properties of the RL-induced policy to identify *critical decisions* and our results showed that the critical decisions can be identified by using the theoretically "effective" policies that were identified by using PDIS with soft-max transformation and the original reward function.

In summary, we make the following contributions:

- We directly compared three IS-based policy evaluation metrics against the empirical results from real classroom studies across four different RL-induced policies. Our results showed that PDIS is the best one and its results can align with empirical results.
- As far as we know, this is the first study to compare different deployment settings (original/normalized rewards or deterministic/stochastic policy transformation) on IS-based policy evaluation metrics. Our results showed that settings have a direct impact on the effectiveness of the evaluation metrics. Only PDIS with soft-max transformation and the original reward function agreed 100% with the empirical results.
- We investigated using information from the RL-induced policies to identify *critical decisions* to shed some light on the induced policies. As far as we know, this is the first attempt to differentiate critical decisions from trivial ones.

3.3 Related Work

3.3.1 OPE Metrics

OPE is used to evaluate the performance of a target policy given historical data generated by an alternative behavior policy. A good OPE metric is especially important for real-world applications where the deployment of a bad or inefficient policy can be costly [PST16]. ECR is one of the most widely used OPE metrics, which is designed especially for the MDP framework. Tetreault et al. [JRT07] estimated the reliability of ECRs by repeated sampling to estimate *confidence intervals* for ECRs. In simulation studies, they showed the policy induced by the confidence interval of ECR performed more reliably than the baseline policies but this phenomenon did not hold when evaluating the RL-induced policies in ITSs for empirical studies [Chi11].

Importance Sampling (IS) [Ham64] is a widely used OPE metric, which considers the mathematical characteristics of the decision-making process and can be applied to any MDP, POMDP, or DeepRL framework. Precup [Pre00] proposed four IS-based OPE metrics: IS, weighted importance sampling (WIS), per-decision importance sampling (PDIS), and weighted per-decision importance sampling (WPDIS). They used the IS-based estimator as the policy evaluation for Q-learning and then compared the effectiveness of estimators on a series of 100 randomly constructed MDPs based on the mean square error (MSE). Their results showed that IS made the Q-learning process converge slowly and caused high variance, and WIS performed better than IS, but PDIS performed inconsistently and WPDIS performed the worst. Similarly, Thomas [Tho15] compared the performance of several IS estimators using mean squared error in a grid-world simulation, showing PDIS outperformed all others.

In summary, previous work has explored the effectiveness of IS and its variants in simulation studies, which motivated the work reported here. Different from previous work, we

mainly focus on comparing the theoretical evaluation with the empirical evaluation in order to determine whether IS-based methods are indeed reliable and robust for ITSs.

3.4 Markov Decision Process & RL

Some of the prior work on applying RL to induce pedagogical policies used Markov Decision Processes (MDP) frameworks. An MDP can be seen as a 4-tuple $\langle S, A, T, R \rangle$, where S denotes the observable state space, which is defined by a set of features that represent the interactive learning environment and A denotes the space of possible actions for the agent to execute. The reward function R represents the immediate or delayed feedback from the environment with respect to the agent's action(s); $r(s, a, s')$ denotes the expected reward of transiting from state s to state s' by taking action a. Once $\langle S, A, R \rangle$ is defined, T represents the transition probability where $P(s, a, s') = Pr(s'|s, a)$ is the probability of transitioning from state s to state s' by taking action a and it can be easily estimated from the training corpus. The optimal policy π for an MDP can be generated via dynamic programming approaches, such as Value Iteration. This algorithm operates by finding the optimal value for each state $V^*(s)$, which is the expected discounted reward that the agent will gain if it starts in s and follows the optimal policy to the goal. Generally speaking, $V^*(s)$ can be obtained by the optimal value function for each state-action pair $Q^*(s, a)$ which is defined as the expected discounted reward the agent will gain if it takes an action a in a state s and follows the optimal policy to the end. The optimal value function $Q^*(s, a)$ can be obtained by iteratively updating $Q(s, a)$ via equation 3.1 until convergence:

$$Q(s, a) := \sum_{s'} p(s, a, s') \left[r(s, a, s') + \gamma \max_{a'} Q(s', a') \right] \tag{3.1}$$

where $0 \leq \gamma \leq 1$ is a discount factor. When the process converges, the optimal policy π^* can be induced corresponding to the optimal Q-value function $Q^*(s, a)$, represented as:

$$\pi^*(s) = \arg\max_{a} Q^*(s, a) \tag{3.2}$$

Where π^* is the *deterministic* policy that maps a given state into an action. In the context of an ITS, this induced policy represents the pedagogical strategy by specifying tutorial actions using the current state.

3.5 Three OPE Metrics & Two Settings

The following terms will be used throughout this paper.

- $H = s_1 \xrightarrow{a_1, r_1} s_2 \xrightarrow{a_2, r_2} s_3 \xrightarrow{a_3, r_3} \cdots s_L$; denotes *one* student-system interaction trajectory and H^L denotes a trajectory with length L.

- $G(H^L) = \sum_{t=1}^{L} \gamma^{t-1} r_t$ is the discounted return of the trajectory H^L, which generally reflects how good the trajectory is supposed to be.
- $D = \{H_1, H_2, H_3, ..., H_n\}$ denotes the historical dataset containing n student-system interaction trajectories.
- π_b denotes the *behavior* policy carried out for collecting the historical data D.
- π_e denotes the target policy to be *evaluated*.
- $\rho(\pi_e)$ represents the estimated performance of π_e.

3.5.1 Three IS-based OPE Metrics

Importance Sampling (IS) is an approximation method that allows the estimation of the expectation of a distribution, p, from samples generated from a different distribution, q. Suppose that we have sample space x and random variable $f(x)$ which is a measurable function from sample space x to another measurable space. We want to estimate the expectation of $f(x)$ over a strictly positive probability density function $p(x)$. Suppose also that we cannot directly sample from distribution $p(x)$, but we can draw Independent and Identically Distributed (IID) samples from probability density function $q(x)$ and evaluate $f(x)$ for these samples. The expectation of $f(x)$ over probability density function $p(x)$ can be calculated as:

$$
\begin{aligned}
E_p[f(x)] &= \int f(x)p(x)dx && (3.3)\\
&= \int f(x)\frac{p(x)}{q(x)}q(x)dx && (3.4)\\
&= E_q[f(x)\frac{p(x)}{q(x)}] && (3.5)
\end{aligned}
$$

where p is known as the target distribution, q is the sampling distribution, $E_p[f(x)]$ is the expectation of $f(x)$ under p, $p(x)/q(x)$ is the likelihood ratio weight and $E_q[f(x)p(x)/q(x)]$ is the expectation of $f(x)p(x)/q(x)$ under q. We can then approximate the expectation of $f(x)$ over probability density function $p(x)$ using the samples drawn from probability density function $q(x)$. In the context of OPE, the target distribution, p, is a probability event whose density function is determined by the target policy, and the sampling distribution, $p(x)$, is a probability event whose density function is determined by the behaviour policy.

Following the general IS technique, we approximated the expected reward of the target policy using the relative probability of the target and behavior policies. Because of the nature of the underlying MDP, samples in RL are sequential. Therefore, we assumed that each trajectory is a sequence of events whose probability density function is determined by its corresponding policy. Assuming the independence among trajectories and following the multiplication rule

of independent events, the probability of occurrence of a trajectory, H^L, under policy π is

$$\mathcal{P}_\pi(H^L) = \mathcal{P}_1(s_1)\prod_{t=1}^{L}\pi(a_t|s_t)\mathcal{P}_T(s_{t+1}|s_t,a_t) \tag{3.6}$$

where $\mathcal{P}_\pi(H^L)$ is the probability of occurrence of the trajectory H^L following a policy π, $\mathcal{P}_1(s_1)$ is the probability of occurrence of the state s_1 at the beginning of the trajectory and $\mathcal{P}_T$ is the state transition probability function.

Similar to the original IS technique, the proportional probability of each trajectory occurring under the target policy, π_e, and behavior policy, π_b, is used as the likelihood ratio weight for that trajectory. Thus, following the importance sampling technique, the importance sampling discounted return is defined as:

$$IS(\pi_e|H^L,\pi_b) = \frac{\mathcal{P}_{\pi_e}(H^L)}{\mathcal{P}_{\pi_b}(H^L)}\cdot G(H^L) \tag{3.7}$$

$$= \frac{\prod_{t=1}^{L}\pi_e(a_t|s_t)}{\prod_{t=1}^{L}\pi_b(a_t|s_t)}\cdot G(H^L) \tag{3.8}$$

Substituting $G(H^L)$ with the discounted return and we have:

$$IS(\pi_e|H^L,\pi_b) = (\prod_{t=1}^{L}\frac{\pi_e(a_t|s_t)}{\pi_b(a_t|s_t)})(\sum_{t=1}^{L}\gamma^{t-1}r_t) \tag{3.9}$$

After having the individual IS estimator for each trajectory, we can calculate the expected reward of the dataset D by averaging the individual IS estimators for each trajectory as

$$IS(\pi_e|D) = \frac{1}{n_D}\sum_{i=1}^{n_D}\left[(\prod_{t=1}^{L^i}\frac{\pi_e(a_t^i|s_t^i)}{\pi_b(a_t^i|s_t^i)})(\sum_{t=1}^{L^i}\gamma^{t-1}r_t^i)\right] \tag{3.10}$$

where s_t^i, a_t^i, and r_t^i refer to the ith trajectory at time t and n_D is the number of trajectories in D.
Weighted Importance Sampling (WIS) is a variant of the IS estimator, a biased but consistent estimator which has lower variance than IS. It normalizes the IS to produce a lower variance. At first, it calculates the weight W_D for the dataset as the summation of the likelihood ratios for each trajectory as shown in equation 3.11. Then, it normalizes the IS estimator as shown in equation 3.12. Finally, the WIS is simply the weighted average of the estimated reward for each sequence in the dataset D, as shown in equation 3.13.

$$W_D = \sum_{i=1}^{n_D}(\prod_{t=1}^{L^i}\frac{\pi_e(a_t^i|s_t^i)}{\pi_b(a_t^i|s_t^i)}) \tag{3.11}$$

$$WIS(\pi_e|H^L,\pi_b) = \frac{IS(\pi_e|H^L,\pi_b)}{W_D} \tag{3.12}$$

$$WIS(\pi_e|D) = \frac{\sum_{i=1}^{n_D}\left[(\prod_{t=1}^{L^i}\frac{\pi_e(a_t^i|^i s_t)}{\pi_b(a_t^i|s_t^i)})(\sum_{t=1}^{L^i}\gamma^{t-1}r_t^i)\right]}{\sum_{i=1}^{n_D}(\prod_{t=1}^{L^i}\frac{\pi_e(a_t^i|s_t^i)}{\pi_b(a_t^i|s_t^i)})} \tag{3.13}$$

Per-Decision Importance Sampling (PDIS) is also a variant of IS. Like IS, it is also unbiased and consistent. IS has a very high variance because, for each reward, r_t, it uses the likelihood ratio of the entire trajectory. However, the reward at step t should only depend on the previous steps. The variance can be reduced by using the likelihood ratio of the trajectory before step t for the reward r_t. It means that the importance weight for a reward at step t is $\prod_{j=1}^{t}\frac{\pi_e(a_j|s_j)}{\pi_b(a_j|s_j)}$. Therefore, the individual PDIS estimator for a trajectory is given in the equation 3.14 and the PDIS for the whole historical dataset D is given in the equation3.15.

$$PDIS(\pi_e|H^L,\pi_b) = \sum_{t=1}^{L}\gamma^{t-1}(\prod_{j=1}^{t}\frac{\pi_e(a_j|s_j)}{\pi_b(a_j|s_j)})r_t \tag{3.14}$$

$$PDIS(\pi_e|D) = \frac{1}{n_D}\sum_{i=1}^{n_D}\left[\sum_{t=1}^{L^i}\gamma^{t-1}(\prod_{j=1}^{t}\frac{\pi_e(a_j^i|s_j^i)}{\pi_b(a_j^i|s_j^i)})r_t^i\right] \tag{3.15}$$

3.5.2 Two Different Settings

To evaluate the three IS-based metrics, we explored different deployment settings from two aspects: one is the transformation function to convert the RL-induced deterministic policy to a stochastic policy used in IS-based metrics and the other is reward functions: the original reward function vs. the normalized reward.

3.5.2.1 Three Transformation Functions:

As described above, IS-based metrics require the policy to be stochastic but our MDP induced policies are deterministic, i.e. given the current state s, the agent should take the deterministic optimal action a^* following the optimal policy π^*. To transform the deterministic policy into a stochastic policy, we explore three types of transformation functions: Hard-code, Q-proportion, and Soft-max. The basic idea behind them is: for any given state s, the assigned stochastic probability for an action a should reflect its value of $Q(s,a)$.

1. **Hard-code Transformation**

$$\pi(a|s) = \begin{cases} 1-\varepsilon & \text{optimal action} \\ \varepsilon & \text{otherwise} \end{cases} \tag{3.16}$$

In eqn 3.16, ε is a fixed small probability like $\varepsilon = 0.001$ which is assigned to actions with smaller Q-value while $1 - \varepsilon$ is assigned to the action with the largest Q-value.

2. **Q-proportion Transformation**

$$\pi(a|s) = \frac{Q(s,a)}{\sum_{a' \in A} Q(s,a')} \tag{3.17}$$

As shown in eqn 3.17, the probability to take action a given state s is the proportion of a's Q-value among all possible actions $a^{'}$ in the state s. Thus, the action which has the highest Q-value is guaranteed to have the highest probability. In practice, because some of the Q-values are smaller than 0, we add a constant to Q-values in the same state so that $Q(s,a) \geq 1$.

3. **Soft-max Transformation**

$$\pi(a|s) = \frac{e^{\theta \cdot Q(s,a)}}{\sum_{a' \in A} e^{\theta \cdot Q(s,a')}} \tag{3.18}$$

Soft-max is a classical function used to calculate the probability distribution of one event over n possible events. In our application, given state s, the soft-max function will calculate the probability of action a over all possible actions using the equation 3.18. The main advantage of soft-max is the output probability range is 0 to 1 and the sum of all the probabilities will be 1 and it can handle negative Q-values. θ is a weight parameter to the Q-value.

3.5.2.2 Two Types of Reward Functions:

The effectiveness of RL-induced policies is very sensitive to the reward functions. In our application, the range of the reward function is very large $[-200, 200]$, which may cause large variances for IS, especially when a trajectory is long. One effective way to reduce this variance is to use normalized rewards. Therefore, both original rewards and normalized rewards are considered. More specifically, the normalized reward z is defined as: $z = \frac{x - min(x)}{max(x) - min(x)}$ where min and max are the minimum and maximum values of original reward function x and $z \in [0, 1]$.

3.6 ITS & Four MDP Policies

3.6.1 Our Logic Tutor

Figure 3.2 shows an interface of the logic tutor, which is a data-driven ITS used in the undergraduate Discrete Mathematics (DM) course at a large university. It [MB15] provides students with a graph-based representation of logic proofs which allows students to solve problems by applying rules to derive new statements, represented as nodes. The system

automatically verifies proofs and provides immediate feedback on logical errors. Every problem in the tutor can be presented in the form of either WE or PS. By focusing on the pedagogical decisions of WE and PS, the tutor allows us to strictly control the content to be *equivalent* for all students.

Figure 3.2: Tutor Problem Solving Interface

3.6.2 Four MDP Policies and Empirical Study

Four MDP policies, MDP_1 - MDP_4 were induced from an exploratory pre-collected dataset following different feature selection procedures. The detailed descriptions of our policy induction process are described in [She16b; She16a] and will be omitted here to save space. The effectiveness of each MDP policy was empirically evaluated against the Random policy in strictly controlled studies during three consecutive semesters. In each strictly controlled study, students were randomly assigned to two conditions: MDP policy, or a Random baseline policy which makes random yet reasonable decisions because both *PS* and *WE* are always considered to be *reasonable* educational interventions in our learning context. Moreover, all students went through the identical procedure on the tutor and the only difference was the pedagogical policy employed. After completing the tutor, students take a post-test which involved two proof questions in a midterm exam. They were 16 points each and graded by one TA using a rubric. Overall, no significant difference was found between our four RL-induced policies and the Random policy on students' *post-test scores* across all studies.

There are many possible explanations for our results. First, while the Random policy is generally a weak policy for many RL tasks, in our situation both of our action choices: WE vs. PS are considered reasonable and more importantly for each decision point, there is a 50% chance that the random policy would carry out the better of the two. Second, non-significant

statistical results do not mean non-existence. Small sample size may play an important role in limiting the significance of statistical comparisons. A post hoc power analysis revealed that to be detected as significant at the 5% level with 80% power, MDP1 vs. Random needed a total sample of 1382 students; MDP2 vs. Random needed 1700 students; MDP3 vs. Random needed 212 students; MDP4 vs. Random needed 394 students. However, in each empirical study, our student sample sizes are much smaller, 59, 50, 57, and 84 respectively. And last but not least, it turned out that all four RL-induced policies were only partially carried out. All of the training problems in our tutor are organized into six strictly ordered levels and in each level students are required to complete 3–4 problems. In level 1, all participants receive the same set of PS problems, and in levels 2–6, our tutor has two hard-coded *action-based constraints* that are required by the class instructors: students must complete at least one PS and one WE and the last problem on each level must be PS. Therefore, over the entire training process, only $\sim 50\%$ of actions are actually decided by the pedagogical policy, and the rest are decided by hard-coded system rules.

In short, although ECR showed that our four RL-induced policies should be more effective than the Random, empirical results showed otherwise because of various potential reasons. So we explored other OPE metrics to evaluate MDP_1 - MDP_4.

3.7 Experiment Setup

Our dataset contains a total of 450 students' interaction logs involved in the strictly controlled studies mentioned above. The goals of our experiment were to 1) investigate whether any of the three IS-based metrics can be used to align the theoretical and empirical results for our four MDP policies and 2) identify critical decisions that are linked to student learning.

3.7.1 Three IS-based Metrics Evaluation

We will describe how we determine whether or not the three IS-based metrics can align the theoretical results with the empirical results for MDP1-MDP4.

For a given RL-induced policy π, we first split all students into High vs. Low based on the actual carry-out percentages according to π. Since there are only two tutorial choices: WE vs. PS, there is a possibility that each actual decision that the tutor made would agree with the decision according to π. For the Random policy, for example, the probability is 50-50. In other words, we can measure each trajectory by the percentage of the tutorial decisions that agree with π. If π is indeed effective, we would expect that the more the tutorial decisions in a trajectory agree with π, the better the corresponding student performance would be. We thus treat all 450 students' interaction log data *equally* regardless of their original assigned conditions and for each student-ITS interaction log we can calculate a carry-out percentage for π using the formula: $percentage = \frac{N_{agree}}{N_{total}}$, where N_{total} is the total number of tutorial decisions in the trajectory and N_{agree} is the number of decisions that agree with π. Then, students are

divided into High Carry-out (High) vs. Low Carry-out (Low) by a median split on carry-out percentages.

Then we *empirically evaluate* the effectiveness of π by checking whether there is a significant difference between the High and Low groups on their post-test scores. Similarly, we compare the High and Low groups' *theoretical* results. To do so, for each student-ITS interaction trajectory, we estimate its reward by exploring different combinations of the three IS-based OPE metrics with the three policy transformation functions and the two reward functions. More specifically, we treat all the trajectories as generated by Random policy regardless of their original behavior policy. So for each student, we have a total of 18 theoretical evaluations for a given π. If π is indeed effective and our OPE metric is reliable, we would expect that the more the tutorial decisions in a trajectory agree with π, the higher the corresponding theoretical rewards would be and vice versa.

Finally, for each of the 18 OPE metric settings, we conduct an alignment test between the theoretical and empirical results on π. This is done by comparing the empirically evaluated results and the theoretical rewards using the corresponding OPE metric. More specifically, they are considered to be **aligned** when:

1. Both empirical and theoretical results were not significant, that is $p >= 0.05$, or
2. Both results were significant, and the direction of the comparison was the same, that is $p < 0.05$, and the sign of the t values are both positive or both negative.

All the remaining cases are considered as not aligned. Thus, for each of 18 OPE metrics, we can test whether its theoretical results would align with the empirical results for π. Since we have four RL-induced policies, MDP1-MDP4, robust and reliable OPE metrics should align the two types of evaluation results across all four policies.

3.7.2 Critical Decision Identification

Next, we will explain how the critical interactive decisions are identified and empirically examined. Note that, there may be critical decisions over which the RL policies have no influence. Hence, we focus only on interactive decisions that are critical. For many RL algorithms, the fundamental approach to induce an optimal policy can be seen as recursively estimating the Q-values: $Q(s, a)$ for any given state-action pair until the Bellman equation is converged. More specifically, $Q(s, a)$ is defined as the expected discounted reward the agent will gain if it takes an action a in a state s and follows the corresponding optimal policy to the end. Thus, for a given state s, a large difference between the values of $Q(s, \text{“}PS\text{”})$ and $Q(s, \text{“}WE\text{”})$ indicates that it is more important for the ITS to follow the optimal decision in the state s. We, therefore, used the absolute difference between the Q-values for each state s to identify critical decisions. Our procedure can be divided into two steps.

Step 1: Identify Critical Decisions: Given an MDP policy, for each state, we calculated the absolute Q-value difference between the two actions (PS vs. WE) associated with it. Figure 3.3

shows the Q-value difference (y-axis) for each state (x-axis) sorted in descending order for MDP1-MDP4 policies respectively. It clearly shows that, across the four MDP policies, the Q-value differences for different states can vary greatly. We used the median Q-value difference to split the x-axis *states* into critical vs. non-critical states. The states with the larger Q-value differences were *critical states* and the rest were non-critical ones. For a given RL-induced policy, the *critical decisions* are defined as those in critical states where the actual carried-out tutorial action agreed with the corresponding policy.

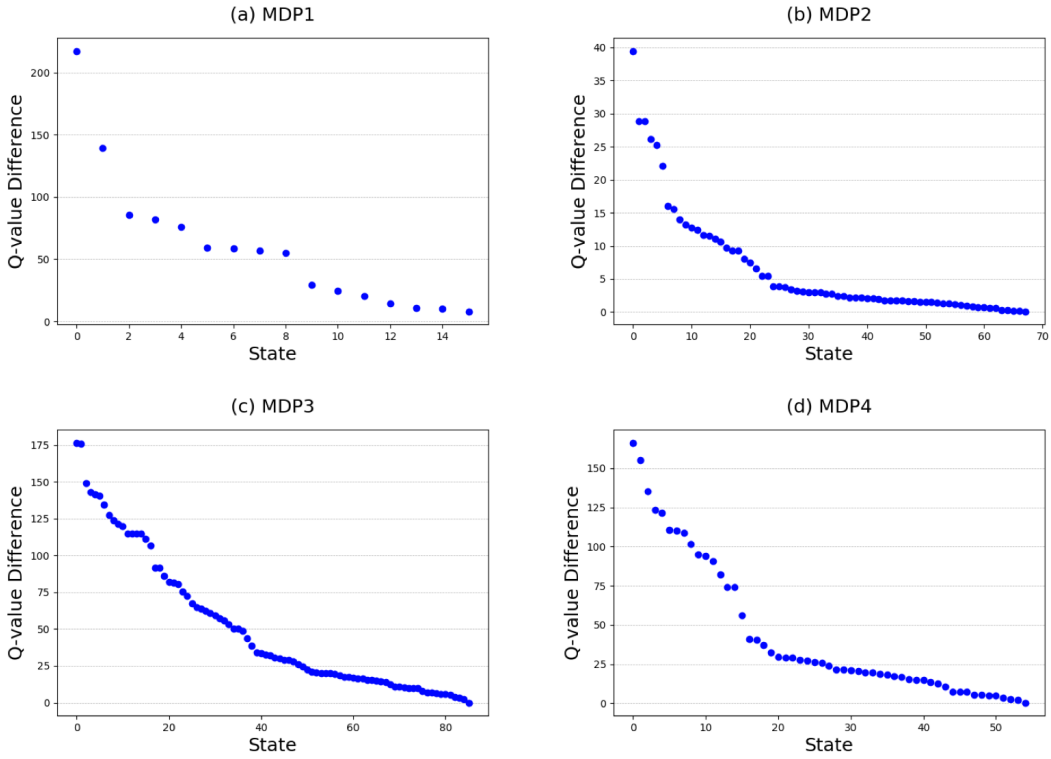


Figure 3.3: Q-value difference in MDP1-MDP4

Step 2: Evaluate Critical Decisions: For each of the four RL-induced policies, we counted the number of critical decisions that each student encountered during his/her training. Then for each policy, students were split into: More vs. Less groups, by a median split on the number of critical decisions experienced in the training process. A t-test was conducted on the post-test scores of More vs. Less groups to investigate whether the students with More critical decisions would indeed perform better than those with Less.

Table 3.1: Empirical Post-test Evaluation Results for High and Low Carry-out Groups

Policy	High	Low	T-test Result
MDP1	79.06(24.64)	83.13(23.69)	$t(448) = -1.78, p = .076$
MDP2	81.46(24.90)	80.44(23.66)	$t(448) = .44, p = .658$
MDP3	82.55(23.48)	79.30(25.00)	$t(448) = 1.24, p = .156$
MDP4	83.43(22.83)	78.43(25.44)	$\mathbf{t(448) = 2.19, p = .029}^{**}$

bold and ** denote significance at $p < 0.05$.

3.8 Results

3.8.1 Three IS-based Metrics Evaluation

Table 3.1 shows the empirical evaluation results by comparing the High vs. Low carry-out groups' post-test scores based on the corresponding RL-induced policy. The motivation is that, if a policy is indeed effective, students in the High group should significantly out-perform their Low peers on the post-test. In Table 3.1, the first column indicates the name of the RL-induced policy; columns 2 and 3 show the mean and standard deviation of classroom post-test scores for High and Low carry-out groups, respectively, and the last column shows the t-test results when comparing post-test between groups. Rows 2-4 show that there is no significant difference between the High vs. Low groups in terms of post-test scores for MDP1, MDP2, and MDP3 policies, but there is a significant difference between the two groups for the MDP4 policy (row 5): $t(448) = 2.19, p = .029$. This result suggests that, among the four MDP policies, only MDP4 seems to be effective in that the students in MDP4's High carry-out group performed significantly better than those in the Low carry-out group.

Table 3.2: Policy Transformation and Normalization Impacts on IS Metric Alignment to Post-test Outcomes

Policy Transformation	Rewards	Metrics		
		IS	WIS	PDIS
Soft-max	Original	50%	50%	**100**%$*$
	Normalized	50%	50%	50%
Q-proportion	Original	25%	25%	75%
	Normalized	25%	25%	25%
Hard-code	Original	75%	75%	75%
	Normalized	75%	75%	75%

Table 3.2 shows the overall IS-based metrics evaluation results showing the impact of

Table 3.3: Detailed IS-based Metrics Evaluation Results Using Original Reward

Transform	Policy	Empirical Result	IS & WIS Result	PDIS Result
Soft-max	MDP1	$t(448) = -1.78, p = .076$	$t(448) = .89, p = .376$	$t(448) = .89, p = .375$
	MDP2	$t(448) = .44, p = .658$	$\mathbf{t(448) = 2.60, p = .010}^{**}$	$t(448) = 1.84, p = .067$
	MDP3	$t(448) = 1.42, p = .156$	$\mathbf{t(448) = 2.50, p = .013}^{**}$	$t(448) = .53, p = .594$
	MDP4	$\mathbf{t(448) = 2.19, p = .029}^{**}$	$\mathbf{t(448) = 2.18, p = .030}^{**}$	$\mathbf{t(448) = 3.23, p = .001}^{**}$
Q-proportion	MDP1	$t(448) = -1.78, p = .076$	$\mathbf{t(448) = 3.78, p < .001}^{**}$	$t(448) = 1.77, p = .077$
	MDP2	$t(448) = .44, p = .658$	$\mathbf{t(448) = 3.26, p = .001}^{**}$	$\mathbf{t(448) = 2.13, p = .034}^{**}$
	MDP3	$t(448) = 1.42, p = .156$	$\mathbf{t(448) = 2.71, p = .007}^{**}$	$t(448) = .31, p = .760$
	MDP4	$\mathbf{t(448) = 2.19, p = .029}^{**}$	$\mathbf{t(448) = 3.69, p < .001}^{**}$	$\mathbf{t(448) = 2.32, p = .021}^{**}$
Hard-code	MDP1	$t(448) = -1.78, p = .076$	$t(448) = 1.19, p = .233$	$t(448) = .96, p = .337$
	MDP2	$t(448) = .44, p = .658$	$t(448) = .71, p = .479$	$t(448) = .84, p = .404$
	MDP3	$t(448) = 1.42, p = .156$	$\mathbf{t(448) = 2.34, p = .020}^{**}$	$\mathbf{t(448) = 2.06, p = .040}^{**}$
	MDP4	$\mathbf{t(448) = 2.19, p = .029}^{**}$	$\mathbf{t(448) = 2.83, p = .005}^{**}$	$\mathbf{t(448) = 3.73, p < .001}^{**}$

Electric-blue cells denote that the theoretical t-test results align with the empirical t-test results (Column 3);
Grey cells denote misaligned t-test results.

policy transformations and original versus normalized rewards on the outcomes of each IS metric. The first column indicates the type of policy transformation applied and the second column shows whether the rewards are normalized. The third through fifth columns show the performance of each IS metric, where performance is determined by the percent alignment between the IS policy predictions and empirical post-test results. Among the three policy transformations, Q-proportion is the worst, with none of its six performance results better than the corresponding results of Soft-max or Hard-code. Hard-code performs slightly better than Soft-max in most cases but never reaches a 100% match. For reward normalization, the original reward performs better than the normalized reward for the PDIS metric, but there was no effect on IS or WIS.

Comparing the three IS metrics, PDIS shows the greatest performance with all 12 of the performance results being the best. For the last two metrics, IS and WIS, the results are the same because WIS is much like multiplying IS by a constant and this kind of re-scale won't change the result of the t-tests. The metric with the best performance is PDIS with Soft-max policy transformation and the original reward, whose performance is 100%. This means that all the t-test results on the PDIS predictions aligned with those on the empirical results in terms of significance match.

Table 3.3 shows the detailed results for metric evaluations using the original reward, providing t-test results when comparing post-test results between High and Low carry-out groups. The first column in table 3.3 shows the type of policy transformation functions applied. The second column shows the four MDP policies considered when splitting the dataset into High and Low carry-out groups. The third column shows the t-test results of the empirical evaluation of High vs. Low carry-out, which served as the ground truth. The fourth and fifth

columns show the t-test results for the prediction of the three IS metrics: IS, WIS, and PDIS respectively. In the tables, electric-blue cells denote that the theoretical t-test results align with the empirical t-test results (Column 3) while grey cells denote mismatched t-test results. From this table, we can see that only PDIS with soft-max transformation and the original reward results in all four t-tests aligning with the corresponding empirical results. IS and WIS are more likely to predict the significant difference between High vs. Low. Meanwhile, Q-proportion tends to cause the metric to predict more significant difference, while Hard-code tends to predict less.

Table 3.4: Critical Decision Evaluation Results

Policy	**More**	**Less**	**T-test Result**
MDP1	78.49(23.08)	83.01(25.44)	$t(448) = -1.97, p = .049$
MDP2	83.22(23.53)	78.86(24.79)	$t(448) = 1.91, p = .057$
MDP3	79.45(24.59)	82.08(24.00)	$t(448) = -1.14, p = .257$
MDP4	83.54(22.89)	78.74(25.21)	$\mathbf{t(448) = 2.10, p = .036}^{**}$

bold and ** denote significance at $p < 0.05$.

In summary, when comparing groups in the RL-induced policy, our results showed that for the MDP4 policy, the students in the High carry-out group significantly outperformed the students in the Low group, but no significant difference was found in the other three policies. This suggests that the MDP4 policy is effective in that the more it is carried out, the better it performs. However, the partially carry-out situation reduced the power of the MDP4 policy so that it did not significantly outperform the baseline random policy. When comparing the empirical evaluation results with theoretical evaluation results, PDIS is the best among the three IS-based metrics, reaching 100% agreement. Our results suggested that proper deployment settings have an impact on the performance of IS-based metrics. When transforming the deterministic policy to stochastic policy, soft-max is the best one, while Q-proportion is the worst, and Hard-code is stable. The comparison between the original reward and normalized reward indicates that the original reward can better reflect the empirical results despite having a larger variance.

3.8.2 Critical Decision Identification

Recall that each policy impacts its own critical decisions: those with higher differences between Q-values for possible decisions are considered to be critical, and we split each group of students according to whether the student received more decisions aligned with the critical decisions. Table 3.4 shows the t-test results comparing the post-test scores between the More vs. Less critical decisions groups. The first column indicates the MDP policies considered when identifying the critical decision. The second and third columns show the average post-test

scores of students in the More and Less groups, showing mean(sd). The fourth column shows the t-test results when comparing the post-test scores of the More and Less critical decisions groups. The MDP1 row shows a significant difference between the More vs. Less critical decisions groups for the MDP1 policy, with $t(448) = -1.97, p = .049$. However, the students in More group perform worse than the Less critical decisions group. The MDP2 and MDP3 rows show that there is no significant difference between the two critical decision groups in terms of post-test scores for the MDP2 or MDP3 policies. Finally, the MDP4 policy shows a significant difference between the two groups, $t(448) = 2.10, p = .036$, which means that students with More critical decisions performed significantly better than students with Less.

For the MDP4 policy, the identified critical decisions comprised 25% of all decisions. This shows that, although critical decisions are a small proportion of all decisions, they can significantly impact the outcome. Also, the results show that the Q-values in the MDP4 policy can be used to identify critical decisions aligned with empirical results, but the other three cannot. Based on the results from Table 3.1, MDP4 was identified as the only effective policy, since its empirical post-test results aligned, with students in the High carry-out group performing significantly better than the Low carry-out group. The critical decision results suggest that MDP4 is also the only policy where larger differences in Q-values had larger impacts on post-test results. Taken together, these results suggest that only Q-values in effective policies work to influence decisions that impact actual post-test performance. This further inspires us to investigate whether we could verify the effectiveness of a policy in reverse: Given a policy, if the decisions with larger Q-value difference are significantly linked to the student performance, then this policy may be more likely to be effective.

3.9 Conclusion

In this work, we explored three IS-based OPE metrics with two deployment settings in a real-world application. Through comparing the effectiveness of four RL-induced policies empirically and theoretically, our results showed that PDIS is the best one for interactive e-learning systems, and appropriate deployment settings (i.e., where policy decisions are carried out) are required to achieve reliable, robust evaluations. We also proposed a method to identify critical decisions by the Q-value differences in a policy. To verify our method, we investigated the relationship between the number of identified critical decisions and student post-test scores. The results revealed that the identified critical decisions are significantly linked to student learning, and further, that critical decisions can be identified by an effective policy but not by ineffective policies.

In summary, results from the OPE approach suggested that critical decisions can be identified by effective policy and Q-value difference. However, the offline analysis only considered the critical decisions with optimal actions, but the impact of critical decisions with sub-optimal actions and non-critical decisions was ignored. Therefore, we proposed an

Adversarial Deep Reinforcement Learning approach to identify critical decisions and evaluated through an empirical study in the next chapter.

CHAPTER 4

IDENTIFYING CRITICAL PEDAGOGICAL DECISIONS THROUGH ADVERSARIAL DEEP REINFORCEMENT LEARNING

Song Ju, Guojing Zhou, Hamoon Azizsoltani, Tiffany Barnes, Min Chi, Identifying Critical Pedagogical Decisions through Adversarial Deep Reinforcement Learning. EDM 2019: 595 – 598.

In this chapter, we explore an Adversarial Deep Reinforcement Learning (ADRL) based framework to identify critical decisions, and a Critical policy is induced by giving optimal actions on critical decisions but random actions on others. The effectiveness of the Critical policy was evaluated against a random yet reasonable (Random) policy in a classroom study. While no significant difference was found between the two conditions, we found that students often experience a consecutive sequence of critical decisions, and those who experienced more learned significantly better.

4.1 Introduction

During tutoring, the ITS makes a series of decisions to provide adaptive instructions. For example, in our ITS, the tutor makes more than 400 sequential decisions during training. Some of the decisions might be *more important and impactful than others*. In this work, we propose to induce compact RL policies that highlight key or critical decisions by taking advantage of the structure of the domain and the structure of our induced policies by leveraging the conditional independence relationships among the state features.

We propose Critical-RL, an adversarial deep reinforcement learning (ADRL) based framework to induce compact policies by making critical decisions. By inferring critical decisions we can identify which tutor actions are minimally necessary for the tutoring process to be effective, which holds great implications for systems research. Additionally, inference of critical relationships is one of the central tasks of science and it is one of the most challenging topics in many disciplines, particularly in the areas where controlled experiments are comparatively expensive or even impossible. In this work, we propose a general framework that fully integrates automatic critical inference and standard reinforcement learning in an ITS setting. We expect that our framework can be spread to other similar domains for related critical tasks.

Intuitively, we define critical decisions as those decisions that with them, the policy would be effective and without them, the policy would be ineffective. In order to reflect the double nature of critical decisions, we applied a deep RL approach Deep Q-Network (DQN) to induce a pair of adversarial pedagogical policies: an original policy that helps students learn and an inversed policy that hinders students learn. The policies decide whether to elicit the next step from the student or to directly show the student how to solve the next step. We refer to such decision as elicit/tell decision. For a given pair of adversarial policies and a decision-making point, whether the decision is critical is determined by comparing the two policies. More specifically, a decision is critical if the two policies select opposite actions and this decision is important for both policies. Please note that in our case, for each decision-making point, there are only two decisions available. Thus, if a decision-making point is critical and we know that one decision is good, then the other one must be bad. This leads to our first critical decision-making point identification rule: where the two policies make opposite decisions. Secondly, since the two policies have different goals, the importance of each decision may differ for them. However, if a decision-making point is really important, the decision made there should have a strong impact on both policies. This leads to our second critical decision-making point identification rule: where the decision is important for both policies.

In order to evaluate our critical decision identification model, we conducted an empirical classroom study with two conditions: *Critical* vs. *Random*. For the Critical condition, a partially carry-out setting was employed where critical decisions were made following the original policy while all other decisions were made randomly. For Random, all the decisions were made following a *random yet reasonable* policy. Since both elicit and tell are always considered to be reasonable educational interventions in our learning context, we refer to such a policy as a random yet reasonable policy or random. With this experiment design, the study would answer our research question that whether our critical decision identification model could effectively identify critical pedagogical decisions. In the experiment, 93 students were randomly assigned to the two conditions, where they were trained following the same general procedure, using the same materials and the same system. The only difference between the conditions was the Critical condition experienced additional critical decisions.

Our preliminary results showed that overall there's no significant difference between

the Critical and the Random condition in terms of student learning performance. A power calculation revealed that the sample size is not large enough to demonstrate a significant difference. However, much to our surprise, a close inspection into student-system interactive logs revealed that critical decisions often appeared consecutively and this appeared to be a "critical phase". More specifically, critical decision-making points often appeared consecutively and the decisions made at these points were the same. Since critical decision-making points are determined by students' learning state, the appearance of the critical phase also depends on students' learning experience in that some students may encounter more critical phases and others may encounter less. During tutoring, for those who encountered more critical phases, the system has more opportunities to intervene in their learning, and thus they are more likely to perform better than those who encountered less critical phases. Based on this observation, we divided students into High and Low critical phase groups where the former encountered more critical phase in training while the latter encountered less. Statistical analysis revealed that while there is no significant difference between the Critical and the Random group for Low students, the former significantly outperformed the latter for High ones. The explanation for the difference in High ones was that Critical policy gave full optimal guidance in the critical phase but Random policy gave poor guidance. In short, we revealed that critical pedagogical decisions should be treated as a period of decisions but not individual ones. More importantly, every decision in the critical phase is critical to the final outcome that poor choice of actions can significantly hurt student learning.

In sum, we make the following contributions:

- This is the first study focusing on critical pedagogical decision identification and evaluation.
- We proposed an Adversarial Deep Reinforcement Learning framework for critical pedagogical decisions identification.
- We discovered that when students encountered more critical phases, critical policy significantly helped them than random policy.

4.2 Related Work

4.2.1 Exploiting Q-value Difference

Some prior work exploited the Q-value difference between actions to simplify the decision-making process/problem in the context of ITS. For example, Mitchell et al. relied on the Q-value difference to reduce the feature space [Mit13]. More specifically, they proposed a policy evaluation metric, separation ratio for feature selection, which is defined as $\frac{2*|Q(s,a_1)-Q(s,a_2)|}{(Q(s,a_1)+Q(s,a_2))}$ where $Q(s,a_i)$ is the Q value for the state-action pair (s,a_i). The feature selection approach was then combined with RL to induce pedagogical policies for a dialog system, the Java tutor.

Zhou et al. [Zho17a] relied on Q-value difference to reduce the policy space. More specifically, they applied a weighted decision tree with post-pruning to extract a compact set of 529 rules from a full set of 3706 rules. During the extraction, each rule was weighted by the Q-value difference between two alternative actions and thus increased the carry-out likelihood of more important decisions. The policies were empirically evaluated in a classroom study. Results showed that the full RL policy and the compact DT policy together were significantly more effective than a random policy and there is no significant difference between the full RL policy and the compact DT policy.

In sum, prior studies have considered the Q-value difference between actions as a heuristic function of action importance. The larger the difference, the more important the decision is. However, prior work didn't quantitatively study how large Q-value difference is a critical decision. In this work, we explored the Q-value difference thresholds by classifying decisions into two categories: critical and non-critical and evaluating the quality of the critical decisions.

4.3 Method

Figure 4.1 shows an overview of our critical decision identification framework, which includes 2 stages: 1) adversarial policies induction and 2) critical decision determination. The first stage performs once for a given data set while the second stage performs once for each decision-making point. We describe each of the stages in the rest of this section.

4.3.1 Adversarial Reinforcement Learning

Adversarial Reinforcement Learning is a variation of RL which can be used to induce policies for multiple agents. The agents usually have conflict or mutually exclusive goals, which results in an adversarial relationship among them. The convention of RL goals still holds here in that they are defined by the rewards. During the policy induction procedure, each agent maintains its own learning process and may exchange information with other agents. However, each agent is given distinct rewards for the actions it takes, and oftentimes, the sum of the rewards is fixed. In other words, if one agent gets more, all others get less.

There are different ways to induce adversarial policies. One method is *Minimax Q-learning* which is specifically designed for zero-sum game in which two agents have opposite goals and share the same reward function [Uth03]. In Minimax Q-learning, there is only one single reward function, agent 1 tries to maximize its expected future reward while agent 2 tries to minimize. With a reward sign flip for agent 2: $r_2 = -r_1$, both agents aim to maximize their own rewards respectively and achieve opposite goals.

In our approach, we applied *reversed reward* to induce a pair of adversarial policies separately. An *Original Policy* is induced from the original reward while an *Inversed Policy* is induced from reversed reward, which the negative value of the original reward. Based on the Minimax Q-learning, reversed reward could guarantee that the original and inversed policies

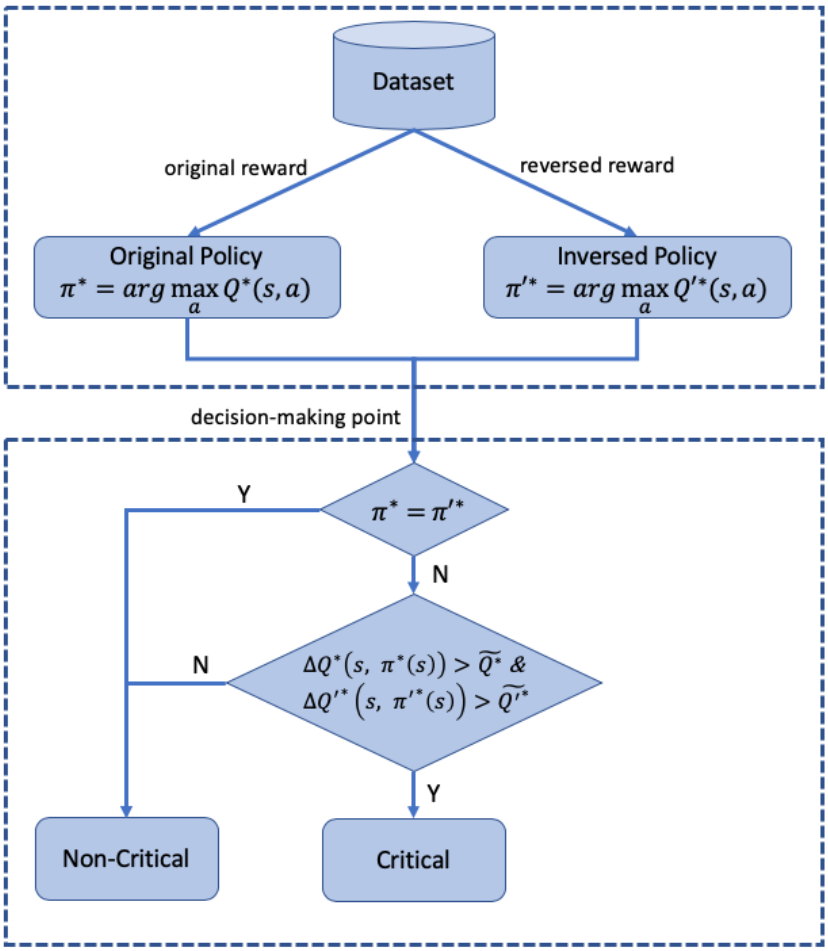


Figure 4.1: Critical Decision Identification Flow

have opposite goals that the former one tries to help student learning while the latter one tries to hinder.

4.3.2 Deep Reinforcement Learning

Much of prior work that applied RL to induce pedagogical policies modeled the decision-making problem as Markov Decision Processes (MDPs) whose solution is a set of optimal decision-making policies which can be found by RL algorithms. An MDP can be defined as a 4-tuple $\langle S, A, T, R \rangle$, where S is a set of state for the environment; A is a set of actions that the agent can take; T denotes the probability of transiting from state s to state s' by taking action a; and R denotes the expected reward of transiting from state s to state s' by taking action a.

Once $\langle S, A, R \rangle$ is defined, a student-ITS interactive log H can be seen as a trajectory:

$$H = s_1 \xrightarrow{a_1, r_1} s_2 \xrightarrow{a_2, r_2} s_3 \xrightarrow{a_3, r_3} \cdots s_L$$

Where $s_i \xrightarrow{a_i, r_i} s_{i+1}$ means that the tutor executed action a_i and received reward r_i in state s_i,

and then transferred to the next state s_{i+1}.

An optimal policy π^* to an MDP is a state-action mapping that specifies the best action to take at each state that would maximize the cumulative reward the agent will receive at the end. There are different ways to find the optimal policies, such as value iteration, policy iteration, temporal difference learning and Q-learning [Sut98]. Here, we adopted the one that has been widely used in combination with deep neural network, Q-learning. Q-learning reaches the optimal policy through finding the optimal action-value function $Q^*(s,a)$, which specifies the expected cumulative reward the agent will receiving if it takes action a in state s and follows the optimal policy to the end. The optimal action-value function $Q^*(s,a)$ can be obtained by iteratively updating the *Bellman equation* 4.1 until it converges.

$$Q_{i+1}(s,a) = \mathbb{E}_{s' \sim \varepsilon}[r + \gamma \max_{a'} Q_i(s',a') | s,a] \tag{4.1}$$

where γ is a discount factor, ε is the environment and Q_i is the action-value function at the ith iteration. Once have the optimal action-value function $Q^*(s,a)$, the optimal policy π^* can be easily generated using the following equation:

$$\pi^*(s) = \arg\max_a Q^*(s,a) \tag{4.2}$$

Deep Q-Network (DQN) As an instance of the general Q-leaning class, Deep Q-Network (DQN) finds the optimal action-value function through updating its action-value function approximator recursively following the Bellman equation as shown in equation 4.1. In DQN, a deep neural network is used as an action-value function approximator. The neural network takes a state as input, which is represented as a numerical vector and outputs its estimation of the Q values for all possible actions as shown in figure 4.2.

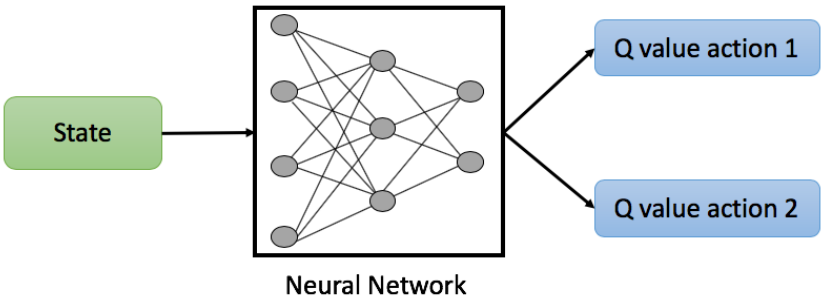


Figure 4.2: Example of DQN with one hidden layer

During training, DQN aims to minimize the expected difference between the target Q and

the predicted Q as shown in equation 4.3.

$$L_i(\theta_i) = \mathbb{E}_{s,a,r,s'}[(y_i - Q(s,a;\theta_i))^2] \tag{4.3}$$

where y_i denotes the target Q at the iteration i, $Q(s,a;\theta_i)$ denotes the predicted Q by the neural network at the iteration i. The target Q is estimated using the parameters from the previous iteration θ_{i-1}, as shown in the following equation:

$$y_i = r + \gamma \max_{a'} Q(s',a';\theta_{i-1}) \tag{4.4}$$

Different from supervised learning whose targets are fixed over the whole learning process, here the targets change in each iteration. More specifically, in each iteration, the target depends on the immediate reward the agent receives for an action and the estimation of the Q value for some states using previous parameters. Thus, the gradient of the loss function with respect to the neural network weights can be achieved by the equation below:

$$\nabla_{\theta_i} L_i(\theta_i) = \mathbb{E}_{s,a,r,s'}[(r + \gamma \max_{a'} Q(s',a';\theta_{i-1}) - Q(s,a;\theta_i))\nabla_{\theta_i} Q(s,a;\theta_i)] \tag{4.5}$$

With this loss function, we can simply apply standard methods like stochastic gradient and backpropagation to update the weights of the neural network.

Overall, DQN is a *model-free* approach that it is focused on estimating the action value functions from the interactions with the environment without constructing a model of the environment. Also, it is an off-policy approach that the new policy is induced based upon the historical data generated by an alternative behavior policy.

4.3.3 Identifying Critical Decision

In this subsection, we describe the critical decision identification procedure, which relies on a pair of adversarial policies: an original policy and an inversed policy. Intuitively, critical decisions should have a strong impact on the success of the task in that the right decision brings great potential value while the wrong decision results in a great loss of potential value. Here we defined two rules for critical decision identification: 1) the two policies take opposite actions at the decision-making point and 2) the decision is important for both policies.

Regarding the first rule, for a given state, the two policies are queried for decisions. If the two policies take the same decision, we consider it as non-critical; otherwise, the second rule will be tested. In order to measure the importance of the decision for each policy, we calculate the Q-value difference between the two alternative actions: $\Delta Q^*(s) = |Q^*(s,a_1) - Q^*(s,a_2)|$. If this difference is greater than a threshold, the decision is considered important for the corresponding policy. There are many possible ways to determine the threshold. In our case , we set the threshold to be the median Q-value difference for all decision-making points in the training dataset.

4.4 Policy Induction

In this section, we first describe our training corpus and then how we applied our ADRL framework to induce critical decision identification models.

4.4.1 Training Corpus

Our training corpus contains a total of 849 students' interaction logs collected from seven classroom studies where students were trained on our ITS following random policies. The studies were given as a regular homework assignment to students. During the studies, all students used the same ITS, followed the same general procedure, studied the same training materials, and worked through the same training problems. The general procedure of the studies was identical to the one where our ADRL framework was evaluated, which will be described in section 4.5. The training corpus provides us with the state representation, action, and reward information for policy induction.

State representation is one of the key factors that determine the effectiveness of the induced policies, as with many machine learning tasks. During tutoring, There are many factors that might determine or indicate students' learning state, but many of them are not well understood. Thus, to be conservative, we extracted varieties of features that might impact student learning from student-system interaction logs. In sum, 142 state features were extracted which can be categorized into the following five groups:

1. **Autonomy (AM):** the amount of work done by the student: such as the number of elicit steps the student received so far *elicitCount* or the number of hints requested *hintCount*.

2. **Temporal Situation (TS):** the time-related information about the work process: such as the average time taken per problem *avgTime*, or the total time spent solving a problem *TotalPSTime*.

3. **Problem Solving (PS):** information about the current problem solving context, such as the diffculty of the current problem *probDiff*, or whether the student changes the difficulty level *NewLevel*.

4. **Performance (PM):** information about the student's performance during problem solving: such as the number of the right application of rules *RightApp*.

5. **Student Action (SA):** the statistical measurement of student's behavior: such as the number of non-empty-click actions that students take *actionCount*, or the number of clicks for derivation *AppCount*.

Action Our ITS makes the elicit/tell decision at each step in a problem. In elicit, the tutor elicits the solution from the student while in tell, the tutor directly shows the solution. These are the two decisions available to the agent at each decision-making point.

Reward The rewards we used for policy induction were the inferred "immediate rewards". In RL, rewards serve as the feedback for the agent's action. In general, immediate rewards are more effective than delayed rewards because the former ties the feedback to single actions which allows the agent to precisely credit or blame them. Delayed rewards, in contrast, mix the feedback of multiple actions and thus make it hard to credit or blame. On the other hand, the most appropriate reward for policy induction is student learning gains which are typically unavailable until the student completes the whole training process. This is because of the complex nature of human learning, which makes it impractical to evaluate student learning moment by moment. Additionally, some instructional strategy that boosts short-term performance may not be effective for long-term learning gain. In order to deal with the reward challenge, we rely on Gaussian Process to infer the immediate rewards from delayed rewards. The delayed rewards were students' *normalized learning gain (NLG)* which measures their learning gain *irrespective of their incoming competence*. NLG is defined as $NLG = \frac{posttest - pretest}{1 - pretest}$ where 1 is maximum score for both pre- and post-test.

4.4.2 Critical Decision Identification Models

This subsection describes how we applied our proposed ADRL framework to build a critical decision identification model. More specifically, it describes 1) how we applied DQN to induce a pair of adversarial policies for our ITS and 2) how we determined the critical Q-value difference threshold for each policy.

In tutoring, our ITS provides students with the same 12 problems in the same order. Among them, the first and the eighth problems are fixed to be problem solving where the students are required to solve all the steps and the step presentation of the rest 10 is determined by the tutor. Thus, we built 10 critical decision identification models, one for each problem. Each of the models consists of two policies: an original policy and an inversed policy. The original policy was induced using the original rewards while the inversed policy was induced using inversed rewards. An inversed reward for action was the negative value of the original reward for such action. Other than the rewards, all other parts of the data were identical, such as state representation and transition samples.

For each of the problems, we fit into the DQN framework with two different types of neural networks: Recurrent Neural Network (RNN) and Long Short Term Memory (LSTM) to induce two sets of policies and select the better one between them. Per decision importance sampling (PDIS) was used to determine which one is better. PDIS is an advanced off-policy policy evaluation method that allows us to evaluate the effectiveness of policy using pre-collected historical data, without the need to conduct the actual online evaluation. For a given policy and historical data, PDIS returns the expected cumulative rewards the agent will receive following the policy. Here, we consider a model good if both of the policies in them are better than those in the other model. More specifically, an original policy is better than the other one if the former has a higher PDIS value; and an inversed policy is better than the other if the

former has a lower PDIS value.

Table 4.1 shows a comparison of the PDIS value for the Random, Original, and Inversed policy for each problem. The first two rows show the problem and the neural network selected for the corresponding problem. The next three rows show the result for the Random, Original, and Inversed policy respectively. The results suggest that for each problem, the original policy was more effective than the random policy while the inversed policy was less effective than the random policy. Thus, by performing neural network selection, we expect that a better model would be selected for each problem.

Once the models are built, whether a decision is critical or not during tutoring can be determined by comparing and examining the two policies in it. As mentioned in section 4.3.3, the decision made by the two policies were first compared, and if they disagree, we further examine the Q-value difference to determine whether the decisions are important for each of the policies. The importance threshold for each of the policies is set to be the median Q-value difference in the training data. More specifically, we applied the induced policies to our training dataset, calculated the Q-value difference for every decision-making point, and determined the median threshold.

Table 4.1: PDIS of Induced Policies in Training Dataset

	P2	**P3**	**P4**	**P5**	**P6**	**P7**	**P9**	**P10**	**P11**	**P12**
DQN Neural Network	RNN	LSTM	LSTM	LSTM	LSTM	LSTM	LSTM	RNN	RNN	LSTM
Random Policy	0.022	-0.018	-0.006	0.008	0.01	0.012	0.002	-0.008	-0.007	-0.006
Original Policy	0.29	0.058	0.052	0.618	0.091	0.053	0.015	1.9	0.017	0.02
Inversed Policy	-0.003	-0.057	-0.083	0.004	0.006	0.01	-0.00002	-1.39	-0.02	-0.54

4.5 Experiment Setup

In order to evaluate our critical pedagogical decision identification approach, we conducted a classroom study comparing the critical policy with the random policy. This section describes the setup of the study.

4.5.1 Participants and Conditions

The participants of this study were undergraduate students enrolled in the Discrete Mathematics class in the 2018 Fall. This study was assigned to students as one of their regular homework assignments. Completion of the study was required for full credit and students were told that they will be graded based on their demonstrated effort, not their learning performance.

In the study, the Critical condition and the Random condition were compared. In the Critical condition, critical decisions are made following the good pedagogical while other

decisions are made randomly. More specifically, at each decision-making point, the critical decision identification model was queried to determine whether the decision was critical. If it was, the decision made by the original policy in the model will be taken; otherwise, the decision will be taken randomly.

120 students were randomly assigned to the Critical Condition and the Random condition. Due to preparation for final exams and the length of the study, 96 students completed the study. 3 students who performed perfectly in the pre-test were excluded from our subsequent statistical analysis. The final group sizes were: N = 50 (Critical) and N = 43 (Random). We performed a Chi^2 test of the relationship between students' condition and their completion rate and found no significant difference between the conditions: $Chi^2(1) = 2.55$, $p = 0.11$.

4.5.2 Pyrenees Tutor

Pyrenees tutor is a web-based ITS for probability. It covers 10 major principles of probability, such as the Complement Theorem and Bayes' Rule. Pyrenees tutor provides step-by-step instruction and immediate feedback. Pyrenees tutor can also provide on-demand hits prompting the student with what they should do next. As with other systems, help in Pyrenees tutor is provided via a sequence of increasingly specific hints. The last hint in the sequence, the bottom-out hint, tells the student exactly what to do. For the purposes of this study, we incorporated two distinct pedagogical decision modes into Pyrenees tutor to match the two conditions.

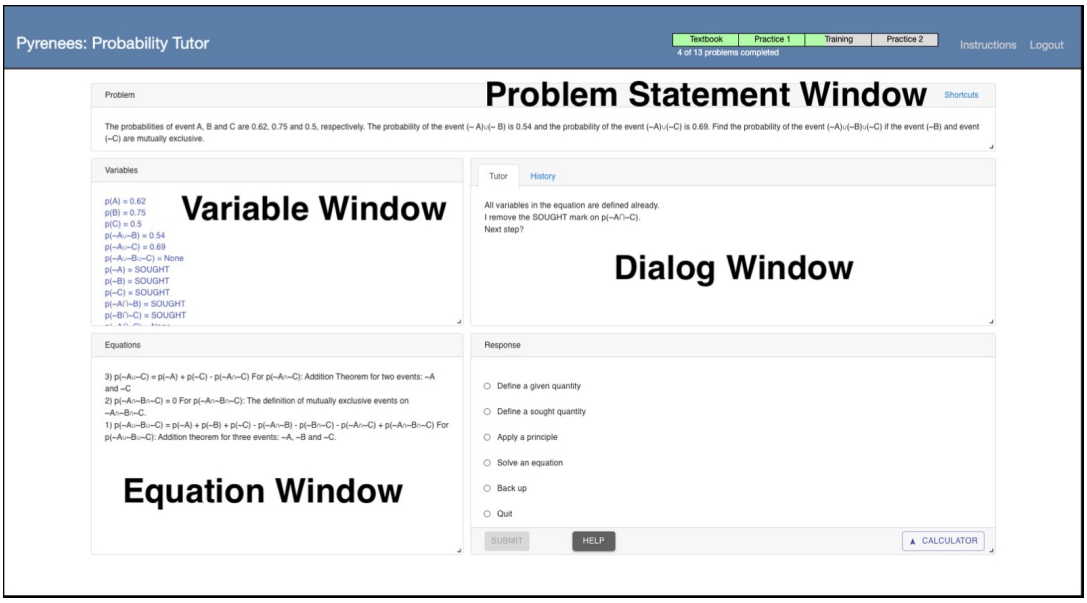


Figure 4.3: The Interface of the Pyrenees Tutor

Figure 4.3 shows the interface of Pyrenees, which consists of multiple windows. The upper half of the dialog window shows the message the tutor provides to the students, such as the explanation for a tell step, or the prompt for an elicit step. In the lower half, the student enters responses such as writing an equation or selecting a choice. Any variables or equations generated through this process are shown on the left side of the screen for reference.

4.5.3 Experiment Procedure

In this experiment, students were required to complete 4 phases: 1) pre-training, 2) pre-test, 3) training on Pyrenees tutor, and 4) post-test. During the pre-training phased, all students studied the domain principles through a probability textbook, reviewed some examples, and solved certain training problems. The students then took a pre-test which contained 14 problems. The textbook was not available at this phase and students were not given feedback on their answers, nor were they allowed to go back to earlier questions. This was also true of the post-test.

During phase 3, students in all conditions received the same 12 rather complicated problems in the same order on Pyrenees tutor. Each main domain principle was applied at least twice. The minimal number of steps needed to solve each training problem ranged from 20 to 50. These steps included defining variables, applying principles, and solving equations. The number of domain principles required to solve each problem ranged from 3 to 11. All of the students could access the corresponding pre-training textbook during this phase. Each step in the problems could have been provided as either a tell or elicit based upon the condition policy. Finally, all of the students completed a post-test with 20 problems. 14 of the problems were isomorphic to the pre-test given in phase 2. The remaining six were non-isomorphic complicated problems.

4.5.4 Grading Criteria

The 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 are 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 $[0, 1]$.

4.6 Results

This section shows the empirical evaluation results. First, we report the comparison between the Critical and the Random condition. Then we report the results where students were split into High and Low critical decision groups based on the number of critical decisions they received. Finally, we report the results where students were split into High and Low critical phase groups based on the number of critical phases they experienced.

4.6.1 Critical vs. Random

Table 4.2: Critical vs. Random

Measure	Critical (50)	Random (43)	Contrast Comparison Result
Pre	0.653 (0.17)	0.681 (0.19)	$t(91) = -0.750, p = 0.455$
Iso Post	0.819 (0.19)	0.819 (0.19)	$t(91) = 0.002, p = 0.999$
Post	0.706 (0.19)	0.711 (0.20)	$t(91) = -0.119, p = 0.905$
Learning Gain	0.053 (0.18)	0.030 (0.14)	$t(91) = 0.681, p = 0.498$
Time	121.6 (37.7)	116.257 (30.47)	$t(91) = 0.744, p = 0.459$

Table 4.2 showed the contrast comparison results between the Critical and the Random condition. The parenthesized values following the condition names in row 1 denoted the number of students in each condition. Rows 2-4 showed a comparison of pre-test scores, post-test scores, learning gain, and training time between the two conditions along with mean and SD for each. Contrast Comparison between the Critical and the Random condition on pre-test score shows no significant difference: $t(91) = -0.750, p = 0.455$. Thus, the two conditions were balanced in terms of incoming competence. In order to examine how students learned on the Pyrenees tutor, we conducted a comparison between the scores on pre-test and isomorphic post-test questions. A repeated measures analysis using test-type (pre-test and isomorphic post-test) as factors and test score as dependent measure showed a main effect for test type: $F(1, 91) = 91.43$, $p < 0.0001$. Further comparisons on group by group basis showed that both Critical and Random condition scored significantly higher on isomorphic questions in post-test than in pre-test: $F(1, 49) = 48.77$, $p < 0.0001$ for Critical and $F(1, 42) = 43.82$, $p < 0.0001$ for Random. It revealed that the Pyrenees tutor indeed improved students' learning regardless of the pedagogical policies deployed.

Contrast comparison analysis on the post-test score, learning gain and total training time revealed no significant difference between the two conditions. Although it appears that the Critical condition outperformed the Random condition on learning gain 0.053 vs. 0.03, such difference was not significant p = 0.498. One of the possible reasons is that the group size was not large enough to exhibit significance. A post hoc power analysis revealed that a total

sample of 1544 students was required to detect significance at .05 on small effects (d=.14), with 80% power using a contrast.

Furthermore, there are only two types of actions (elicit or tell) so that the random policy has a great chance to execute the optimal one. It means that Random policy could have similar power with the Critical policy. In the Critical condition, the percentage of critical decisions is 14%, 2797 out of 19827. Through running the Critical policy script on the raw log data of the random condition, we found that the percentage of critical decisions in the random condition is 12%, 2039 out of 16731. Among the 2039 critical decisions, 1030 critical decisions have the same action as the Critical policy. It revealed that the Random policy has partial power over the Critical-RL policy.

Since whether a decision is critical or not is determined by the students' learning state, students who encountered critical states more often are more likely to be affected by the pedagogical policy. In order to examine how critical decisions may affect student learning, we split students into High and Low critical decision groups.

4.6.2 From Critical Decision to Critical Phase

We divided the Critical condition students into two groups: High (n=26) and Low (n=24) based on the median split on the number of critical decisions. The High students experienced more critical decisions than the Low students. A contrast comparison was performed between these two groups in terms of student learning gain. The result showed that there's no significant difference in student learning gain: $t(48) = -1.117, p = 0.270$ with $M = 0.027, SD = 0.19$ for High vs. $M = -0.082, SD = 0.17$ for Low.

A deep inspection was conducted on the student-system interactive logs. We found that critical decisions always appeared in groups and each group of consecutive critical decisions had the same action. For example, below shows a sequence of decisions in the log:

$$c_{elicit} \rightarrow c_{elicit} \rightarrow c_{elicit} \rightarrow r \rightarrow r \rightarrow c_{tell} \rightarrow c_{tell} \rightarrow c_{tell}$$

where r is a non-critical decision made randomly, c_{elicit} is a critical decision executes elicit, c_{tell} is a critical decision executes tell. We found that it's very common to have continuous c_{elicit} or c_{tell} in the Critical condition. This is aligned with the existing learning theory that the learning process is a continuous process. Students can stay in the same learning state during several steps but this continuous learning state is hard to be represented by current features. In other words, in the same learning state, the agent will continuously give the same actions to the student until he moves to the next learning state. So, we define *Critical Phase* as a period of consecutive critical decision executions with the same action according to the Critical policy.

In the Critical condition, we analyzed the relationship between critical decision and critical phase. First, there were 2436 out of 2933 (83.1%) critical decisions in the critical phase. It indicated that most of the critical decisions occurred consecutively. Second, a Pearson correlation test between the number of critical decisions and the number of critical phases in

each student showed a significant positive correlation: $r = 0.7, df = 48, p < 0.001$. Moreover, we run the Critical policy script on the Random condition to investigate the critical phase even if the optimal action in the critical phase was not fully carried out. Our results showed that 2049 out of 2509 (81.7%) critical decisions were in the critical phase in the Random condition. We also found a significant correlation between the number of critical decisions and the number of critical phases in the Random condition: $r = 0.57, df = 41, p < 0.001$. It revealed that the critical phase did exist and was highly correlated with critical decisions. Next, we will analyze the impact of the critical phase on students' learning performance.

4.6.3 High vs. Low in Critical Phase

In this section, we first counted the number of critical phases for each student in both Critical and Random conditions. If a period of consecutive critical decisions had the same action as Critical policy decision, then this period counted as one critical phase. For random condition, as the execution of critical decisions were partially agreed with the Critical policy, we ignored the actual decision and only focused on Critical policy's decision. Then, we divided students into High vs. Low conditions based upon the median split on the number of critical phases. Thus, we have four groups based upon their critical phase and policies: High-Random (n=20), Low-Random (n=23), High-Critical (n=27), Low-Critical (n=23).

We found significant difference on the learning gain between High-Critical and Low-Critical groups: $t(48) = 2.187, p = 0.031$: $M = 0.098, SD = 0.2$ for High-Critical vs. $M = 0.0008, SD = 0.13$ for Low-Critical. It indicated that when using critical policy, students who experienced more critical phases significantly outperformed those who experienced less. In other words, the more help students got in critical phases, the more improvement the students gain.

A two-way ANOVA using policies {Critical, Random} and critical phase {High, Low} as two factors and the student's learning gain and training time as the dependent measure showed a significant interaction effect $F(1, 89) = 7.163, p = 0.009$ on learning gain, as shown in Figure 4.4. It revealed the critical policy indeed affects student learning. Also, there's a significant difference between the High-Critical and High-Random on learning gain: $t(89) = 2.360, p = 0.02$ with $M = 0.098, SD = 0.2$ for High-Critical vs. $M = -0.011, SD = 0.16$ for High-Random. There's no significant difference between Low-Critical and Low-Random in terms of learning gain. It indicated that when students experienced more critical phases, critical policy significantly helped student learning than random policy.

Additionally, we found a significant main effect from the critical phase: $F(1, 89) = 5.579, p = 0.020$ for the training time. The High condition students spent significantly more time on the training than the Low condition: $M = 127.51, SD = 36.34$ for High condition while $M = 110.56, SD = 30.51$ for Low condition. It indicated that the critical phase may be a phase that the student is struggling with a problem or understanding a concept. It is a moment that students are learning something unfamiliar and time-consuming. Thus, it is reasonable that

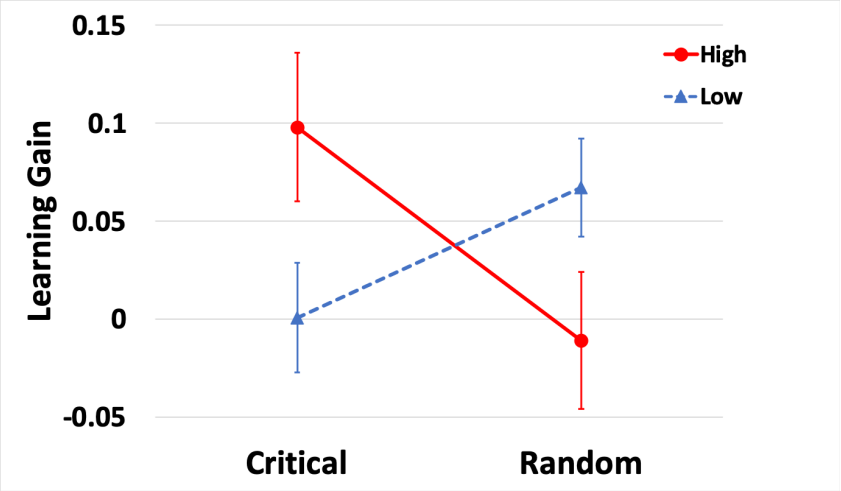


Figure 4.4: Comparison of Learning Performance

High students would perform badly when they are learning something critical and the tutor did not give the correct guidance.

4.6.4 Log Analysis

In this subsection, we investigated the key features in the critical state. In other words, which features can best distinguish critical states from non-critical ones. We applied a gradient boosting tree classifier on Critical-RL log data with 142 features as input X and {critical phase, non-critical phase} as the class label. We trained a classification model and evaluated it by 5 cross-validation. The higher weights of features in the tree classifier indicated more contribution to the classification accuracy. With an accuracy 87.8% in 5 cross-validation, the top three features ordered by weights were *timeSinceLastWrongStepKC*, *stepsSinceLastHint*, *timeOnCurrentProblem*. timeSinceLastWrongStepKC means the time elapsed since the last time student makes a mistake. stepsSinceLastHint means from the last hint request, how many steps the student has done. timeOnCurrentProblem means the time spent on the current problem so far. Much to our surprise, the top three features all belong to the category *Temporal Situation (TS)* described before. It revealed that temporal features are a strong indicator to distinguish student learning state.

4.7 Discuss

Through training real students on our ITS with the Critical policy, we found that critical decision-making points often appeared consecutively and the decisions made at those points were the same. This situation couldn't be observed in the training dataset which was collected via training students with random policies because the critical decisions were not always made following some effective policy. This critical phase is like a critical thinking phase in the

learning process, in which students are learning something new or solving a problem. For example, students may come across a new concept that they never know before. They need time to understand this new concept and store it in the personal knowledge system. Another example is students are struggling in solving a problem. In this case, they need time to figure out the next steps and make sure it's correct. In addition, some students may encounter critical phases more often than others, which makes them more likely to be affected by pedagogical policies. Our empirical results supported this point showing that for those who encountered the critical phase more often, the Critical policy was significantly more effective than the Random policy, but for those who encountered it less often, there was no significant difference between the two policies.

In our ITS, each student needs more than 400 steps to finish the training process. However, not all decisions are equally important. In Critical condition, our critical decisions identification model identified 55 critical decisions for each student on average. It dramatically reduces the size of decision space and is more conducive to educators to analyze student learning. With respect to the critical phase, the number further reduced to 18 for each student on average. This makes it possible to analyze when is the monumental points, what characters do they have, what concepts or problems often cause critical thinking. It gives us the ability to manually inspect what happened to students in the training process on ITS.

4.8 Conclusion

In this study, we proposed an Adversarial Deep Reinforcement Learning framework to identify critical pedagogical decisions in an ITS. Based on this framework, we implemented a critical policy that gives optimal actions on critical decision points but randomly selects actions on non-critical decision points. We empirically compared the critical policy with a baseline random policy in a classroom study for real students. We found that critical decisions were likely to appear in groups and the consecutive ones had the same decision. We refer to this period of consecutive critical decisions as the critical phase. Statistical analysis revealed that students who encountered more critical phases learned significantly better with the critical policy than with the random policy. However, for those who encountered less critical phases, there was no significant difference between the two policies.

In summary, results from the ADRL classroom study showed that the Critical policy didn't outperform the Random policy, which means the identified critical decisions didn't have enough impact on the desired outcome. Therefore, we re-examined the decision-making process in human and animal behaviors, and then proposed a novel Long-Short Term Reward framework.

CHAPTER 5

LONG-SHORT TERM REWARD (LSTM) FRAMEWORK

Publish: Song Ju, Guojing Zhou, Tiffany Barnes, Min Chi, Pick the Moment: Identifying Critical Pedagogical Decisions Using Long-Short Term Rewards. EDM 2020: 126-136

In this chapter, we investigated the animal and human decision-making behavior studies and proposed a novel RL-based Long-Short Term Rewards (LSTR) framework for critical decision identification. For RL policy induction, we modified the Bellman equation to consider critical decisions and proposed a Critical Deep Q-Network (Critical-DQN) algorithm. Before applying to real-world applications, we evaluated the effectiveness of the LSTR framework on an ideal GridWorld game and a real-world ITS (Pyrenees) dataset. The results showed that it indeed identifies critical decisions in the sequences and only carrying out critical decisions alone is as effective as a fully executed policy. Overall, we found that in order to identify critical decisions, we need to separate *critical states* from critical decisions. More specifically, there are two factors in identifying critical decisions: 1) identify critical states and 2) select optimal actions.

5.1 Introduction

Recent advances in computational neuroscience have enabled researchers to simulate and study the decision-making mechanisms of humans and animals through computational approaches [Mor06; Roe07; Sul11; Li11; McC04]. A number of works showed that RL-like learning and decision-making processes exist in humans/animals and we humans use immediate reward and Q-value to make decisions [Li11; McC04].

Motivated by research in human and animal behaviors, we propose Long-Short Term

Rewards (LSTR) framework to identify *critical* decisions based on RL-induced policy. For policy induction, we propose a Critical Deep Q-Networks (Critical-DQN) algorithm to consider the critical decisions in the training loop. More specifically, critical decisions should be the moments where optimal actions have to be made for the desired outcomes. To quantify their impacts, we define critical policy as the one which will carry out the optimal actions in the critical states while taking random actions in the non-critical states. To identify critical states, we propose to use both RL-induced policy's action-value functions (long-term) and immediate rewards (short-term). The effectiveness of the proposed LSTR framework is evaluated on an ideal GridWorld game and a Pyrenees historical dataset. Our results show that the proposed LSTR framework indeed identifies critical decisions and moreover, carrying out the critical decisions alone is as effective as a fully executed policy. Our main contributions can be summarized as follows: 1) we proposed the Long Short Term Rewards framework to identify critical decisions and evaluated on an ideal GridWorld game and a real-world ITS dataset. 2) we proposed Critical-DQN to improve the long-term rewards regarding identifying critical decisions and investigated its advantages and disadvantages.

5.2 Method

We follow the conventional Reinforcement Learning (RL) notation. An agent interacts with an environment over a series of decision-making steps. The environment is framed as a Markov Decision Process (MDP). At each timestep t, the agent observes the state the environment is in, denoted s_t; then the agent chooses an action from a discrete set of possible actions: $A \in (a_1, a_2, ..., a_n)$. As a result, the environment provides a scalar *immediate reward* r. We assume that the future rewards are discounted by the factor $\gamma \in (0, 1]$, and the agent's goal is to maximize the expected discounted sum of future rewards, also known as the return. The return at time-step t is defined as $R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$, where T is the last time-step in the episode.

The goal of the agent is to find the optimal action-value function $Q^*(s, a)$, which will result in the agent receiving the highest possible expected return, starting from state s, taking action a, and following the optimal policy π^* thereafter. Formally, we define the optimal action-value function as $Q^*(s, a) = \max_\pi \mathbb{E}[R_t | s_t = s, a_t = a, \pi]$. The optimal action-value function must follow the *Bellman Equation* shown in Equation 5.1, which states that the Q-value for a given state and action should be equal to the immediate reward obtained after taking that action, plus the discounted Q-value of the optimal action a' taken from the next state s'. Note that this is an expectation over the next states sampled from the environment.

$$Q^*(s, a) = \mathop{\mathbb{E}}_{s' \sim \mathcal{E}} [r + \gamma \max_{a'} Q^*(s', a') | s, a] \tag{5.1}$$

In our case, we follow the batch Reinforcement Learning formulation in that we have a fixed-size dataset $\mathcal{D}$ consisting of all historical sample episodes and each episode is denoted

as $s_1 \xrightarrow{a_1,r_1} s_2 \xrightarrow{a_2,r_2} s_3 \xrightarrow{a_3,r_3} \cdots s_L$). To make this task more general, we assume that the state distribution and behavior policy that was used to collect this data are both unknown.

In the following, we will describe the two ways of defining critical decisions: long-term reward vs. long-short term reward and the two DRL algorithms explored: original DQN and Critical DQN. Based on two types of DRL methods and two types of roles: identify critical state and decide optimal action, we will compare four different policies.

5.2.1 Two Types of Critical Rewards

5.2.1.1 Long Term Rewards (LongTRs)

In RL, $Q(s, a)$ is defined as the expected rewards the agent will receive by taking action a at state s and following the policy to the end. Intuitively, if all the actions for a given state have the same Q-value, then it does not matter which action should be taken because all the actions will lead to the same final reward. But if the difference of Q-values among different actions is large, then taking a wrong action can result in a significant loss in the final reward. So, we define ***the Long Term Reward (LongTR)*** as the difference between the cumulative future rewards of the best action and that of the worst action:

$$LongTR(s) = \max_a Q(s, a) - \min_a Q(s, a) \tag{5.2}$$

which is the difference between the maximum and minimum Q-values in the state s. In general, the higher the LongTR, the more important the decision is.

5.2.1.2 Long-Short Term Rewards (LSTRs)

Besides cumulative future rewards, we argue that it is also important to consider the immediate rewards because if any action for a given state can lead to a large positive or very negative immediate reward, such a state can also be important. In the following, we refer to the immediate rewards as ***the Short Term Reward (ShortTR)***. On the other hand, in many real-world applications like healthcare, the rewards are often delayed until the end of the trajectory. Different from the delayed rewards in the classic mouse-in-the-maze situations where agents receive insignificant rewards along the path and a significant reward in the final goal state (the food), in healthcare, there are immediate rewards along the way but they are often *unobservable*. Therefore, the challenge is how to infer these unobservable, immediate rewards from the delayed rewards, while taking the noise and uncertainty in the data into account. In this work, we apply a neural network based approach (InferNet) to infer "immediate" rewards from delayed rewards.

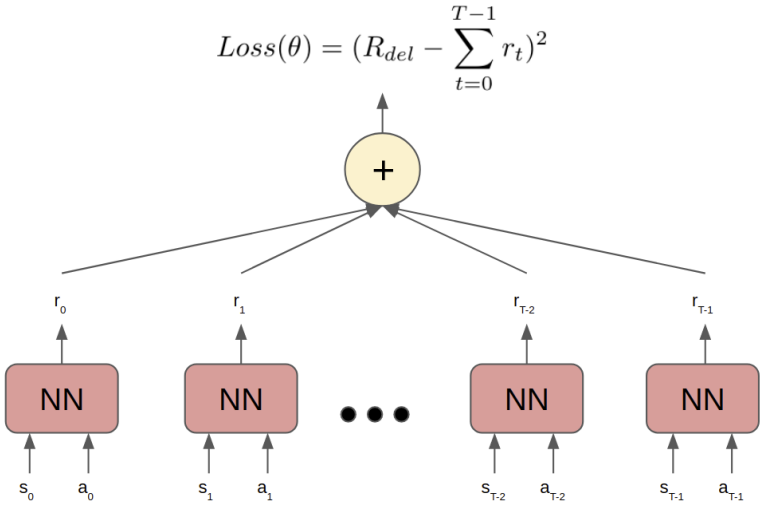


Figure 5.1: InferNet Architecture

5.2.1.3 InferNet

InferNet is designed and implemented by my labmate Markel Sanz Ausin. The intuition behind InferNet is rather straightforward. InferNet uses a deep neural network to infer the immediate rewards from the delayed reward in an episode. In Figure 5.1 at each timestep, the observed state and action are passed as input to the neural network, which will output a single scalar, the inferred immediate reward for that state and action: $r_t = f(s_t, a_t|\theta)$. Here θ indicates the parameters (weights and biases) of the neural network. The constraint on the predicted rewards is: the sum of all the predicted rewards in one episode should be equal to the delayed reward. Therefore, our loss function is defined as $Loss(\theta) = (R_{del} - \sum_{t=0}^{T-1} r_t)^2$, which is the difference between the sum of predicted rewards and the delayed reward. This way, we train InferNet by minimizing the loss function and then use it to predict the immediate reward for each state-action pair.

5.2.2 Two Types of Deep RL Policy

5.2.2.1 Original DQN

Deep Q-Network (DQN) is one of the most promising approaches, which is widely used in areas like robotics and video games [Mni15b]. Fundamentally, DQN is a version of Q-learning which uses neural networks to approximate the Q-values of the different state-action couples. In order to train the DQN algorithm, the two neural networks with equal architectures are employed: one for calculating the Q-value of the current state and action: $Q(s, a)$ and another neural network to calculate the Q-value of the next state and action: $Q(s', a')$. The former is the main network and its weights are denoted by $\boldsymbol{\theta}$ and the latter is the target network, and its weights are denoted by $\boldsymbol{\theta}^-$. The *Bellman Equation* for DQN is shown in Equation 5.3 and it is

trained through running a gradient descent algorithm to minimize the squared difference of the two sides of the equality.

$$Q(s,a;\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{s'\sim\mathcal{E}}[r + \gamma \max_{a'} Q(s',a';\boldsymbol{\theta}^{-})] \tag{5.3}$$

The main network is trained on every training iteration, while the target network is frozen for a number of training iterations. Every k training iterations, the weights of the main neural network are copied into the target network. This is one of the techniques used in order to avoid divergence during the training process. In practice, DQN also uses an experience replay buffer to store the recently collected data and to uniformly sample (s,a,r,s') steps from it. By sampling uniformly, it breaks the correlations between samples of the same episode, making the learning process more robust and stable. In this work, as we are doing batch RL, our whole dataset will be the experience replay buffer, and it will not change during the training process.

Basically, DQN is a Q-learning method that finds the optimal action-value function by updating its action-value function approximator recursively. Its major difference from the traditional RL is that a deep neural network is used as an action-value function approximator and this allows it to deal with the tasks with high dimensional state space.

5.2.2.2 Critical DQN

In the original DQN, the Q functions are estimated based on the assumption that the optimal policy will be followed to the end. We define *critical policy to be the one that the optimal decision will be carried out in critical decision points while random decisions in the rest*. By not taking the optimal actions on non-critical decisions, we fundamentally change the dynamics of the Bellman equation which assumed full execution of the policy. Therefore we need to modify it so that it can incorporate our critical decisions into consideration.

For a single (s,a,r,s') tuple, the original Bellman Equation can be expressed as:

$$Q(s,a) = r + \gamma \max_{a'} Q(s',a') \tag{5.4}$$

where r is the immediate reward for taking action a at state s; γ is the discount factor; and $Q(s',a')$ is the action-value function for taking action a' at the subsequent state s'.

To induce a critical policy, we modify the original Bellman equation based on whether a decision is critical or not. The intuition behind the Critical-DQN is that if a decision is important, then the agent should take the best action otherwise the agent can randomly choose an action to take. Therefore, we have:

$$Q(s,a) = \begin{cases} r + \gamma max Q(s',a') & \text{s' is critical} \\ r + \gamma mean Q(s',a') & \text{s' is non-critical} \end{cases} \tag{5.5}$$

In equation 5.5, to update the Q-value for any given s and a, it will consider whether the next

state s' is critical or not. If it is critical, the maximum Q-value for s' will be used to update Q(s, a); if the decision s' is non-critical, then the average Q-value among all the actions on s' will be used to update Q(s, a).

Algorithm 1 presents the pseudo-code for the Critical-DQN. First of all, the immediate rewards in the training dataset are inferred from the delayed rewards by the InferNet model. Then, there are three parameters in the algorithm: $T^{+}_{ShortTR}$ and $T^{-}_{ShortTR}$ are the ShortTR thresholds originated from the elbows of the inferred immediate reward distribution and, T_{LongTR} is the LongTR threshold used to determine the Q-value difference threshold.

In the algorithm, it first applies the InferNet model to predict the maximum and minimum inferred immediate rewards for the next state s'. If the maximum is larger than $T^{+}_{ShortTR}$ or the minimum is smaller than $T^{-}_{ShortTR}$, the next state s' is critical and a label c_i is added to the tuple. Second, it initializes all Q-values using the inferred immediate rewards to avoid the bias of the neural network. Then in the main training loop, for each iteration, the algorithm first calculates the Q-value difference for all the states. The Q-value difference threshold T_{Δ} is defined as the top T_{LongTR} percent value in the training dataset. It means that T_{LongTR} percent states with higher Q-value difference are critical. Finally, for each (s, a, r, s', c') tuple, if the Q-value difference of s' is larger than T_{Δ} or it is identified as critical by the immediate rewards $c' == True$, we consider the state s' as critical and its value function is $max_{a'}Q(s', a'; \theta^{-})$; for non-critical states, their value function are defined as $mean_{a'}Q(s', a'; \theta^{-})$.

In summary, the algorithm applies ShortTR to identify one set of critical states and LongTR to identify another set of critical states. The final critical states are the union of the two sets. More specifically, the set of ShortTR is static because the thresholds $T_{ShortTR}$ and InferNet are pre-defined before training. But the set of LongTR is dynamic and determined by the RL policy and threshold T_{LongTR}.

5.2.3 Identifying & Evaluating Critical Decision

The effectiveness of our LSTR framework on identifying critical decisions is evaluated by the performance of the **critical policy**. Unlike standard RL policies which carry out the optimal actions all of the time, our critical policy works as follows: *for critical states, it takes optimal actions while for non-critical states, it can take any action*. Therefore, our critical policy considers two factors: how to identify critical states and how to select optimal actions. For the former, both Critical-DQN (CriQN) and DQN can be used to calculate the Q-value difference to determine whether the state is critical or not. Similarly, for selecting optimal actions, both policies can be used to determine what is the best action to take once the state is determined to be critical. By combining the two ways of identifying critical states {CriQN, DQN} $\times$ two ways of suggesting optimal actions {CriQN, DQN}, we explored four critical policies shown in Table 5.1.

For example, the CriQN-DQN refers to the use of the CriQN to identify critical states and DQN to select optimal actions. Consequently, the performance of the critical policy is

Algorithm 1 Pseudocode of Critical-DQN

Initialize the training dataset D as (s, a, r, s') tuples.
Initialize the Q function with random parameters θ
Initialize the target $\hat{Q}$ function with parameters $\theta^- = \theta$
Load InferNet model
Set user-defined parameters: $T^+_{ShortTR}$, $T^-_{ShortTR}$, T_{LongTR}

// Initialize critical states based on immediate rewards
for each (s_i, a_i, r_i, s'_i) in D **do**
    $r'_{max} = max(InferNet(s'_i, a'))$
    $r'_{min} = min(InferNet(s'_i, a'))$
    if $r'_{max} > T^+_{ShortTR}$ **or** $r'_{min} < T^-_{ShortTR}$ **then**
        $c'_i = True$
    else
        $c'_i = False$
    end if
    $D \leftarrow (s_i, a_i, r_i, s'_i, c'_i)$
end for

// Initialize $Q(s, a)$ as immediate reward
for each $(s_i, a_i, r_i, s'_i, c'_i)$ in D **do**
    set $y_i = r_i$
end for
Perform gradient descent on $(y_i - Q(s_i, a_i; \theta))^2$
Reset $\hat{Q} = Q$

// Main Training Loop
for iteration k = 1, 2, ... till convergence **do**
    Initialize empty array Q_{diffs}
    for each $(s_i, a_i, r_i, s'_i, c'_i)$ in D **do**
        $Q_{diffs} \leftarrow (maxQ(s_i, a'; \theta^-) - minQ(s_i, a'; \theta^-))$
    end for
    $T_{\Delta(Q)} =$ top T_{LongTR} percent of Q_{diffs}

    for each $(s_i, a_i, r_i, s'_i, c'_i)$ in D **do**
        if terminal s'_i **then**
            Set $y_i = r_i$
        else
            $Q_{diff} = maxQ(s'_i, a'; \theta^-) - minQ(s'_i, a'; \theta^-)$
            if $Q_{diff} > T_{\Delta(Q)}$ **or** $c'_i == True$ **then**
                Set $y_i = r_i + \gamma max_{a'} Q(s', a'; \theta^-)$
            else
                Set $y_i = r_i + \gamma mean_{a'} Q(s', a'; \theta^-)$
            end if
        end if
    end for
    Perform gradient descent on $(y_i - Q(s_i, a_i; \theta))^2$
    Every C steps reset $\hat{Q} = Q$
end for

Table 5.1: Four Critical Policies

Critical-Policy	Identify Critical State	Select Optimal Action
CriQN-CriQN	CriQN	CriQN
DQN-DQN	DQN	DQN
CriQN-DQN	CriQN	DQN
DQN-CriQN	DQN	CriQN

determined by both factors: the accuracy of critical state identification and the choice of optimal action in the critical state. In the following, we investigate 1) how the two factors affect the performance of the critical policy and 2) how close the critical policy's performance is to a fully executed policy.

5.3 Experiment on GridWorld Testbed

5.3.1 GridWorld Description

The GridWorld environment is like a maze that the agent learns an optimal path from the start point to the end point. Figure 5.2 shows our GridWorld environment, which consists of 7 by 14 cells. The agent starts from the start state (right bottom corner), explores the 2D space, and finishes at the end state (left upper corner). There are several walls in the GridWorld which are marked as black blocks. The agent state is simply represented by the X and Y coordinates.

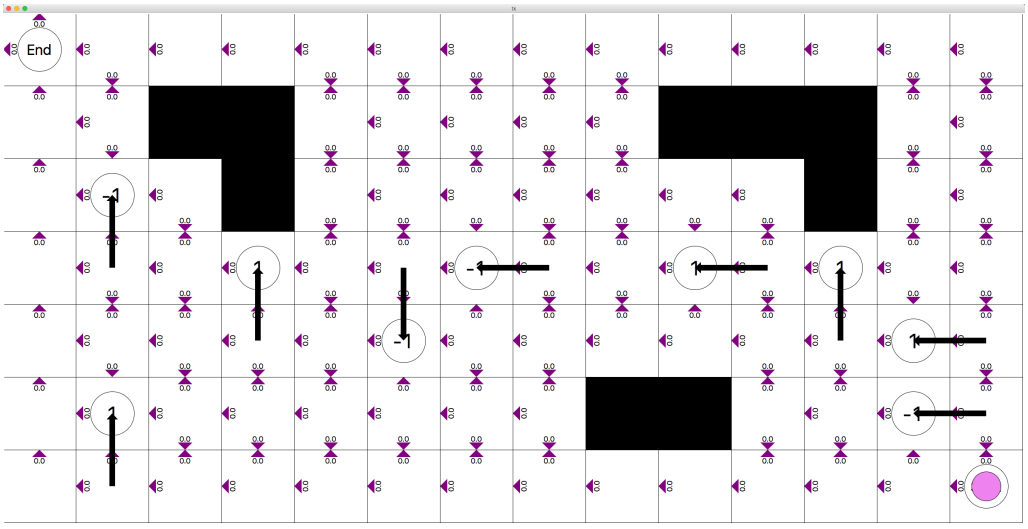


Figure 5.2: The Interface of the GridWorld Game

Action: At each step, the agent can take three actions: up, down, and left. In Figure 5.2, the

possible actions for each state are labeled as small purple triangles that some states have three possible actions while some have two or one possible actions. The possible actions for each state are predefined in the environment so that the agent never hits the wall or the boundary.
Reward: When moving in the GridWorld, there is a -0.1 reward penalty for each step and the agent can collect -1 and +1 rewards. In order to simulate the real world, the reward function is designed in state-action-state way, $R(s, a, s')$. The black arrows indicate that only enter the reward state along with the arrow, the agent can get the reward -1 or +1. Otherwise, the agent won't receive rewards. Furthermore, when the agent hits the reward state, it is forced to move left. This design aims to avoid the agent from collecting the same +1 reward repeatedly without forwarding it to the terminal state.
Stochastic: The GridWorld environment is stochastic in that the same state-action pair can result in different next states. For example, if the agent takes action 'left', it only has 85% chance of moving left, and 15% chance of moving to other possible directions.

Finally, the performance of the RL-induced policy in the GridWorld is evaluated by the final delayed reward which is the cumulative rewards during a trial. A good RL-induced policy should collect more +1 rewards, avoid -1 rewards and spend fewer steps to reach the goal.

5.3.2 Experiment Setup

In this experiment, we focus on the offline Reinforcement Learning approach and follow the three steps: 1) collect the training dataset by random exploration, 2) induce the policies offline and, 3) evaluate the performance of induced policies online.
Data Collection: The training dataset contains 1000 trajectories that are generated from random policy under different random seeds.
Offline Learning: Before inducing the policies, we apply the InferNet to infer the immediate rewards in the training dataset. The InferNet is implemented by Keras in Python. It contains three dense layers and each layer has 256 units with a dropout rate of 0.2. The batch size is 20 with a learning rate of 0.001. The training would stop when reaching the maximum training steps of 1,000,000. For policy induction, both Critical-DQN and original DQN are implemented by Keras. The neural network has two dense layers with 64 units. The batch size is 20 and the learning rate is 0.001. The frequency of copying the target network to the main network is 50 epochs, which means reading the entire training dataset through different batch samples 50 times. The training would stop when doing the copy 50 times, which results in about four million updates to the neural network.

For Critical-DQN, since the inferred immediate rewards do not change over the training process, the $T^{+}_{ShortTR}$ and $T^{-}_{ShortTR}$ are fixed as $+0.5$ and -0.5 based on the elbows of the inferred immediate reward distribution in the training dataset. For the T_{LongTR} parameter, we explore [10%, 20%, 30%, 40%, 50%] five different percentages, and thus five Critical-DQN policies are induced on the training dataset.

Online Evaluation: In the online evaluation, for any given state, we first apply InferNet to estimate the inferred immediate reward for each action. If the maximum inferred reward is larger than $T^{+}_{ShortTR}$ or the minimum is smaller than $T^{-}_{ShortTR}$, then the state is critical. Second, if the state is not critical, then we utilize RL policy to calculate Q-value difference and compare with its threshold $T_{\Delta(Q)}$, which is calculated based on the corresponding policy and T_{LongTR} in the training dataset. In the end, if the state is critical, the agent follows the corresponding policy to take the optimal action. Otherwise, the agent can select any action. Thus, there are two stages in the online evaluation, identify critical state and then select optimal action. For example, the critical policy CriQN-DQN will apply CriQN policy to calculate Q-value difference and compare with its threshold to determine criticalness, but apply DQN policy to select optimal actions.

The performance of the critical policy is measured by the average of cumulative rewards over 100 trials under different random seeds. To reduce the bias caused by data collection, we repeat the whole experiment 20 times with completely different random seeds to generate a more robust result.

5.3.3 Results

Figure 5.3 shows the online evaluation results. The X-axis indicates the LongTR threshold used by the corresponding identification policy to identify critical states. It is important to note that we define 100% as a fully-executed DQN policy that carries out the optimal actions all the time and 0% as a fully random policy that always randomly selects actions. Thus, 0% and 100% indicate the lower and upper performance bounds for critical policy. The Y-axis shows the reward (average of 20 replications with the shadows depicting the standard error) received by each critical policy. Overall, there is a general trend that the larger the threshold (the more states classified as critical states and take optimal actions), the better the critical policy performs. Next, we investigate in detail how the execution policy and identification policy may impact the performance of the critical policies.

5.3.3.1 Performance Comparison

Figure 5.3 shows that CriQN-CriQN (red) and CriQN-DQN (blue) perform very closely to each other while the performance of DQN-CriQN (green) and DQN-DQN (magenta) are very close; more importantly, the former two outperform the latter two across different LongTR thresholds. It suggests that the Critical-DQN is more accurate in identifying critical states than DQN while for carrying out the optimal actions, both Critical-DQN and DQN can be effective. Finally, when comparing to the fully-executed policy (100% threshold), the CriQN-CriQN and CriQN-DQN with threshold 50% can reach 90% performance of a fully-executed DQN policy.

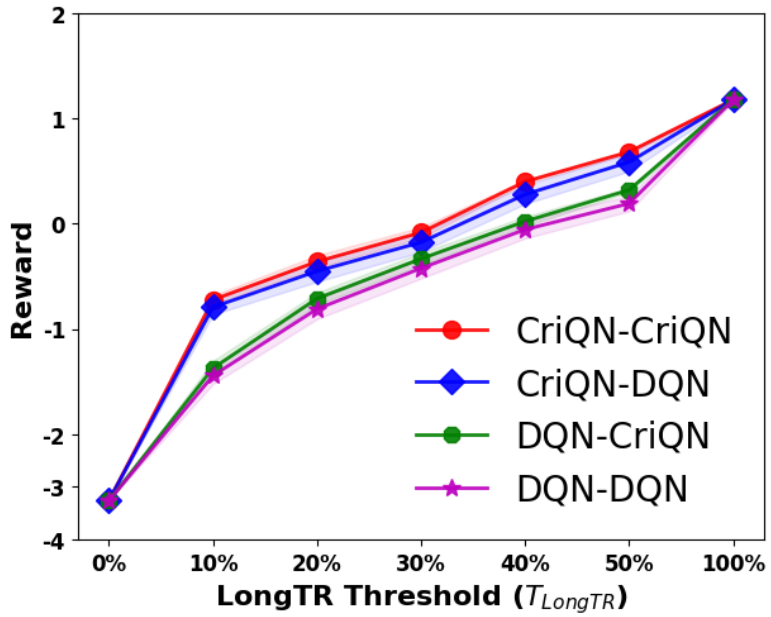


Figure 5.3: GridWorld Online Evaluation Result

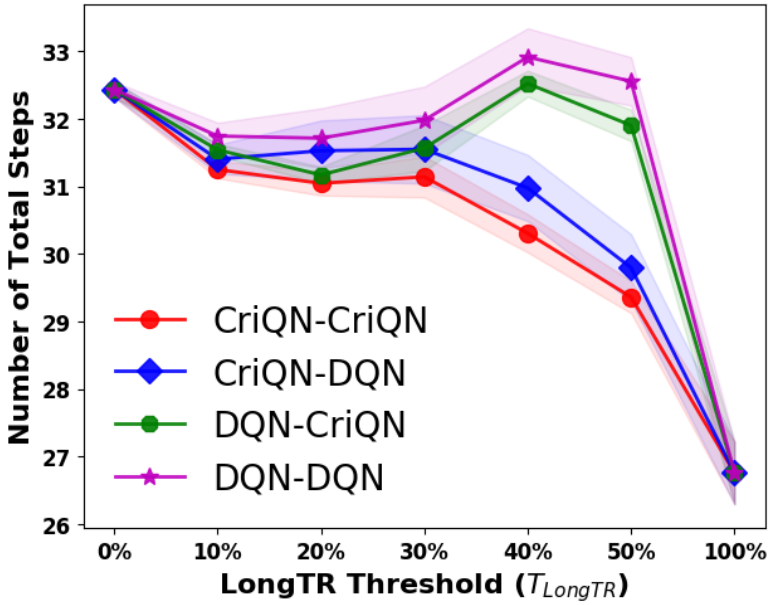


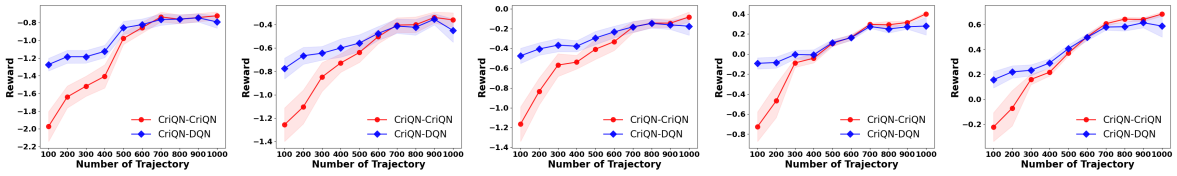
Figure 5.4: Total Steps in GridWorld Online Evaluation

5.3.3.2 Step Comparison

Figure 5.4 shows the run-time steps for each critical policy. The X-axis is the LongTR thresholds while the Y-axis is the number of total steps from start to end point in the online evaluation. In Figure 5.4, before threshold 30%, all the four critical policies take similar steps. However, after 30%, CriQN-CriQN and CriQN-DQN take significantly fewer steps than the DQN-CriQN and

DQN-DQN. It suggests that the critical states identified by Critical-DQN are more effective in reducing the number of steps in the trajectory. Furthermore, DQN-CriQN and DQN-DQN take more steps than the random policy at 40%. It suggests that inaccurate critical states can misguide the agent to take more steps.

5.3.3.3 Data-Efficiency for Critical-DQN



(a) T_{LongTR} : 10% (b) T_{LongTR} : 20% (c) T_{LongTR} : 30% (d) T_{LongTR} : 40% (e) T_{LongTR} : 50%

Figure 5.5: Critical-DQN vs. Original-DQN in Decision Making

From the previous results, we could draw a conclusion that Critical-DQN is better than original DQN regarding the identification of critical states, but there's no big difference in selecting optimal actions. This is because both Critical-DQN and DQN have enough data to induce an optimal policy and choose the best action. So what if we don't have enough data to train an optimal policy, how does the Critical-DQN perform?

Figures 5.5 (a)-(e) show the online performance of CriQN-CriQN vs. CriQN-DQN as the number of training trajectories increasing. The X-axis is the number of trajectories used to train the critical policies. The Y-axis is the reward received by each critical policy. In this experiment, we applied different LongTR thresholds to identify critical states and the only difference is which RL policy makes the decision in the critical states. The results show that when the training dataset is less than 400 trajectories, the CriQN-CriQN is worse than the CriQN-DQN across all five figures. When the training dataset is larger than 400 trajectories, they have similar performance. This indicates that the Critical-DQN needs more data to converge to an optimal policy. In summary, the Critical-DQN could provide the best LSTRs to identify critical states but it needs more data to make good decisions.

5.4 Offline Evaluation on Pyrenees

5.4.1 Experiment Setup

5.4.1.1 Training Dataset

Our training dataset comes from the Pyrenees 4.5.2 study which contains a total of 1148 students' interaction log collected over six semesters' classroom studies (16 Fall to 19 Spring).

The studies were assigned as regular homework to students. During the studies, all students used the same tutor, followed the same general procedure, studied the same training materials, and worked through the same training problems.

State: From the student-system interaction logs, 142 features were extracted which describes the student learning state. All the 142 features can be categorized into five groups that Autonomy features describe the amount of work done by the student; Temporal features are the time-related information during tutoring; Problem Solving features indicate the context of the problem itself; Performance features denote student's performance, and Student Action features record the student behavior information.

Action: For each problem, there are three possible actions: worked example (WE), problem solving (PS), and faded worked example (FWE). In WE, the student observes how the tutor solves a problem; in PS, the student solves the problem; in FWE, the student solves a portion of steps in a problem while the tutor shows how to solve the others. Within the problem, there are two possible actions for each step: elicit and tell. In elicit, the students solve the step by themselves; In tell, the tutor shows the step to students.

Reward: There's no immediate reward during tutoring and the delayed reward is the student's Squared Normalized Learning Gain (NLG-SR). Different from the traditional NLG: $\frac{posttest-pretest}{1-pretest}$, the NLG-SR: $\frac{posttest-pretest}{\sqrt{1-pretest}}$ has a square root in the denominator, which aims to reduce the variance. More specifically, figure 5.6 shows the distribution of NLG and NLG-SR in the training dataset. The X-axis is the pre-test score and the Y-axis is the corresponding Normalized Learning Gain value. In Figure 5.6, the blue dots have a lower variance than the orange dots. Particularly, the students with high pre-test scores often have a very low NLG. For NLG-SR, with the squared root on the denominator, the high-pre students could have a higher NLG-SR, which balanced to the low-pre students. Thus, we apply the NLG-SR to induce the optimal RL policy but the NLG is still the most essential metric for evaluating students' learning performance.

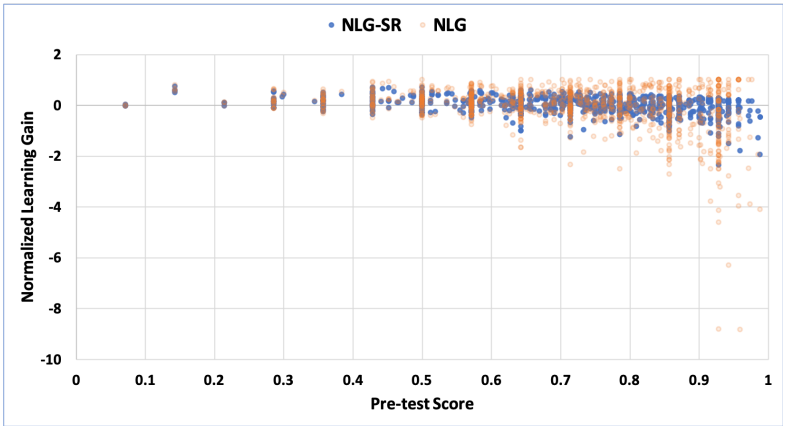


Figure 5.6: Distribution of NLG vs. NLG-SR in the Training Dataset

5.4.1.2 Offline Learning and Evaluation

The offline learning process follows the same process as the GridWorld in section 5.3.2. First, InferNet was applied to infer the immediate rewards for the training dataset. Then, policy DQN and CriQN were induced to construct the critical policy CriQN-CriQN, DQN-DQN, and CriQN-DQN. Finally, we fixed the threshold of ShortTRs based on the elbows in the distribution and explored the relationship between different LongTRs thresholds and the performance of the critical policies. For InferNet, we applied the same neural network setting as GridWorld. For policy induction, since the Pyrenees data is more complicated than the GridWorld environment, the neural network was consist of three dense layers with 256 units. All the other settings are the same.

Different from the online evaluation in the GridWorld game, we applied *off-policy* policy evaluation (OPE) metrics to evaluate the performance of the critical policies. In general, there are two types of OPE: model based and Importance Sampling (IS) based. The work in chapter 3 shows that Per Decision Importance Sampling (PDIS) is the best metric to evaluate the performance of RL-induced policies in the context of ITSs. Thus, PDIS was applied to evaluate the critical policies on the historical dataset. More specifically, if a state is identified as critical, the probability of taking an action is calculated by the softmax of Q-values among all the possible actions. On the contrary, if the state is identified as non-critical, then the probability of taking an action is the random probability 1/3 as there are three possible actions for each problem.

It is important to note that Pyrenees experiments are conducted early and the Critical-DQN algorithm is slightly different from the one described in algorithm 1, which also applies to the empirical study in Chapter 6. The difference is that 1) we didn't include the critical states identified by the ShortTRs in the training loop and 2) the parameter T_{LongTR} was always 50% during training. However, we didn't change the most important part of the algorithm: the modified Bellman Equation.

5.4.2 Results

In this section, we first present the offline evaluation results for all three critical policies. Then, we explore the distribution of identified critical states in the historical dataset.

5.4.2.1 Offline Evaluation Results

Figure 5.7 shows the offline evaluation results on the Pyrenees tutor dataset. The X-axis is the LongTR thresholds and the Y-axis is the PDIS value. In general, the higher the PDIS, the better the policy.

First of all, the general trend still holds that the more critical states (larger LongTR threshold), the better the policy would perform. When comparing the three lines, CriQN-DQN is the worst before 40% threshold. However, CriQN-DQN significantly outperforms

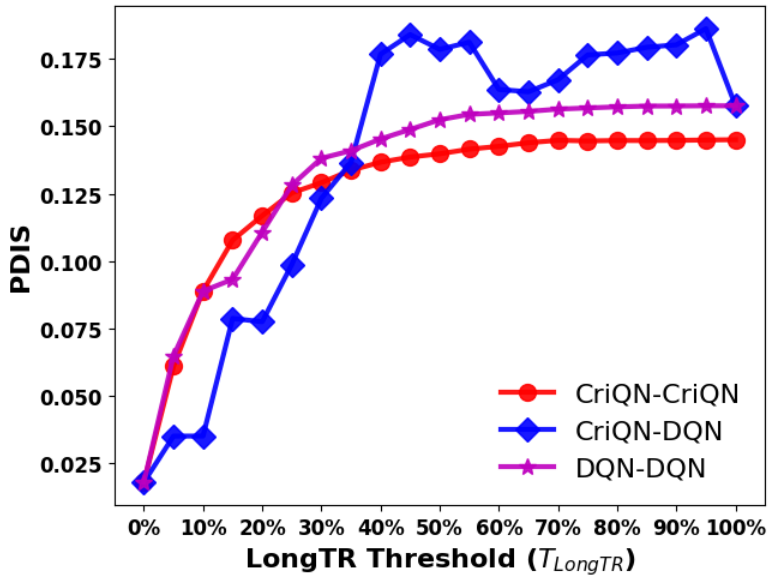


Figure 5.7: Offline Evaluation Results on Pyrenees Dataset

the other two critical policies after 40%. The reason is that the Pyrenees dataset is not large enough for the Critical-DQN to find an optimal policy, but the LSTRs in Critical-DQN are still accurate to identify critical states. Furthermore, the dramatic increase of the CriQN-DQN performance around 50% demonstrates the reliability of the Critical-DQN algorithm, because the LongTR threshold is always 50% during training. In summary, the result reflects that the LSTR framework is effective in identifying critical decisions.

5.4.2.2 Exploring Critical States

Table 5.2: Distribution of Critical States in each Problem

	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10
Long-Term Rewards	3%	16%	16%	13%	15%	11%	7%	7%	6%	6%
Short-Term Rewards	0%	1%	6%	6%	6%	10%	13%	19%	19%	20%
Long-Short Term Rewards	3%	14%	15%	12%	14%	11%	8%	8%	8%	8%

In order to further investigate the critical decisions identified by LSTRs in the tutor dataset, we analyzed where did they occur. 50% threshold for the Critical-DQN in figure 5.7 was applied to identify critical states in the tutor dataset and table 5.2 shows the distribution of critical states in each problem. The first row represents the 10 problems in chronological order. The second row indicates the percentage of critical states identified by LongTRs in different problems. For example, 3% of critical states happen in P1 while 16% appear in P2. It indicates

that the LongTRs focus on critical decisions in the early to mid-stage. In the meantime, the third row shows that the ShortTRs focus on the critical decisions in the late stage. The fourth row shows the critical states identified by LSTRs, which is the union set of the critical decisions from LongTRs and ShortTRs. Overall, critical states are almost evenly distributed among all the problems except the first one. It is not surprising that the first one is not so important because in the first problem, students are not familiar with the system and the policy needs more data to know the student status better. Furthermore, the result reflects that the LongTRs and ShortTRs complement each other. If we only focus on LongTRs, we will miss the important decisions in the late stage, otherwise we will miss the important decisions in the early to mid-stage.

5.5 Conclusion

In this chapter, we proposed and explored the Long-Short Term Rewards framework to identify critical decisions in both an ideal GridWorld game and a real-world ITS dataset. Based on the LSTR framework, we proposed a Critical-DQN algorithm to induce critical policy whose Q-value difference is more heuristic and sensitive to the decision importance. In order to investigate the effectiveness of the LSTR framework, we evaluated the performance of critical policies with different LongTR thresholds through online evaluation on GridWorld and offline evaluation on Pyrenees tutor's dataset. The results showed that the Critical-DQN is significantly better than the original DQN in identifying critical states. However, the Critical-DQN needs more data to converge to an optimal policy. So, the best critical policy CriQN-DQN is using Critical-DQN to identify critical states but utilizing the original DQN to make decisions.

In summary, through identifying critical decisions by LSTR framework and Critical-DQN, carry out the optimal policy on 50% decisions (critical ones) could achieve 90% performance of carrying out on all decisions. However, we only explored the LSTR framework offline on a real-world ITS dataset. In the next chapter, we will evaluate the effectiveness of the LSTR framework through an empirical classroom study.

CHAPTER 6

EVALUATION OF LSTR USING AN INTELLIGENT TUTORING SYSTEM

Publish: Song Ju, Guojing Zhou, Mark Abdelshiheed, Tiffany Barnes and Min Chi, Evaluating Critical Reinforcement Learning Framework In the Field. AIED 2021, To appear.

In this chapter, we further evaluated the effectiveness of the LSTR framework on real-world ITS Pyrenees via an empirical classroom study. The results showed that the identified critical decisions are indeed critical that 1) optimal and sub-optimal actions in critical states can make a significant difference, 2) only carry out optimal actions in critical states is as effective as a fully-executed policy. In summary, the studies in this chapter provide strong evidence of the effectiveness of our LSTR framework in identifying critical decisions.

6.1 Introduction

In this work, we *empirically* evaluated the effectiveness of the LSTR framework on a real-world ITS: Pyrenees4.5.2, where the student-agent interactions can be viewed as a temporal sequence of steps [And95; Koe97]. Most ITSs are *tutor-driven* in that *the tutor* decides what to do next. For example, the tutor can *elicit* the subsequent step from the student either with prompting or without (e.g., in a free form entry window where each equation is a step). When a student enters an entry on a step, the ITS records its success or failure and may give feedback (e.g. correct/incorrect markings) and/or hints (suggestions for what to do next). Alternatively, the tutor can choose to *tell* them the next step directly. Each of such decisions affects the student's successive actions and performance and some may be more impactful than others. *Pedagogical policies* are used for the agent (tutor) to decide what action to take next in the face of alternatives.

To confirm whether the identified critical decisions are indeed critical, we argue that our identified critical decisions and induced Critical policy should satisfy two conditions. First, they should satisfy the *Necessary Hypothesis* stating that it is *necessary* to carry out optimal actions in critical states otherwise the performance would suffer. To validate it, we compared two policies: Critical-optimal (Critical-Opt) vs. Critical-suboptimal (Critical-Sub). Both policies would carry out random actions in non-critical states and the only difference is that in critical states, Critical-Opt takes optimal actions while Critical-Sub takes suboptimal actions. As expected, our results showed that the former was indeed significantly more effective than the latter. Second, our induced Critical policy should satisfy the *Sufficient Hypothesis* stating that carrying out optimal actions in the critical states is *sufficient*. In other words, only carrying out optimal actions in critical states is as effective as a fully-executed RL policy. To validate it, we compared the Critical-Opt policy with a Full RL policy which takes optimal actions in every state. Our results showed that no significant difference was found between them.

In this work, we focus on pedagogical decisions at two levels of granularity: *problem* and *step*. More specifically, our tutor will first make a problem-level decision and then make step-level decisions based on the problem-level decision. For the former, our tutor 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 tutor 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 tell a step based on the corresponding induced step-level policy. While much of the prior work has relied on hand-coded or RL-induced pedagogical policies on these decisions, there is no well-established theory or widely accepted consensus on how WE vs. PS. vs. FWE can be best used and how they may impact students' learning. As far as we know, no prior research has investigated *when it is critical to give WE vs. PS vs. FWE*. In this work, by empirically confirming that our identified critical decisions and Critical policy satisfy the two hypotheses, we argue that the proposed LSTR framework sheds some light on identifying the moments that offering WE, PS, or FWE can make a difference.

6.2 Related Work

6.2.1 WE, PS and FWE

A variety of studies have explored the effectiveness of WE, PS, FWE, and their various combinations [Swe85; McL08; McL11; McL14; VG11; Ren02; Sch09; Naj14; Sal10; Zho15]. For example, Mclaren et al. compared WE-PS pairs with PS-only in a study [McL08] and WE-only, PS-only and WE-PS pairs in another study [McL11]. Overall, results suggested

that studying WE can be as effective as doing PS, but students spend less time on WE. For FWE-involved studies, Renkl et al. [Ren02] compared WE-FWE-PS with WE-PS pairs. Results showed that the WE-FWE-PS condition significantly outperformed the WE-PS condition, and there is no significant time-on-task difference between them. Similarly, Najar et al. [Naj14] compared adaptive WE/FWE/PS with WE-PS pairs and found the former is significantly more effective than the latter. In summary, prior studies have demonstrated that adaptively alternating amongst WE, PS, and FWE is more effective than hand-coded expert rules in terms of improving student learning. However, it is still not clear which alternating is critical to the student learning outcome.

6.3 Hierarchical RL Policy Induction

Our tutor Pyrenees can make both problem-level decisions (WE/PS/FWE) and step-level decisions (elicit/tell). With the two levels of granularity, we extended the existing flat-RL algorithm to Hierarchical RL (HRL), which aims to induce an optimal policy to make decisions at different levels. Most HRL algorithms are based upon an extension of Markov Decision Processes (MDPs) called Discrete Semi-Markov Decision Processes (SMDPs). Different from MDPs, SMDPs have an additional set of complex activities [Bar03] or options [Sut99], each of which can invoke other activities recursively, thus allowing the hierarchical policy to function. The *complex* activities that 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. For HRL, learning occurs at multiple levels. The global learning generates a policy for the complex level decisions and local learning generates a policy for the primitive level decisions in each complex activity. More importantly, the goal of local learning is not inducing the optimal policy for the overall task, but the optimal policy for the corresponding complex activity. Therefore, our HRL approach learns a global problem level policy to make decisions on WE/PS/FWE and learns a local step level policy for each problem to choose between elicit/tell. More specifically, both problem and step level policies were learned by recursively using DQN or Critical-DQN to update the Q-value function until convergence.

When inducing Critical-DQN and original DQN policies, we applied the same training dataset and neural network settings as described in Section 5.4.1. The only difference is that through utilizing HRL, the critical policies could make decisions on both problem and step levels.

6.4 Experiment Setup

6.4.1 Conditions

In this study, the effectiveness of the LSTR framework is evaluated by comparing three policies: Critical-optimal (Critical-Opt), Critical-suboptimal (Critical-Sub), and Full. These three policies are designed based upon two hypotheses:

- **Necessary Hypothesis**: optimal actions must be carried out in critical states.
- **Sufficient Hypothesis**: only carry out optimal actions in critical states can be as effective as a fully executed policy.

First, the *Necessary Hypothesis* stating that it is *necessary* to carry out optimal actions in critical states otherwise the performance would suffer. To validate it, we compared two policies: Critical-Opt vs. Critical-Sub, as shown in Table 6.1. Both policies would carry out random actions in non-critical states and the only difference is that in critical states, Critical-Opt takes optimal actions while Critical-Sub takes suboptimal actions. If the LSTR-identified critical states are indeed critical, then different actions in the critical states can make a significant difference, and we expect the Critical-Opt policy is significantly more effective than the latter in terms of improving students' learning performance.

Table 6.1: Necessary Hypothesis Conditions

Policy	**Critical State**	**Non-Critical State**
Critical-Opt	optimal action	random action
Critical-Sub	suboptimal action with minimum Q-value	random action

Second, the *Sufficient Hypothesis* stating that carrying out optimal actions in the critical states is *sufficient*. In other words, only carrying out optimal actions in critical states is as effective as a fully-executed RL policy. To validate it, we compared the Critical-Opt policy with a Full RL policy which takes optimal actions in every state, as shown in Table 6.2. If the LSTR-identified critical states are indeed critical, then different actions in the non-critical states have little impact on the final outcome, and we expect the Critical-Opt policy is as effective as the Full policy.

Table 6.2: Sufficient Hypothesis Conditions

Policy	**Critical State**	**Non-Critical State**
Critical-Opt	optimal action	random action
Full	optimal action	optimal action

For the three policies, we induced a standard DQN policy as the Full policy to carry out optimal actions in all states. Then Critical-DQN ($T_{LongTR} = 50\%$) policy was induced to identify critical states. The Critical-Opt policy would carry out optimal actions in critical states but the Critical-Sub policy would take sub-optimal actions with minimum Q-value. In non-critical states, both of them acted randomly.

6.4.2 Participants

The participants of this study were undergraduate students enrolled in the Discrete Mathematics class in 2020 Spring. This study was assigned to students as one of their regular homework assignments. Completion of the study was required for full credit and students were told that they will be graded based on their demonstrated effort, not their learning performance.

164 students were randomly assigned to the three conditions. Due to preparation for final exams and the length of study, 129 students completed the study. 14 students were excluded from our subsequent statistical analysis in which 8 students performed perfectly in the pre-test and 6 students worked in groups. The final group sizes were $N = 37$ (Critical-Opt), $N = 39$ (Critical-Sub) and $N = 39$ (Full). We performed a Chi-square test on the relationship between students' condition and their completion rate and found no significant difference among the conditions: $\emptyset^2(2) = 0.167$, $p = 0.92$.

6.4.3 Procedure and Measures

The classroom study followed the same procedure: 1) pre-training, 2) pre-test, 3) training on Pyrenees tutor, and 4) post-test as described in 4.5.3. After the study, we performed comprehensive statistic analysis on the students' learning performance, percentage of critical states and policy's behavior. Specific measures are listed below:

- **Pre-test:** calculated based on the 14 pre-test problems;
- **Isomorphic Post-test:** calculated based on the 14 post-test problems that are isomorphic to pre-test;
- **Full Post-test:** calculated based on the 20 post-test problems;
- **Isomorphic NLG:** calculated based on the pre- and isomorphic post-test score with traditional equation: $\frac{isoposttest - pretest}{1 - pretest}$;
- **NLG:** calculated based on the pre- and full post-test score with traditional equation: $\frac{posttest - pretest}{1 - pretest}$;
- **Percentage of Critical States:** calculated following the equation: $\frac{number\ of\ critical\ steps}{number\ of\ total\ steps}$;

6.5 Results

We will report our results based on the two hypotheses. For the Necessary Hypothesis, we compare Critical-Opt vs. Critical-Sub conditions and for the Sufficient Hypothesis, we compare Critical-Opt vs. Full conditions.

6.5.1 Necessary Hypothesis: Critical-Opt vs. Critical-Sub

Table 6.3 presents the mean and standard deviation (SD) for students' learning performance and the pairwise t-test results between the Critical-Opt vs. Critical-Sub conditions, Table 6.4 shows the percentage of critical states in both problem and step levels, Table 6.5 shows the number of different types of decisions the students received in the training phase.

Table 6.3: Learning Performance in Critical-Opt vs. Critical-Sub

Measure	**Critical-Opt**	**Critical-Sub**	**Pairwise T-test Result**
Pre	0.75 (0.18)	0.72 (0.20)	$t(112) = 0.564$, $p = 0.57$, $d = 0.13$
Iso Post	0.89 (0.16)	0.86 (0.16)	$t(112) = 0.806$, $p = 0.42$, $d = 0.18$
Post	0.82 (0.19)	0.78 (0.19)	$t(112) = 0.991$, $p = 0.32$, $d = 0.23$
Iso NLG	0.70 (0.36)	0.40 (0.85)	$t(112) = 2.274$, $p = 0.025$, $d = 0.52$
NLG	0.41 (0.39)	0.01 (1.25)	$t(112) = 2.183$, $p = 0.031$, $d = 0.49$
Time on Task (minutes)	94.49 (35.14)	78.14 (26.67)	$t(112) = 2.302$, $p = 0.023$, $d = 0.52$

Table 6.4: Percentage of Critical States in Critical-Opt vs. Critical-Sub

Measure	**Critical-Opt**	**Critical-Sub**	**Pairwise T-test Result**
Problem Level (%)	46.9 (23.4)	31.5 (17.6)	$t(112) = 3.686$, $p < 0.001$, $d = 0.84$
Step Level (%)	60.2 (20.1)	45.3 (26.0)	$t(112) = 2.424$, $p = 0.017$, $d = 0.55$

Table 6.5: Tutor Decisions in Critical-Opt vs. Critical-Sub

Measure	**Critical-Opt**	**Critical-Sub**	**Pairwise T-test Result**
PS Count	3.56 (1.85)	2.38 (1.41)	$t(112) = 3.596$, $p < 0.001$, $d = 0.81$
WE Count	2.54 (1.87)	5.13 (1.51)	$t(112) = -7.274$, $p < 0.001$, $d = 1.65$
FWE Count	3.90 (2.00)	2.49 (1.34)	$t(112) = 4.008$, $p < 0.001$, $d = 0.91$
Elicit Count	83.28 (49.18)	43.97 (30.64)	$t(112) = 4.372$, $p < 0.001$, $d = 0.99$
Tell Count	82.92 (50.33)	55.41 (35.16)	$t(112) = 3.059$, $p = 0.003$, $d = 0.69$

Pre-test Score: A pairwise t-test showed no significant difference between Critical-Opt vs. Critical-Sub on the pre-test scores: $t(112) = 0.564$, $p = 0.57$, $d = 0.13$. The result suggests that the two conditions are balanced in terms of incoming competence.

Improvement through training: In order to examine how students learned on the Pyrenees tutor, we conducted a comparison between the scores on pre-test and isomorphic post-test questions. A repeated measures analysis using test-type (pre-test and isomorphic post-test) as factors and test score as dependent measure showed all the two conditions score significantly higher on isomorphic questions in post-test than in pre-test: $F(1, 38) = 13.68$, $p = 0.0004$, $\jmath = 0.392$ for Critical-Opt and $F(1, 38) = 11.5$, $p = 0.0011$, $\jmath = 0.362$ for Critical-Sub. It reveals that the Pyrenees tutor indeed improves students' learning regardless of the pedagogical policies deployed.

Learning Performance: To investigate students' learning performance between the two conditions, we compared their isomorphic NLG (calculated based on Pre- and Iso Post-test) and full NLG (based on Pre- and Full Post-test). The full post-test contains six additional multiple-principle problems. Pairwise t-tests showed that Critical-Opt scored significantly higher than Critical-Sub on both the isomorphic NLG: $t(112) = 2.274$, $p = 0.025$, $d = 0.52$ and the full NLG: $t(112) = 2.183$, $p = 0.031$, $d = 0.49$. The results showed that the Critical-Opt policy is more effective than the Critical-Sub policy. It supports our hypothesis that different actions in the critical states can make a significant difference, so optimal actions *must* be made in critical states.

Time on Task and Percentage of Critical States: A pairwise t-test analysis revealed that Critical-Opt spend significantly more time than Critical-Sub in the training phase: $t(112) = 2.302$, $p = 0.023$, $d = 0.52$. In Table 6.4, pairwise t-test showed that Critical-Opt experienced significantly more critical states than Critical-Sub on both problem level: $t(112) = 3.686$, $p < 0.001$, $d = 0.84$ and step level: $t(112) = 2.424$, $p = 0.017$, $d = 0.55$. This suggests that the Critical-Opt policy is more likely to lead students to the critical intersections that make a difference.

Tutor Decisions: We investigated the number of different types of actions students received during training, as shown in the Table 6.5. *Note that for step level decisions, we only considered the elicits and tells in the FWEs.* For the problem level, Critical-Opt received significantly more PS: $t(112) = 3.596$, $p < 0.001$, $d = 0.81$, more FWE: $t(112) = 4.008$, $p < 0.001$, $d = 0.91$ and fewer WE: $t(112) = -7.274$, $p < 0.001$, $d = 1.65$ than Critical-Sub. For the step level, the former also received significantly more elicit: $t(112) = 4.372$, $p < 0.001$, $d = 0.99$ and more tell: $t(112) = 3.059$, $p = 0.003$, $d = 0.69$ than Critical-Sub. The results indicate that the Critical-Sub policy prefers WEs while the Critical-Opt policy prefers PSs and FWEs.

6.5.2 Sufficient Hypothesis: Critical-Opt vs. Full

Similarly, we conduct the comparisons between Critical-Opt vs. Full on the learning performance as shown in Table 6.6, the percentage of critical states as shown in Table,6.7 and the

number of different types of decisions as shown in Table 6.8.

Table 6.6: Learning Performance in Critical-Opt vs. Full

Measure	Critical-Opt	Full	Pairwise T-test Result
Pre	0.75 (0.18)	0.70 (0.19)	$t(112) = 1.178$, $p = 0.24$, $d = 0.27$
Iso Post	0.89 (0.16)	0.84 (0.20)	$t(112) = 1.231$, $p = 0.22$, $d = 0.28$
Post	0.82 (0.19)	0.75 (0.20)	$t(112) = 1.478$, $p = 0.14$, $d = 0.34$
Iso NLG	0.70 (0.36)	0.56 (0.40)	$t(112) = 0.999$, $p = 0.32$, $d = 0.23$
NLG	0.41 (0.39)	0.18 (0.55)	$t(112) = 1.242$, $p = 0.217$, $d = 0.29$
Time on Task (minutes)	94.49 (35.14)	91.50 (31.72)	$t(112) = 0.416$, $p = 0.678$, $d = 0.1$

Table 6.7: Percentage of Critical States in Critical-Opt vs. Full

Measure	Critical-Opt	Full	Pairwise T-test Result
Problem Level (%)	46.9 (23.4)	38.4 (12.4)	$t(112) = 2.02$, $p = 0.046$, $d = 0.46$
Step Level (%)	60.2 (20.1)	62.1 (34.0)	$t(112) = -0.294$, $p = 0.769$, $d = 0.07$

Table 6.8: Tutor Decisions in Critical-Opt vs. Full

Measure	Critical-Opt	Full	Pairwise T-test Result
PS Count	3.56 (1.85)	3.32 (0.81)	$t(112) = 0.721$, $p = 0.472$, $d = 0.17$
WE Count	2.54 (1.87)	5.24 (1.19)	$t(112) = -7.496$, $p < 0.001$, $d = 1.72$
FWE Count	3.90 (2.00)	1.43 (1.10)	$t(112) = 6.913$, $p < 0.001$, $d = 1.59$
Elicit Count	83.28 (49.18)	33.19 (35.10)	$t(112) = 5.498$, $p < 0.001$, $d = 1.26$
Tell Count	82.92 (50.33)	29.76 (28.30)	$t(112) = 5.833$, $p < 0.001$, $d = 1.34$

Pre-test Score: A pairwise t-test analysis showed that there is no difference between Critical-Opt vs. Full on the pre-test score: $t(112) = 1.178$, $p = 0.24$, $d = 0.27$. This suggests again that our random assignment indeed balanced students' incoming competence.

Improvement through training: A repeated measures analysis using test-type (pre-test and isomorphic post-test) as factors and test score as dependent measure showed that similar to Critical-Opt, Full scored significantly higher in isomorphic post-test than in pre-test: $F(1, 36) = 11.0$, $p = 0.0015$, $\jmath = 0.363$.

Learning Performance: The pairwise t-tests showed that there is no significant difference between the Critical-Opt and Full conditions on the two learning metrics, isomorphic NLG: $t(112) = 0.999$, $p = 0.32$, $d = 0.23$ and full NLG: $t(112) = 1.242$, $p = 0.217$, $d = 0.29$. It implies

that only carrying out optimal actions in critical states is as effective as a fully-executed policy.

Furthermore, to determine whether these null results are significant, that is, the Critical-Opt indeed performs as effective as Full, we calculated the effect size on all the comparisons and we found that they are all not statistically significant in that $\beta < 0.8$: 0.17 for isomorphic NLG while 0.23 for NLG. On the other hand, across all the comparisons, Critical-Opt was slightly better than the Full. This result suggests that if we have enough population samples, the former can outperform the latter.

Time on Task and Percentage of Critical States: A pairwise t-test analysis revealed that the Critical-Opt condition spend a similar amount of time as the Full condition in the training phase: $t(112) = 0.42$, $p = .678$, $d = 0.10$. In Table 6.7, pairwise t-tests showed that the Critical-Opt condition has significantly more critical states than the Full condition in the problem level: $t(112) = 2.02$, $p = 0.046$, $d = 0.46$ but no difference in the step level: $t(112) = -0.294$, $p = 0.769$, $d = 0.07$. The result suggests that the optimal actions in the non-critical states could reduce the chance of entering critical states.

Tutor Decisions: For the problem level, the Critical-Opt condition has significantly more FWE: $t(112) = 6.913$, $p < 0.001$, $d = 1.59$, fewer WE: $t(112) = -7.496$, $p < 0.001$, $d = 1.72$ decisions than the Full condition, but no difference on PS: $t(112) = 0.721$, $p = 0.472$, $d = 0.17$. For the step level, the Critical-Opt condition received significantly more elicit: $t(112) = 5.498$, $p < 0.001$, $d = 1.26$ and more tell: $t(112) = 5.833$, $p = 0.003$, $d = 1.34$ than the Full condition. The results suggest that the random actions in non-critical states could lead the RL policy to give more FWE and fewer WE in critical states.

6.6 Conclusion

In the empirical classroom study, we evaluated the effectiveness of the LSTR framework by comparing the Critical-Opt policy with two baseline policies: Critical-Sub policy and Full policy. The comparisons are based upon two hypotheses: 1) optimal actions must be carried out in critical states, 2) only carry out optimal actions in critical states can be as effective as the fully-executed policy. The results showed that in terms of students' learning performance, 1) the Critical-Opt policy significantly outperforms the Critical-Sub policy and 2) the Critical-Opt policy performs as effectively as the Full policy. It suggests that our LSTR framework indeed identifies the critical decisions and satisfies the two hypotheses that 1) taking optimal actions in the identified critical states significantly outperform taking suboptimal actions; 2) only taking optimal actions in the critical moments can be as effective as taking optimal actions in every moment.

In summary, the empirical classroom study provides strong evidence to draw a conclusion that our LSTR framework indeed identifies the critical pedagogical decisions in students' learning. In the next chapter, we will generalize our LSTR framework to a healthcare task: sepsis treatment.

CHAPTER 7

EVALUATION OF LSTR USING HEALTHCARE DATASETS

In this chapter, we generalized our framework to a real-world healthcare task: sepsis treatment. The goal of identifying critical decisions is to reduce the healthcare workload and in the meantime keep the sepsis treatment effective. The results showed that our Critical-DQN induced policy could automatically identify the critical moments in the septic treatment process, thus reducing the burden of medical decision-makers by allowing them to focus on shock patients without negatively impacting non-shock patients.

7.1 Introduction

In healthcare systems, the heavy workload of practitioners is a critical concept affecting the quality of care and patient outcomes. Workload is defined as “the task demand of accomplishing mission requirements for the human operator” [Hoo17]. Interpretation and quantification of workload in healthcare delivery depends on many factors [Swi11] and has been quantified using objective, physiological and subjective measures [Hoo16]. Treatment decision-making is a type of workload which plays a critical role in clinician performance. In hospitals, physicians make a large number of clinical decisions from defining the problem, to evaluating test results, to treatments. For example, in one study, an average of 13.4 decisions was made during a patient visit [Ofs18]. However, clinical decision-makers do not always make optimal or consistent decisions in such complex tasks for many reasons. One of them is "decision fatigue". After a long series of decisions, people tend to develop cognitive fatigue which can lead them to favor the seemingly easiest option over all the others. One study found that decision-makers tend to procrastinate, be less persistent, and even fail to recognize

decision opportunities at all [Pig20]. For instance, nurses who suffer from decision fatigue are more likely to make conservative decisions, which indicates that they prefer to choose the 'default' option [All19]. An abundance of conservative decisions can have unwanted consequences in resource and time-limited environments.

We focus on an extremely challenging task: sepsis treatment. Sepsis, defined as infection plus systemic manifestations of infection, is the greatest in-hospital cause of mortality and source of expense; the syndrome has a high mortality (43.8% in the high-target group and 42.3% in the low-target group at 90 days) even if treated according to recommended guidelines [Rho17]. This is due, in part, to difficulties in diagnosis and delayed treatment. For every one-hour delay in treatment of severe sepsis/shock with antibiotics, there is a 10% decrease in patient survival probability. On the other hand, there are many barriers to timely, effective treatment of sepsis: response and treatment depend on many factors including the type of infection and the predisposition. The treatment of sepsis patients is complex – the patient's condition is stochastic and dynamically changing during the diagnosis process. Furthermore, the diagnosis of sepsis requires the selection and ordering of potentially invasive and/or costly imprecise tests. The patient's response to treatment is uncertain, and the treatment itself evolves over time as the care provider learns more about the patient's condition through lab tests, vital signs, and the patient's response to treatment over time. In septic treatment, furthermore, relatively unimportant clinical examinations might hinder detecting the deteriorating patients [Han18]. Such finding implies it is important to find critical decision timings to successfully treat a septic patient.

Like many real-world tasks, sepsis treatment can be characterized as a temporal sequential multi-step decision process, where the outcome of the selected treatment is delayed. Reinforcement Learning (RL) offers an effective data-driven solution based on a mathematically grounded framework that learns an optimal policy from data to maximize expected reward [**sutton2018**]. In particular, Deep RL (DRL) effectively models high-dimensional data and has been applied to sepsis treatment [Rag17]. Despite these efforts, in real-world domains like healthcare, such automated decision-making approaches are undesirable and unacceptable due to ethical, legal, and moral reasons. Therefore, we expect humans will remain as the primary decision-makers, and RL as the secondary decision-makers to support humans. In this scenario, our LSTR framework has two goals: 1) to induce a septic treatment policy that aims to minimize the septic shock rate, and 2) to identify the critical moments in the septic progress to enhance the physicians' decision-making in the treatment for septic patients.

In this study, we evaluated the effectiveness of the LSTR framework on two real-world healthcare datasets from two aspects: policy performance and percentage of nudges. The policy performance reflects the power of policy in preventing patients from getting septic shock. The nudge is the moment where it is critical and the RL policy has a different decision from the physician. In other words, nudge is the **alerting** in our human-machine mixed-initiative decisionmaking framework shown in the figure 1.1 (c) in chapter 1. It is important to note

that in healthcare, physicians are the primary decision-makers and RL should not interfere too much. So, the critical policy should only intervene when the moment is critical and it disagrees with the physician's decision. Our result shows that the LSTR framework indeed identifies the critical moments in the septic treatment. Furthermore, the critical policy could unburden the physicians from the non-shock patients but keep attention on the shock patients.

7.2 Two Healthcare Datasets

7.2.1 CCHS Dataset

7.2.1.1 Description

One of our datasets is the electronic health records (EHR) data collected from Christiana Care Health System (CCHS). In total there are 210,289 visits and 9,029,493 events. By combining the International Classification of Diseases, Ninth Revision (ICD-9), and clinician rules, we sampled 1,800 positive septic shock trajectories and 1,800 negative trajectories (no shock), keeping the same distribution of age, gender, race, and the length of hospital stay. To impute the missing value, we applied the expert imputation rules that 1) the values of vital signs were carried forward for 8 hours, 2) the values of lab results were carried forward for 24 hours and, 3) the remaining missing values were imputed by mean values. The final dataset contains 3,600 visits (50% shock, 50% no shock) and 84,160 events where the average length of trajectories is 24 and the maximum length is 317.

7.2.1.2 State, Action, Reward

In order to fit the data into the RL framework, appropriate state, action, and reward were build based on the advice from domain experts. The state contains twenty-one sepsis-related features including vital signs, lab results, and oxygen controls. The action space is a binary combination of two types of medical treatments: antibiotic and oxygen control, and thus we have four actions in total. To define rewards, four septic stages were defined based on the clinical rules, and the delayed reward for each stage was set as follows: Infection (± 1), Inflammation (± 50), OrganFailure (± 100) and Shock (± 1000). The designated negative reward was given when a patient enters the corresponding stage, and its positive reward was given back when the patient recovers from the stage. In this way, an optimal policy should keep patients from getting negative rewards and help them stay in non-negative states.

7.2.2 Mayo Dataset

7.2.2.1 Description

Besides the CCHS dataset, the other dataset contains the EHR data collected from Mayo clinic. In total, there are 221,700 visits and 144,693,491 events. Similarly, by combining the ICD-9

and clinician rules, we sampled 2,205 positive septic shock trajectories and 2,205 negative trajectories (no shock), keeping the same distribution of age, gender, race, and length of hospital stay. To impute the missing value, we applied the same rule with the CCHS dataset. The final dataset includes 4,410 visits (50% shock, 50% no shock) and 392,850 events where the average length of trajectories is 65 and the maximum length is 1160.

7.2.2.2 State, Action, Reward

In the Mayo dataset, fourteen sepsis-related features were selected to represent the patients' states. Four medical treatments were regarded as actions: no treatment, oxygen control, and administration of two types of medicine: anti-infection drug and vasopressor. The reward function was defined by two leading clinicians from two hospitals based on the severity of several septic stages: infection, inflammation, four levels of organ failure (OF), and septic shock as follows: Infection ($\pm$1), Inflammation ($\pm$2), Single OF level-1 ($\pm$5), Single OF level-2 ($\pm$10), Multiple OF level-1 ($\pm$20), Multiple OF level-2 ($\pm$30), and Shock ($\pm$50). Similarly, the designated negative reward was given when a patient enters the corresponding stage, and its positive reward was given back when the patient recovers from the stage.

7.3 Experiment Setup

7.3.1 Offline Learning

Similar to prior experiments, the offline learning process follows the same process as the GridWorld in section 5.3.2. First, InferNet was trained and applied to infer the immediate rewards for the training dataset. Second, $T^{+}_{ShortTR}$ and $T^{-}_{ShortTR}$ were selected based on the elbows of the inferred immediate reward distribution. Finally, DQN and Critical-DQN policies were induced with different LongTR thresholds. More specifically, we compared three critical policies: CriQN-CriQN, DQN-DQN, and CriQN-DQN with two baseline policies: a fully-executed DQN (Full) policy and a Physician's policy. To train a policy that follows the physician's actions, we followed the same procedure as described in [Azi19] by using SARSA. When training InferNet, we applied the same neural network setting as GridWorld. For policy induction, we explored different hyper-parameters and the final optimal one was a neural network of two dense layers with 256 units for each layer. During training, the batch size was 50 and the learning rate was 0.001. The number of epochs was 10 and the number of iterations was 50. For comparisons, we conducted 5-fold cross-validation and the dataset was split into 80% training and 20% test sets with an equal number of positive/negative shock trajectories.

7.3.2 Offline Evaluation

The effectiveness of critical policies was evaluated offline on the test dataset using two metrics: 1) septic shock rate and 2) the percentage of nudges. In general, we expect an effective critical

policy to have as low septic shock rate as the Full or Physician policies but with fewer nudges.

The *septic shock rate* r_{shock} was first used in [Rag17] and the assumption behind it is: when a septic shock prevention policy is indeed effective, the more the treatments in a patient trajectory agree with the induced policy, the lower the chance the patient would get into septic shock; vice versa, the less the treatments in a patient trajectory agree with the induced policy (more dissimilar), the higher the chance the patient would get into septic shock. In this analysis, we compared three critical policies against the Full and Physician policies by looking at septic shock rate for the 10% most similar group and 10% least similar group (most dissimilar). To do so, first, for each trajectory, a similarity rate r_s between the policy's action and the actual physicians' action is calculated. The higher the r_s, the more similar the RL policy is to the physicians' treatment. Then we sort the trajectories by their similarity rate in ascending order and calculate the septic shock rate for the top 10% of trajectories with the highest similarity rate, referred to as *10% most similar group* and the bottom 10% of trajectories with the least similarity rate, referred as *10% least similar group*. The septic shock rate is defined as: $r_{shock} = v_{shock}/v_s$, where v_s is the number of trajectories and v_{shock} is the number of positive-shock-trajectories.

The percentage of nudges reflects how often the critical policy would draw the physician's attention. We define a nudge as a decision where the state is critical and the physicians' action is different from the critical policy's action. Note that when identifying critical states, the LongTR threshold functions as a hyper-parameter which approximately determines the percentage of the critical decisions for which the physicians must follow the corresponding policy (the higher the LongTR threshold, the more states will be considered critical). In other words, it affects the percentage of nudges. In the offline evaluation, we explored the LongTR thresholds from 10% to 50% and investigated how fewer nudges are necessary to achieve the same effect with the Full policy.

7.4 Results

7.4.0.1 Septic Shock Rate

Figures 7.1 and 7.2 show the results of septic shock rate on CCHS and Mayo, respectively. In the X-axis, the suffix after critical policy indicates the LongTR threshold. For example, CriQN-DQN-0.1 means CriQN-DQN policy with a threshold of $LongTR = 10\%$. For each bar, the red column shows the septic shock rate of the 10% most similar group while the grey column indicates the 10% least similar group. For the critical policies, we considered the similarity in the critical states only. The black horizontal line represents the septic shock rate of the Full policy. In these results, we focus on the septic shock rate of 10% most similar group. Overall, the performance of critical policies increases as the LongTR threshold increases and the CriQN-DQN outperforms the other critical policies on both datasets. For CCHS, the CriQN-DQN policy beats the Full policy when $LongTR = 40\%$ while for Mayo, the CriQN-

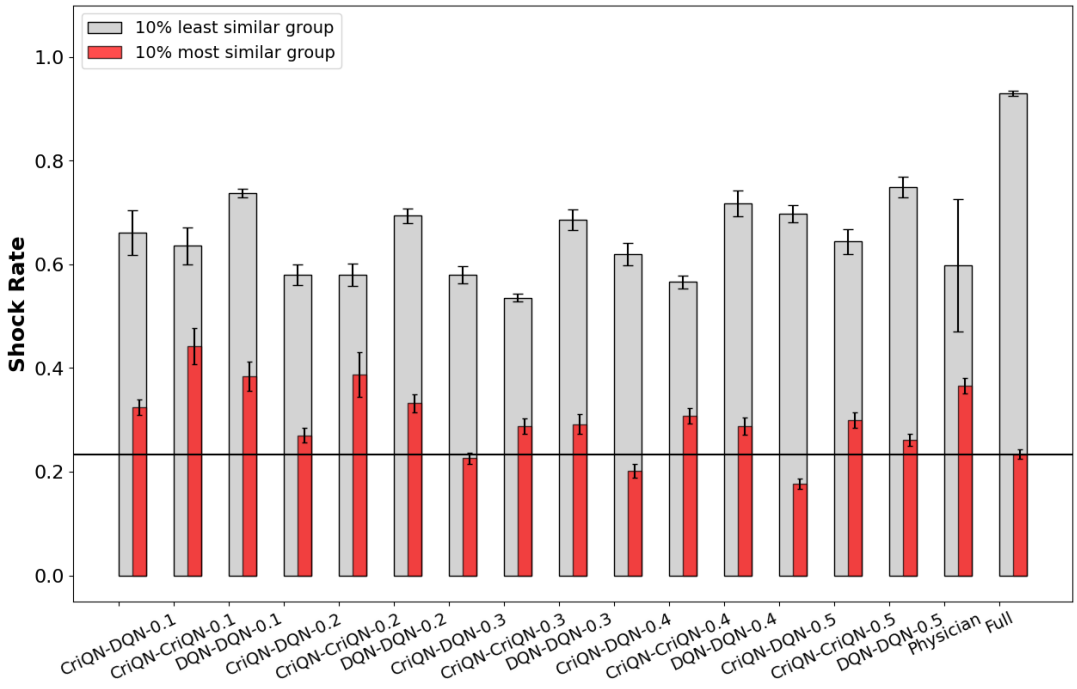


Figure 7.1: Septic Shock Rate for CCHS

DQN achieves the best performance when $LongTR = 50\%$, which is slightly better than the Full policy. Therefore, we could stop the LongTR threshold exploration at 40% for CCHS and 50% for Mayo. In the following results, we will focus on the comparison between critical policies ($LongTR = 40\%$ for CCHS and $LongTR = 50\%$ for Mayo) with two baseline policies: Full and Physician policies.

First, the Physician policy has the worst performance with the highest shock rate for the 10% most similar group and the lowest shock rate for the 10% least similar group on both datasets. It suggests that the RL-induced policy can provide better sepsis treatments than physicians. Second, the CriQN-DQN policy performs better than the Full policy in CCHS with the lower shock rate for the 10% most similar group. For Mayo, CriQN-DQN performs slightly better than Full but not much. Note that for the 10% least similar group, the result does not hold and this is because the similarity was calculated in the critical states only for the three critical policies but counted every state for the Full and Physician's policy. Overall, among all the three critical policies, CriQN-DQN is the best critical policy and can be as effective as a fully executed policy or even better.

7.4.0.2 Percentage of Nudges

Figure 7.3 shows the average percentage of nudges per trajectory identified by the corresponding policies in the test dataset. Similarly, the critical policies have a LongTR threshold of 40% for CCHS and 50% for Mayo. Note that CriQN-DQN vs. DQN-DQN apply the same DQN

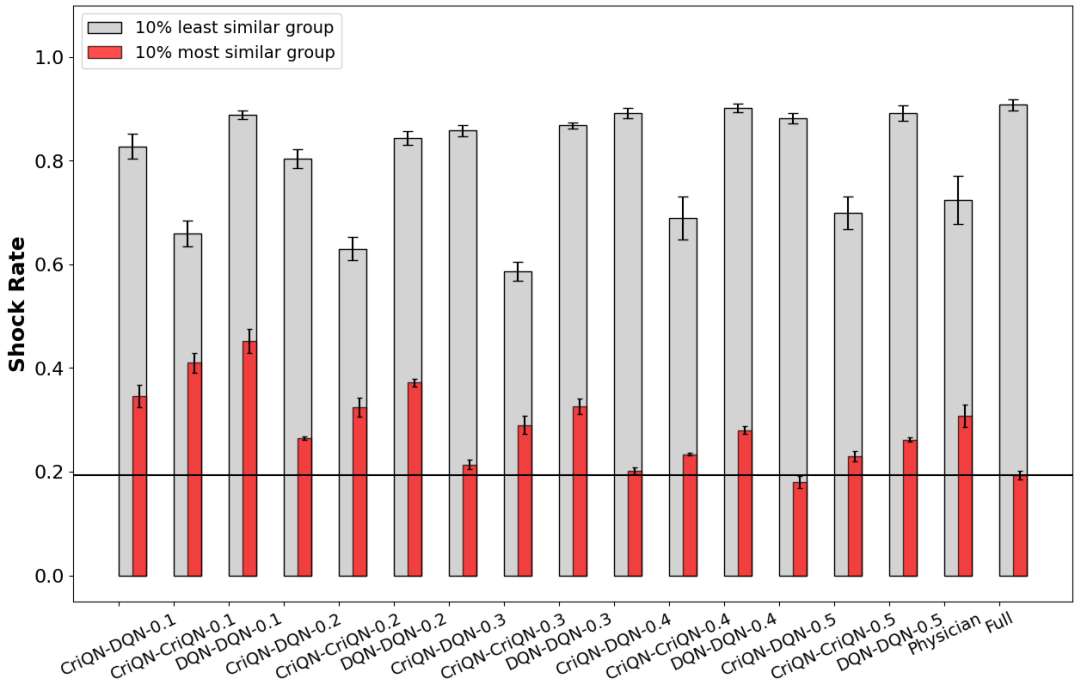


Figure 7.2: Septic Shock Rate for Mayo

policy to determine the optimal action while the only difference is on how they identify the critical states. In Figure 7.3, we split the patients into shock and non-shock groups for each bar. First, in both medical systems, all the policies have more nudges on shock patients than on the non-shock patients. This is reasonable because the shock patients have more severe moments and need more attention. Second, when comparing CriQN-DQN vs. DQN-DQN, CriQN-DQN has more nudges on the whole population and this comes from more nudges on shock patients. It suggests that CriQN is more reliable for the intensive attention to shock patients. Third, the Physician policy has more nudges than the others across all settings. It reflects the fact that the physicians are not always taking consistent treatments. Finally, when compared with the Full policy, the CriQN-DQN nudges 45% times and could save $37\% = (0.71 - 0.45)/0.71$ nudges on the whole population in CCHS while it only nudges less than 20% and saves 27% nudges in Mayo. More specifically, this saving mostly comes from the non-shock patients as it saves 66% in CCHS and 51% in Mayo. It is important to note that the number of non-shock patients (before sampling) is several times that of shock patients. So, the critical policy could save tremendous nudges in real life.

7.5 Conclusion

In this work, we evaluated the effectiveness of our LSTR framework on two real-world healthcare datasets: CCHS and Mayo. More specifically, we compared three critical policies

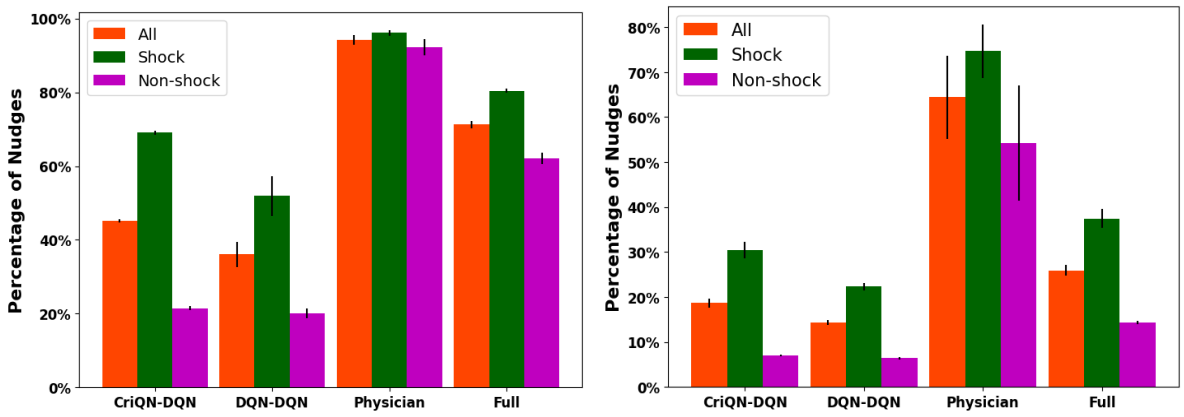


Figure 7.3: Percentage of nudges on different groups of patient. CCHS (Left) vs. Mayo (Right)

(CriQN-DQN vs. CriQN-CriQN vs. DQN-DQN) from two aspects: septic shock rate and percentage of nudges. The septic shock rate indicates whether the critical policy could prevent patients from getting septic shock. The percentage of nudges demonstrates how much the critical policy honors the physicians' decision-making. Our results on sepsis treatment show that the induced critical policy could reduce the percentage of nudges while keeping the septic shock rate as low as a fully executed policy. In summary, this work provides some evidence for employing our proposed general human-machine mixed-initiative decisionmaking framework in the healthcare domain where physicians are always overloaded and efficient alert is needed.

CHAPTER 8

CONCLUSION AND FUTURE WORK

8.1 Conclusions

In this dissertation, we propose and explored different Reinforcement Learning (RL) approaches to identify critical decisions in the sequential decision-making process. We explore three different RL approaches including offline off-policy evaluation (OPE), Adversarial Deep RL (ADRL) framework, and Long-Short Term Rewards (LSTR) framework in the context of Intelligent Tutor Systems and healthcare. This section summarizes the conclusion and findings of each approach as follows.

OPE Approach: This is our first attempt to identify critical decisions. In RL, the Q-value difference between two actions in a state indicates how much impact the decision will have on the final outcome. In other words, the larger the Q-value difference, the more important the corresponding decision is. Thus, in a historical student-ITS interaction log dataset, four groups of decisions (50% of overall decisions) are identified by four existing RL-induced policies based on their Q-value difference. The result shows that only one group of decisions is highly correlated with the students' learning performance, in which the more optimal decisions students received, the better they learned. In the meanwhile, the off-policy policy evaluation result shows that the corresponding RL-induced policy is effective in improving students' learning but the other three are ineffective. In this study, we find evidence of critical decisions that some decisions are highly correlated to the students' learning while some are not.

ADRL Framework: In this study, we consider the critical decisions from the perspective of necessity and sufficiency. Intuitively, a high Q-value difference is a sufficient condition for critical decisions. For the necessary condition, assume there are two decision-makers with opposite goals (increase/decrease the final outcome), if the current decision is critical, then the two decision-makers should have different choices. Based on this idea, we propose the ADRL framework that a pair of adversarial policies is induced with opposite goals: one is to

improve student learning while the other is to hinder, and the critical decisions are identified by comparing the two adversarial policies and using their corresponding Q-value differences. A classroom study is conducted to compare a Critical policy with a baseline Random policy. The Critical policy only takes optimal action in critical states but random actions in others. The empirical results show that there's no difference in students' learning performance between the two policies. However, we find that the critical decisions are always occurring in groups and the Critical policy is effective for the subgroup of students who experienced more critical decisions. In this study, we learn that the ADRL is too restrict in identifying critical decisions and the baseline random policy is not weak that it has a 50% chance to select the optimal action in the critical states because the tutor only has two actions.

LSTR Framework: Motivated by neuroscience, we propose LSTR framework to identify critical decisions. Furthermore, we propose a Critical-DQN algorithm to consider critical and non-critical decisions in the policy induction process. We evaluate the LSTR framework on three testbeds: an ideal GridWorld game, ITS, and healthcare. In the GridWorld, the results show that the LSTR framework indeed identifies critical decisions and the best critical policy CriQN-DQN is the one that applies Critical-DQN to identify critical states while executes the original DQN to determine optimal actions. This is because the Critical-DQN is sensitive to the critical states but needs more data to converge to the optimal policy. For ITS, we evaluate the LSTR framework through offline evaluation on a historical dataset and an empirical classroom study. Similarly, the offline evaluation result shows that the best critical policy is CriQN-DQN. Then, a classroom study is conducted to compare the Critical-Opt policy with two baseline policies: a Critical-Sub policy and a Full policy. Note that the Critical-Sub policy takes suboptimal actions in the critical states while random actions in the non-critical states. The empirical results show that 1) the Critical-Opt policy significantly outperforms the Critical-Sub policy and 2) the Critical-Opt policy performs as effectively as the Full policy. It demonstrates that the LSTR framework indeed identifies critical decisions in students' learning. Finally, for healthcare, we evaluate the effectiveness of the LSTR framework on two healthcare datasets from CCHS and Mayo. The result indicates that the best critical policy is still CriQN-DQN, which could identify the critical medical interventions in preventing patients from getting septic shock. Since the physicians are the primary decision-makers in septic treatment, the RL should not interfere with physicians too frequently. The percentage of nudges result indicates that the critical policy could dramatically reduce the number of nudges in non-shock patients but keep attention to the shock patients. Overall, the LSTR framework is effective in identifying critical decisions in both simulation environment (GridWorld) and real-world applications (ITS and healthcare).

8.2 Limitations

Although the experiments are carefully designed and conducted, I am still aware of some limitations listed as follows.

Off-policy Policy Evaluation: Although PDIS is one of the best OPE methods to evaluate the performance of policy in an offline way, it still has high variance and is not convincing compared to the online evaluation. To evaluate the effectiveness of the LSTR framework in real-world applications, online evaluation is often expensive or not ethical to conduct. In this work, we also tried PDIS to evaluate the critical policy performance in healthcare but didn't present the result. This is because the patient trajectory is too long so that the PDIS results are extremely big at the scale of 10^{20} and become meaningless. Instead of the septic shock rate, we still need to explore a typical RL offline evaluation method to estimate the performance of critical policies in long trajectories.

Data Efficiency in Critical-DQN: In chapter 5, the experiment on GridWorld shows that the Critical-DQN needs more data to converge to an optimal policy. So, the best critical policy is a combination of Critical-DQN and original DQN that one is used to identify critical states and the other one is used to select optimal actions. This is not efficient in RL policy induction that every time we have to induce two policies separately. It increases the complexity of tuning hyperparameters and the duration of inducing policy. The ideal case is to identify critical states and select optimal actions in a signal RL policy.

8.3 Future Work

First, the most difficult challenge in this work is policy evaluation in real-world applications. Unlike Atari games, in domains like healthcare and education, it's illegal to try and error on the patients or it's costly to executing the policy on real students. So, we mainly rely on the off-policy policy evaluation method to estimate the performance of policy instead of really executing it. One future direction is to explore a more robust and lower variance OPE method to generate more reliable evaluation results. This will close the loop from training to testing in the RL policy induction.

Second, in chapter 5, the CriQN-CriQN is slightly better than CriQN-DQN but not significant. It is possible that the optimal action for the critical state is not globally optimal. The global optimal action assumes that the agent will take optimal action on every step. However, it is not true in the real world that humans do not always make optimal decisions on every choice. Thus, if the agent takes random actions in the non-critical states, the optimal action in the critical states may be different from the global optimal action. Due to the data efficiency issue in the Critical-DQN algorithm, it is too risky to try the CriQN-CriQN on real-world applications. So, the second future direction is to improve the convergence of the Critical-DQN algorithm. We believe that the idea in the Critical-DQN algorithm is closer to the human

decision-making behavior and it should be the one that not only identifies critical states but also make critical decisions.

BIBLIOGRAPHY

[All19] Allan, J. L. et al. "Clinical decisions and time since rest break: An analysis of decision fatigue in nurses". *Health Psychology* **38**.4 (2019), pp. 318–324.

[Ami16] Amir, O. et al. "Interactive Teaching Strategies for Agent Training". *the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI* (2016), pp. 804–811.

[And95] Anderson, J. R. et al. "Cognitive tutors: Lessons learned". *The journal of the learning sciences* **4**.2 (1995), pp. 167–207.

[And18] Andrychowicz, M., Baker, B., et al. "Learning dexterous in-hand manipulation". *arXiv preprint arXiv:1808.00177* (2018).

[Azi19] Azizsoltani, H. et al. "Unobserved Is Not Equal to Non-existent: Using Gaussian Processes to Infer Immediate Rewards Across Contexts". *IJCAI* (2019), pp. 1974–1980.

[Bac17] Backer, D. & Dorman, T. "Surviving Sepsis Guidelines: A Continuous Move Toward Better Care of Patients With Sepsis". *JAMA* **317**.8 (2017).

[Bar08] Barnes, T. & Stamper, J. "Toward automatic hint generation for logic proof tutoring using historical student data". *Proceedings of the 9th International Conference on Intelligent Tutoring Systems*. 2008, pp. 373–382.

[Bar03] Barto, A. G. & Mahadevan, S. "Recent advances in hierarchical reinforcement learning". *Discrete event dynamic systems* **13**.1-2 (2003), pp. 41–77.

[Bec00] Beck, J. et al. "ADVISOR: A machine learning architecture for intelligent tutor construction". *AAAI/IAAI* **2000**.552-557 (2000), pp. 1–2.

[Bra99] Bransford, J. D. & Schwartz., D. L. "Rethinking Transfer: A Simple Proposal with Multiple Implications." *Review of Research in Education* (1999), pp. 61–100.

[Bud19] Budaev, S. et al. "Decision-Making From the Animal Perspective: Bridging Ecology and Subjective Cognition". Vol. 7. 2019, p. 164.

[Chi11] Chi, M. et al. "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 (2011), pp. 137–180.

[Cle16] Clement, B. & al., et. "A comparison of automatic teaching strategies for heterogeneous student populations". *EDM 16-9th EDM*. 2016.

[Clo96] Clouse, J. A. "On Integrating Apprentice Learning and Reinforcement Learning". *PhD thesis* (1996).

[Cor96] Cordova, D. I. & Lepper, M. R. "Intrinsic motivation and the process of learning: Beneficial effects of contextualization, personalization, and choice." *Journal of educational psychology* **88**.4 (1996), p. 715.

[Dud11] Dudik Miroslav, L. J. & Li, L. "Doubly Robust Policy Evaluation and Learning". *the 28th International Conference on Machine Learning* (2011).

[Fac17] Fachantidis, A. et al. "Learning to Teach Reinforcement Learning Agents". *Machine Learning and Knowledge Extraction* (2017).

[Fan14] Fan, M. & Jin, Y. *Obesity and Self-control: Food Consumption, Physical Activity, and Weight-loss Intention*. Vol. 36. 1. Applied Economic Perspectives and Policy, 2014, pp. 125–145.

[Fuj19] Fujimoto, S. et al. "Benchmarking Batch Deep Reinforcement Learning Algorithms". *arXiv preprint arXiv:1910.01708* (2019).

[Ham64] Hammersley, J. & Handscomb, D. "General Principles of the Monte Carlo Method". *Springer* (1964).

[Han18] Haniffa, R. et al. "Decision-making in the detection and management of patients with sepsis in resource-limited settings: the importance of clinical examination". *Crit Care* **22**.53 (2018).

[Hoo17] Hooey, B. L. et al. "The underpinnings of workload in unmanned vehicle systems". *IEEE Transactions on Human-Machine Systems* **48**.5 (2017), pp. 452–467.

[Hoo16] Hoonakker, P. et al. "Measuring workload of ICU nurses with a questionnaire survey: the NASA Task Load Index (TLX)". *IIE Trans Healthcare Syst Eng* **64** (2016), pp. 244–254.

[Igl09a] Iglesias, A. et al. "Learning teaching strategies in an adaptive and intelligent educational system through reinforcement learning". *Applied Intelligence* **31**.1 (2009), pp. 89–106.

[Igl09b] Iglesias, A. et al. "Reinforcement learning of pedagogical policies in adaptive and intelligent educational systems". *Knowledge-Based Systems* **22**.4 (2009), pp. 266–270.

[Jaq19] Jaques, N. et al. "Way off-policy batch deep reinforcement learning of implicit human preferences in dialog". *arXiv preprint arXiv:1907.00456* (2019).

[JRT07] Joel R. Tetreault Dan Bohus, D. J. L. "Estimating the Reliability of MDP Policies: A Confidence Interval Approach". *The Association for Computational Linguistics* (2007), pp. 276–283.

[Kal11] Kalenscher, T. & Wingerden, M. van. "Why We Should Use Animals to Study Economic Decision Making – A Perspective". *Frontiers in Neuroscience*. Vol. 5. 82. 2011.

[Koe97] Koedinger, K. R. et al. "Intelligent tutoring goes to school in the big city" (1997).

[Lev16] Levine, S. et al. “End-to-End Training of Deep Visuomotor Policies.” *JMLR* 17(39) (2016), pp. 1–40.

[Li11] Li, J. & Daw, N. D. *Signals in Human Striatum Are Appropriate for Policy Update Rather than Value Prediction*. Vol. 31. 14. 2011, pp. 5504–5511.

[Man14] Mandel, T. et al. “Offline policy evaluation across representations with applications to educational games”. *AAMAS*. 2014, pp. 1077–1084.

[McC04] McClure, S. M. et al. “Separate Neural Systems Value Immediate and Delayed Monetary Rewards”. *Science* (2004), pp. 503–507.

[McC07] McClure, S. M. et al. “Time Discounting for Primary Rewards”. *Neuroscience* (2007), pp. 5796–5804.

[McL11] McLaren, B. M. & Isotani, S. “When is it best to learn with all worked examples?” *AIED*. Springer. 2011, pp. 222–229.

[McL08] McLaren, B. M. et al. “When and how often should worked examples be given to students? New results and a summary of the current state of research”. *CogSci*. 2008, pp. 2176–2181.

[McL14] McLaren, B. M. et al. “Exploring the Assistance Dilemma: Comparing Instructional Support in Examples and Problems”. *Intelligent Tutoring Systems*. Springer. 2014, pp. 354–361.

[Mit13] Mitchell, C. M. et al. “Evaluating State Representations for Reinforcement Learning of Turn-Taking Policies in Tutorial Dialogue Christopher.” *SIGDIAL* (2013), pp. 339–343.

[Mni13] Mnih, V. et al. “Playing Atari with deep reinforcement learning.” *In NIPS Deep Learning Workshop* (2013).

[Mni15a] Mnih, V. et al. “Human-level control through deep reinforcement learning.” *Nature* 518 (2015), pp. 529–533.

[Mni15b] Mnih, V. et al. “Human-level control through deep reinforcement learning”. *Nature* **518**.7540 (2015), p. 529.

[Mni16] Mnih, V. et al. “Asynchronous Methods for Deep Reinforcement Learning.” *In ICLR* (2016).

[Mor06] Morris, G. et al. “Midbrain dopamine neurons encode decisions for future action”. *Nature Neuroscience* **9**.8 (2006), pp. 1057–1063.

[MB15] Mostafavi Behrooz, Z. L. & Barnes, T. “Data-driven Proficiency Profiling”. *Proc. of the 8th International Conference on Educational Data Mining*. 2015.

[MR09] MR, A. et al. “Involvement in decision-making and breast cancer survivor quality of life”. *Health Psychol* **28**.1 (2009), pp. 29–37.

[Naj14] Najar, A. S. et al. "Adaptive Support versus Alternating Worked Examples and Tutored Problems: Which Leads to Better Learning?" *UMAP*. Springer, 2014, pp. 171–182.

[Nar15] Narasimhan, K. et al. "Language understanding for text-based games using deep reinforcement learning". *arXiv preprint arXiv:1506.08941* (2015).

[Niv09] Niv, Y. "Reinforcement learning in the brain". *Journal of Mathematical Psychology* **53**.3 (2009), pp. 139–154.

[Ofs18] Ofstad, E. H. et al. "Clinical decisions presented to patients in hospital encounters: a cross-sectional study using a novel taxonomy". *BMJ Open* **8** (2018).

[PST16] Philip S. Thomas, E. B. "Data-Efficient Off-Policy Policy Evaluation for Reinforcement Learning". *In International Conference on Machine Learning*. 2016, pp. 2139–2148.

[Pig20] Pignatiello, G. A. et al. "Decision fatigue: A conceptual analysis". *Journal of Health Psychology* **25**.1 (2020), pp. 123–135.

[Pre00] Precup D., S. R. S. S. S. "Eligibility traces for off-policy policy evaluation". *17th International Conference on Machine Learning* (2000), pp. 759–766.

[Raf15] Rafferty, A. N. et al. "Inferring Learners' Knowledge From Their Actions". *Cognitive Science*. Vol. 39. 3. 2015, pp. 584–618.

[Raf16] Rafferty, A. N. et al. "Faster teaching via pomdp planning". *Cognitive science* **40**.6 (2016), pp. 1290–1332.

[Rag17] Raghu, A. et al. "Deep Reinforcement Learning for Sepsis Treatment". *arXiv preprint arXiv:1711.09602* (2017).

[Rea91] Real, L. A. "Animal Choice Behavior and the Evolution of Cognitive Architecture". *Science*. Vol. 253. 1991, pp. 980–986.

[Ren02] Renkl, A. et al. "From example study to problem solving: Smooth transitions help learning". *The Journal of Experimental Education* **70**.4 (2002), pp. 293–315.

[Rho17] Rhodes, A. et al. "Surviving Sepsis Campaign: International Guidelines for Management of Sepsis and Septic Shock: 2016." *Intensive Care Med* **43** (2017), pp. 304–377.

[Rob92] Robbins, A. "Awaken the Giant Within: How to Take Immediate Control of Your Mental, Emotional, Physical and Financial Destiny". New York: Simon Schuster, 1992.

[Roe07] Roesch, M. R. et al. "Dopamine neurons encode the better option in rats deciding between different delayed or sized rewards". *Nature Neuroscience* **10**.12 (2007), pp. 1615–1624.

[Sal10] Salden, R. J. et al. "The expertise reversal effect and worked examples in tutored problem solving". *Instructional Science* **38**.3 (2010), pp. 289–307.

[San15] Santos, L. R. & Rosati, A. G. "The evolutionary roots of human decision making". *Annual Review of Psychology*. Vol. 66. 2015, pp. 321–347.

[Sch17] Schwab, D. & Ray, S. "Offline reinforcement learning with task hierarchies". *Machine Learning* **106**.9-10 (2017), pp. 1569–1598.

[Sch09] Schwonke, R. et al. "The worked-example effect: Not an artefact of lousy control conditions". *Computers in Human Behavior* **25**.2 (2009), pp. 258–266.

[Seo08] Seo, H. & Lee, D. "Cortical mechanisms for reinforcement learning in competitive games". *Philosophical Transactions of the Royal Society B* (2008), pp. 3845–3857.

[She16a] Shen, S. & Chi, M. "Aim Low: Correlation-based Feature Selection for Model-based Reinforcement Learning." *EDM*. 2016, pp. 507–512.

[She16b] Shen, S. & Chi, M. "Reinforcement Learning: the Sooner the Better, or the Later the Better?" *Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization*. ACM. 2016, pp. 37–44.

[Sil16a] Silver, D. et al. "Mastering the Game of Go with Deep Neural Networks and Tree Search." *Nature* 529(7587) (2016), pp. 484–489.

[Sil16b] Silver, D. et al. "Mastering the game of Go with deep neural networks and tree search". *nature* **529**.7587 (2016), p. 484.

[Sil18] Silver, D. et al. "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play". *Science* **362**.6419 (2018), pp. 1140–1144.

[Sol16] Sollisch, J. "The cure for decision fatigue". *Wall Street Journal* (2016).

[Sta11] Stamper, J. C. et al. "Experimental evaluation of automatic hint generation for a logic tutor". *International Conference on Artificial Intelligence in Education*. Springer. 2011, pp. 345–352.

[Sul11] Sul, J. H. et al. "Role of rodent secondary motor cortex in value-based action selection". *Nature Neuroscience* **14**.9 (2011), pp. 1202–1208.

[Sut98] Sutton, R. S. & Barto., A. G. "Reinforcement Learning: An Introduction." *MIT Press* (1998).

[Sut18] Sutton, R. S. & Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

[Sut99] Sutton, R. S. et al. "Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning". *Artificial intelligence* **112**.1-2 (1999), pp. 181–211.

[Swe85] Sweller, J. & Cooper, G. A. "The use of worked examples as a substitute for problem solving in learning algebra". *Cognition and Instruction* **2**.1 (1985), pp. 59–89.

[Swi11] Swiger, P. A. et al. "Nursing workload in the acute-care setting: A concept analysis of nursing workload". *Nursing Outlook* **1**.2 (2011), pp. 131–43.

[Tho15] Thomas, P. "Safe Reinforcement Learning". *PhD thesis* (2015).

[Tor13] Torrey, L. & Taylor, M. E. "Teaching on a Budget: Agents Advising Agents in Reinforcement Learning". *International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '13* (2013), pp. 1053–1060.

[Uth03] Uther, W. & Veloso, M. "Adversarial reinforcement learning." *Technical Report.* (2003).

[VG11] Van Gog, T. et al. "Effects of worked examples, example-problem, and problem-example pairs on novices' learning". *Contemporary Educational Psychology* **36**.3 (2011), pp. 212–218.

[Van06] Vanlehn, K. "The behavior of tutoring systems". *International journal of artificial intelligence in education* **16**.3 (2006), pp. 227–265.

[Voh08] Vohs, K. D. et al. "Making Choices Impairs Subsequent Self-Control: A Limited-Resource Account of Decision Making, Self-Regulation, and Active Initiative". *Journal of Personality and Social Psychology*. 2008, pp. 883–898.

[Wan17] Wang, P. et al. "Interactive narrative personalization with deep reinforcement learning". *IJCAI*. 2017.

[Wan16] Wang, Z. et al. "Dueling Network Architectures for Deep Reinforcement Learning." *In ICLR* (2016).

[Wan07] Wansink, B. & Sobal, J. "Mindless eating: The 200 daily food decisions we overlook". *Environment and Behavior* **39**.1 (2007), pp. 106–123.

[Zha15] Zhang, F. et al. "Towards vision-based deep reinforcement learning for robotic motion control." *Australasian Conference on Robotics and Automation (ACRA)* (2015).

[Zha18a] Zhao, X. et al. "Deep Reinforcement Learning for Page-wise Recommendations." *In Proceedings of the 12th ACM Conference on Recommender Systems.* (2018), pp. 95–103.

[Zha18b] Zhao, X. et al. "Recommendations with Negative Feedback via Pairwise Deep Reinforcement Learning." *In Proceedings of the 24th ACM SIGKDD.* (2018).

[Zhe18] Zheng, G. et al. "DRN: A Deep Reinforcement Learning Framework for News Recommendation." *In Proceedings of the 2018 World Wide Web Conference on World Wide Web.* (2018), pp. 167–176.

[Zho15] Zhou & al., et. "The Impact of Granularity on Worked Examples and Problem Solving". *CogSci*. 2015, pp. 2817–2822.

[Zho17a] Zhou, G. & al., et. "Towards Closing the Loop: Bridging Machine-induced Pedagogical Policies to Learning Theories". *EDM*. 2017.

[Zho17b] Zhou, Z. et al. “Optimizing chemical reactions with deep reinforcement learning.” *ACS Cent. Sci.* (2017).

[Zhu17] Zhu, Y. et al. “Target-Driven Visual Navigation in Indoor Scenes using Deep Reinforcement Learning.” *ICRA* (2017).

[Zim13] Zimmer, M. et al. “Teacher-Student Framework: A Reinforcement Learning Approach”. *AAMAS Workshop Autonomous Robots and Multirobot Systems* (2013).