ABSTRACT

WANG, PENGCHENG. Deep Reinforcement Learning for Interactive Narrative Planning (Under the direction of Dr. James C. Lester).

Interactive narrative-centered educational games feature rich interactive story scenarios. They tightly integrate curricular content with engaging plots that are designed to naturally unfold during gameplay. To generate interactive narratives that are tailored to individual students, an intelligent interactive narrative planner can create personalized gameplay experiences that target effective learning outcomes. Data-driven interactive narrative planning shows particular promise for inducing adaptable interactive narrative planners. Data-driven methods can learn narrative planning policies from prior interaction experiences so that the learning outcome can be optimized.

Data-driven interactive narrative planning presents two major challenges. First, because training an interactive narrative planning policy requires enormous player interaction training corpora, there is the challenge of collecting enough data for both training and evaluating interactive narrative planning policies. Second, because interactive narrative planners’ interactions with players are usually complex and nonlinear, there is a significant challenge of modeling complex interaction patterns to learn effective interaction policies.

This dissertation research investigates a data-driven interactive narrative adaptation framework for educational games based on deep learning methods. A statistical predictive simulated player model is proposed to accurately imitate human players’ behaviors, which has two components: player actions and player outcomes. Specifically, deep recurrent and deep convolutional neural network models are investigated to encode the sequential interaction patterns between interactive narrative planners and players. With the proposed simulated player model, a deep reinforcement learning-based interactive narrative planner is
derived. The reinforcement learning framework naturally aligns with the sequential decision-making problem of interactive narrative adaptation, and the deep neural network function approximator endows narrative planners with the ability to learn highly complex interaction patterns.

Empirical results suggest that a deep recurrent highway network-based player action predictor and a deep convolutional neural network-based player outcome predictor construct the best performing simulated player model compared with multiple non-neural network baseline models and other deep learning models. These results support the idea that neural network depth is an important factor in building high-fidelity simulated player models. The effects of multiple regularization techniques are demonstrated in training player action and outcome predictors. With the high-fidelity simulated player model, multiple deep reinforcement learning-based interactive narrative planners are trained and evaluated. The results suggest that interactive narrative planning policy’s generalizability is correlated with reinforcement learning method’s exploratory strategy. More exploratory reinforcement learning methods derive more generalizable interactive narrative planning policies when evaluation environment is very different from training environment. Further, recurrent neural networks offer advantages in learning effective reinforcement learning-based interactive narrative planning policies when the evaluation environment is similar to the training environment. Stabilization techniques promote the convergence of reinforcement learning-based interactive narrative planner training, yielding more robust interactive narrative planning policies.
Deep Reinforcement Learning for Interactive Narrative Planning

by

Pengcheng Wang

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

2018

APPROVED BY:

__________________________________________________________
Dr. James C. Lester
Committee Chair

__________________________________________________________
Dr. Bradford W. Mott

__________________________________________________________
Dr. Min Chi

__________________________________________________________
Prof. Patrick FitzGerald
DEDICATION

Dedicated to my parents and fiancée who have shown trust, support, and love in my life.
**BIOGRAPHY**

Pengcheng Wang was born in Xuanhua, China. He entered Beijing University of Technology in 2006, where he majored software engineering. At BJUT he was awarded three undergraduate student scholarships from 2007 to 2009, the student technical innovation award, and the outstanding undergraduate student award in 2009. He served as the president of the Microsoft Club of School of Software Engineering from 2008 to 2009, when he led a team to win third prize in the Microsoft Imagine Cup software development contest with the Digital Farm project, a solution to effectively collect and analyze agricultural data and connect farmers and experts. An upgraded version of the project won the third prize in the Second China College Students’ Software Development Innovation Competition.

Pengcheng entered the Department of Computer Science at Beijing University of Technology to pursue his master’s degree in 2010. Advised by Dr. Baocai Yin, he studied real-time water rendering for particle systems-based fluid simulation. He published multiple journal and conference papers and was recognized with the graduate student scholarship in 2011 and the Outstanding Master’s Thesis Award in BJUT in 2013.

Pengcheng joined the Department of Computer Science at North Carolina State University in 2013 to pursue his doctoral degree. Because of his interest in machine learning, he joined the IntelliMedia group and studied with Dr. James Lester. His research spans artificial intelligence, machine learning and personalized learning technologies. He has worked extensively on the problem of interactive narrative adaptation in educational games, which have led to publications at top-tier AI conferences.
ACKNOWLEDGMENTS

I would like to sincerely thank my advisor, Dr. James Lester, for his insightful advice on my research work, as well as his perfect demonstration of being a successful person. James encourages me to pursue the research field of my interest and is always supportive of my research efforts. During my PhD study, I was able to have multiple competitive publications under his guidance. Many times he directed my paper writing in late evenings or weekends, for which I cannot thank him enough. James is a very efficient person. His calendar is always full. He can finish many tasks across multiple projects every day. He walks along the greenway at Lake Johnson much faster than I do. I am fortunate to have a seat close to his office and see how he works. I learned a lot from him.

I want to thank all the people in the IntelliMedia group at NCSU. I collaborated extensively with Dr. Jon Rowe on both academic research and project development. Jon helped me start my research in interactive narrative planning and generously shared his knowledge of this field with me. Dr. Brad Mott also helped me better understand the field of interactive narrative. I attended my first academic conference during my PhD together with Brad at Los Angeles. Dr. Wookhee Min and Andy Smith supported me with their experience studying and conducting research in the IntelliMedia group. I followed their guidance on completing the milestones in the PhD program. I want to thank all the other PhD students in the IntelliMedia group, including Robert Sawyer, Bita Akram, Ziwei Wu, and Kyungjin Park. It has been great to have all these talks on research, interesting findings in AI, and all fun things including football, basketball and soccer. I’d like to thank the development team, as well as all the art students in IntelliMedia. My experience of working together with Kirby Culbertson, Robert Taylor, Barry Liu, Dawoda Kah, Amanda Dean, Jennifer Sotoamaya and
Margo Jordan was so cool that we together created an awesome 3D game, Inspire. I cannot thank all my other professors and friends enough in the Department of Computer Science. I appreciate all the help and support to make finish my PhD study.

I want to thank my parents, my fiancée Wei Zhang and all my family members for their persistent encouragement and continuous support. The weekly one-hour video call to my family is the most important thing to me since I came the US.
# TABLE OF CONTENTS

LIST OF TABLES ..................................................................................................................... viii

LIST OF FIGURES .................................................................................................................. ix

CHAPTER 1 INTRODUCTION .......................................................................................... 1

1.1 Problem ......................................................................................................................... 3

1.2 Approach ....................................................................................................................... 5

1.3 Thesis Statement and Research Questions .................................................................. 9

1.4 Organization .................................................................................................................. 12

CHAPTER 2 BACKGROUND AND RELATED WORK .................................................. 14

2.1 Interactive Narrative Generation and Adaptation ..................................................... 14

2.2 Simulated Player Model .............................................................................................. 21

2.3 Deep Learning ............................................................................................................ 28

2.4 Reinforcement Learning ............................................................................................. 34

CHAPTER 3 CRYSTAL ISLAND INTERACTIVE NARRATIVE .......................... 40

3.1 CRYSTAL ISLAND Backstory and Game Environment ........................................... 40

3.2 CRYSTAL ISLAND Player Interaction Representation .......................................... 42

3.3 Narrative Adaptation in CRYSTAL ISLAND ........................................................... 45

3.4 CRYSTAL ISLAND Interactive Narrative Corpus .................................................... 47

CHAPTER 4 STATISTICAL PREDICTIVE SIMULATED PLAYER MODEL .... 49

4.1 Simulated Player Modeling in CRYSTAL ISLAND .................................................... 49

4.2 Neural Network Models for Sequence Data Modeling ............................................ 53

4.3 Long Short-Term Memory Network for Sequence Data Modeling ....................... 56

4.4 Deep Recurrent Neural Networks for Sequence Data Modeling ......................... 58

4.5 Deep Convolutional Neural Networks for Sequence Data Modeling .................. 61

4.6 Regularization Techniques in Neural Network Models .......................................... 63

  4.6.1 Dropout ................................................................................................................... 64

  4.6.2 Mixture of Experts ............................................................................................... 65

  4.6.3 Multi-Task Learning ............................................................................................ 67

CHAPTER 5 REINFORCEMENT LEARNING-BASED INTERACTIVE NARRATIVE PLANNING ................................................................................................................... 70

5.1 Reinforcement Learning Framework for Interactive Narrative Planning .............. 70

5.2 Q-Network-Based Interactive Narrative Planning .................................................. 73

  5.2.1 Deep Reinforcement Learning Stabilization ....................................................... 74

  5.2.2 Recurrent Q-Network ........................................................................................ 76

  5.2.3 Dueling Q-Network ............................................................................................ 78

5.3 Actor-Critic Network-Based Interactive Narrative Planning ................................ 80

5.4 Exploratory Strategy in Interactive Narrative Planning ......................................... 81
# TABLE OF CONTENTS

**CHAPTER 6 EVALUATION** .................................................................................................................. 84
6.1 Statistical Simulated Player Model Evaluation ........................................................................ 84
6.2 Reinforcement Learning-Based Interactive Narrative Planner Evaluation .......................... 90

**CHAPTER 7 DISCUSSION** .................................................................................................................. 99
7.1 Constructing High-Fidelity Simulated Player Models ............................................................ 99
7.2 Deriving Interactive Narrative Planners .................................................................................. 102
7.3 Limitations ..................................................................................................................................... 104

**CHAPTER 8 CONCLUSIONS AND FUTURE WORK** ................................................................. 106
8.1 Research Questions Revisited .................................................................................................. 106
8.2 Summary ....................................................................................................................................... 112
8.3 Future Work ............................................................................................................................... 115

**REFERENCES** .................................................................................................................................... 116
LIST OF TABLES

Table 3.1: CRYSTAL ISLAND players’ gameplay actions.................................................................43
Table 6.1: Player action prediction performance of non-deep learning models ..................86
Table 6.2: Player action prediction performance of deep learning models ..........................87
Table 6.3: Player outcome prediction performance of non-deep learning models ..........88
Table 6.4: Player outcome prediction performance of deep learning models .................89
Table 6.5: Interactive narrative planning policy evaluation – RL methods with distinct exploratory strategies..........................................................................................................................92
Table 6.6: Interactive narrative planning policy evaluation – value function-based RL methods........................................................................................................................................93
LIST OF FIGURES

Figure 2.1: Plot graph of the interactive fiction, *Anchorhead* ........................................15
Figure 2.2: Architecture of a spoken dialogue system with simulated users....................26
Figure 3.1: *CRYSTAL ISLAND* interactive narrative .....................................................40
Figure 3.2: *CRYSTAL ISLAND* open-world game environment ....................................42
Figure 4.1: Example of interactive narrative planner action timeline and player action
            timeline ........................................................................................................52
Figure 4.2: RNN structure and its unfolding along time axis ............................................55
Figure 4.3: 2D Grid-LSTM unit .......................................................................................60
Figure 4.4: RHN model for player action prediction .........................................................61
Figure 4.5: CNN model for player outcome prediction ......................................................63
Figure 4.6: Mixture of Experts model structure .................................................................66
Figure 4.7: Multi-task LSTM network implementation of a simulated player model for
            interactive narrative planning ............................................................................69
Figure 5.1: Deep reinforcement learning-based interactive narrative personalization
            framework ........................................................................................................72
Figure 5.2: Neural network structures of Q-network, Recurrent Q-network, and Dueling Q-
            network ........................................................................................................80
CHAPTER 1

INTRODUCTION

Interactive narrative is a form of interactive experience which allows players to create or influence a dramatic storyline during the interaction between the players and the interactive narrative systems, by assuming the role of a player character in the narrative environment, or directly allowing players to manipulate the narrative environment state (Riedl & Bulitko, 2012). Interactive narrative systems afford players the opportunities to be able to fundamentally affect the unfolding story and the interaction outcomes. Narrative intelligence (Blair & Meyer, 1997; Mateas & Sengers, 1999), which studies the ability to organize experiences in narrative forms, have been a research focus in artificial intelligence and cognitive science communities for a long time. Narrative intelligence tasks, such as story understanding (Wilensky, 1981; Cullingford, 1981; Cox, 1996), story generation (Gervás, 2009; Young et al., 2004; Li et al., 2012), and commonsense reasoning (Mott & Lester, 2006; Riedl & Stern, 2008; Young & Riedl, 2003) present significant promises and challenges.

Although largely applied in entertainment settings (Thue & Bulitko, 2007; Sharma & Ontanon, 2010; McCoy et al., 2011; Porteous et al., 2010, Mateas & Stern, 2003), interactive narratives have demonstrated great potentials and promising results for educational and training purposes (Mott & Lester, 2006; Riedl et al., 2008; Rowe et al., 2011; Rowe et al., 2014; Rowe & Lester, 2015; Wang, Rowe, Mott, & Lester, 2016, Wang et al., 2017a, Wang et al., 2017b, Wang et al., 2018). Educational interactive narratives incorporate curriculum content within the unfolding story, so that players experience entertaining interaction while fulfill the educational purposes. Players’ (often students’) narrative interactions are usually in the form of problem solving embedded in stories (Rowe et al., 2011), or immerse interactions
in narrative scenarios created by experience managers (Riedl & Stern, 2008). Empirical
evidence has revealed interactive narrative-centered games’ capabilities in improving
players’ (often students’) engagement and learning outcomes in educational settings (Lester
et al., 2013).

To appropriately generate or adapt interactive narratives, a wide range of
computational methods, such as search-based drama management (Bates, 1992; Weyhrauch
1997; Lamstein & Mateas, 2004; Nelson & Mateas, 2005; Garber-Barron & Si, 2013),
STRIPS-planning (Porteous et al., 2015; Robertson & Young, 2015), and reinforcement
learning (Nelson et al., 2006; Roberts et al., 2006; Harrison & Riedl, 2016; Rowe et al.,
2014; Rowe & Lester, 2015; Wang, Rowe, Mott, & Lester, 2016, Wang et al., 2017a, Wang
et al., 2017b) have been investigated and evaluated. Data-driven approaches to interactive
narrative generation have been the topic of increasing interest. A broad range of machine
learning methods have demonstrated promise in inducing interactive narrative planners to
create innovative narrative scenarios and adaptive gameplay experiences. Dynamic Bayesian
networks have been used to model director agents’ decisions in interactive narrative
generation (Lee et al., 2014), and prefix-based collaborative filtering has been utilized to
predict players’ preferences to story branches (Yu & Riedl, 2014). Crowdsourcing has been
employed to construct plot graphs for deriving new stories (Li et al., 2013), and
reinforcement learning has been used to tailor interactive narratives to individual players in
game-based learning environments (Rowe et al., 2014; Wang, Rowe, Mott, & Lester, 2016).
Compared to classical search-based methods and STRIPS-planning methods, data-driven
interactive narrative approaches relax the restriction that the designers of the interactive
experience manager should understand the dynamics of the narrative environment. Data-
driven methods can fully or partially learn the narrative environment dynamics, player interaction preferences, and player’s interaction outcome tendencies from prior interaction experiences. This property makes data-driven method an appropriate choice to derive adaptable interactive narrative experience managers in open-world narrative environment, which has limited restriction on player’s gameplay behavior; as well as educational narratives, in which narrative evaluation does not have to correlate with the narrative structure.

1.1 Problem

Despite the great promise, data-driven interactive narrative generation faces substantial challenges. The first major challenge is that data-driven interactive narrative generation approaches often depend on high volumes of player interaction data, largely because of the significant uncertainty inherent in the complex interactions between interactive narrative experience managers and human players. Because collecting high-quality player data is resource intensive, it is common to see small datasets from dozens or hundreds or thousands of players utilized in interactive narrative generation works (Li et al., 2013; Yu & Riedl, 2014; Lee et al., 2014; Rowe et al., 2014; Harrison & Riedl, 2016; Wang, Rowe, Mott, & Lester, 2016). Small corpora prohibit interactive narrative experience manager to fully explore the narrative space, and insufficient samples of player—narrative experience manager interactions introduce extra risks in modeling the high uncertainty of the interactive narrative environment. Although several approaches have been proposed and experimented to cope with the problem, they introduced other potential problems by either simplifying the problem (Rowe et al., 2014; Wang, Rowe, Mott, & Lester, 2016), or utilizing synthetic data that have not been validated with human players (Nelson et al., 2006; Roberts et al., 2006), or
designing and adopting data collection protocols that are not easy to generalize to other domains (Li et al., 2013; Harrison & Riedl, 2016). So, to study the problem of interactive narrative generation using data-driven methods, generating reliable and sufficient data is a major problem to solve.

The second major challenge of data-driven interactive narrative generation and adaptation emphasized in this dissertation, given the assumption that training data can be unlimited using simulation, is how to construct an effective interactive narrative planner which is able to adapt narratives to each individual player, so that each player’s interaction outcome (learning outcome in educational setting) can be maximized. Reinforcement learning (Sutton & Barto, 1998) techniques have shown particular promise in interactive narrative generation and personalization because of their inherent support for sequential decision-making under uncertainty with delayed rewards. Computational models of interactive narrative generation should account for uncertainty because human players’ actions are poorly understood and difficult to predict. Further, interactive narrative generation involves delayed rewards because judgments about interactive narratives’ quality are often deferred until after story episodes are complete. Despite this natural alignment, applications of RL in interactive narratives are not intuitive. Prior work on RL-based interactive narrative generation has often utilized low-dimensional state features and linear RL models to account for the limited availability of training data, particularly where interaction data from human players is involved (Rowe et al., 2014; Wang, Rowe, Mott, & Lester, 2016). Alternatively, synthetic data generated from simulations of human players have been utilized, although these models were relatively simple and have not been thoroughly validated (Nelson et al., 2006). These design decisions introduced a severe
restriction in RL-based interactive narrative planning to identify complex non-linear player interaction patterns that are relevant to effectively personalize interactive narratives. In this dissertation, the major research question about how to construct effective reinforcement learning-based interactive narrative planner will be thoroughly studied and evaluated.

1.2 Approach

In order to obtain enough data to study the problem of interactive narrative adaptation using data-driven methods, a statistical predictive simulated player modeling method is proposed and investigated in this dissertation. Player’s narrative interaction behavior is represented as a combination of player actions and player outcomes. A statistical predictive simulated player model imitates player’s behavior by predicting player’s action at each player interaction time step according to player’s action history, player traits, and interactive narrative experience manager’s past decisions. When the interaction reaches to the end, the simulated player model predicts the interaction outcome of the player, which represents the interactive narrative quality, also reflects the capability of the interactive narrative experience manager, whose purpose is to adapt the unfolding of the narrative to maximize the interaction result. Such a predictive simulated player model has several advantages over the utilization of raw human player generating dataset (Rowe et al., 2014; Wang, Rowe, Mott, & Lester, 2016) and assumption-based synthetic data (Nelson et al., 2006; Roberts et al., 2006). Compared to raw human players generating dataset, a simulated player model can generate infinite amount of data, which offers the interactive narrative experience manager the opportunity to thoroughly explore the narrative generation space utilizing abundant interaction features. Also, a predictive simulated player model can generate player interaction data in the online form, which enables online learning conducted by the interactive narrative
experience managers. Compared to utilizing assumption-based synthetic data in learning adaptable interactive narrative experience manager, the statistical predictive simulated player model is easier to be validated, which reveals how accurate the simulated player model is to imitate human player’s behavior. Adopting a statistical predictive simulated player model is especially helpful when player action distribution has large variance, e.g., in open-world interactive narrative interaction environment where players are relatively free to move or talk; or in case the player outcome distribution has large variance, e.g., in educational setting when learning outcome is the adaptation measurement metric. Although this category of simulated player modeling method has not been broadly utilized in the field of interactive narrative generation and adaptation, long-term research and prominent achievements have been seen in some other fields, like spoken dialogue system, in which statistical user simulation serves an important role in training and evaluating reinforcement learning-based dialogue managers (Schatzmann et al., 2006; Young et al., 2013). The success of the broad usage of statistical user simulation sheds light on its prominence in serving both training and evaluation of data-driven experience managers in interactive narrative environments.

To solve the problem of interactive narrative generation and adaptation, reinforcement learning (RL) is adopted and investigated as the method to derive interactive narrative experience managers in this dissertation. Following the definition in (Young & Riedl, 2003), an intelligent interactive narrative planner controls the unfolding of a story in the interaction between players and the narrative environment, especially when player’s behavior deviates substantially from the story’s intended structure, which may influence player’s interaction outcome. According to this definition, from now on, the term interactive narrative planner (Young & Riedl, 2003) will be adopted to specifically indicate the intelligent
component in an interactive narrative system controlling the unfolding of the story to maximize the expected interaction outcome. The term of interactive narrative experience manager (Riedl & Bulitko, 2012) used above, which may represent a broader definition of intelligent components (e.g., discourse planner, text generation system, etc.) in interactive narrative systems will no longer be adopted for the purpose of clarity, even though the proposed reinforcement learning framework should also work in constructing other intelligent component in interactive narrative systems if properly configured. Despite the natural alignment of the reinforcement learning framework with the interactive narrative adaptation problem, deriving an effective reinforcement learning-based interactive narrative planner to personalize stories is not intuitive at all. A key open question in RL-based interactive narrative generation is how to model complex player interaction patterns to learn effective policies. In this dissertation, a deep reinforcement learning-based interactive narrative generation framework that leverages synthetic data produced by simulate player model is investigated. A neural network function approximator utilized in deep reinforcement learning methods enables the extraction of nonlinear interaction features, which potentially increases the capability of the reinforcement learning agent to learn more complicated interactive narrative adaptation policies, even though extra difficulties can be introduced to properly train a deep reinforcement learning narrative adaptation system. In recent years, significant results have been achieved in the field of deep reinforcement learning, and new techniques have been proposed to stabilize the training process of deep reinforcement learning agent (Lin 1993; Mnih et al., 2015; Mnih et al., 2016; Schulman et al., 2015; Schulman et al., 2017), formulating observations or rewards in innovative modes (Hausknecht & Stone, 2015; Andrychowicz et al., 2017), and utilizing different forms of
reinforcement learning policy and optimization methods (Mnih et al., 2015; Wang, Schaul, Hessel, Van Hasselt, Lanctot, & De Freitas, 2016; Mnih et al., 2016; Schulman et al., 2015; Schulman et al., 2017; Wu et al., 2017; Lillicrap et al., 2015), so that even human-level performance can be achieved in some commonly adopted reinforcement learning problems (Mnih et al., 2015; Silver et al., 2016; Silver et al., 2017). All of these recent achievements motivate the work in this dissertation to utilize state-of-the-art reinforcement learning techniques in adapting interactive narratives.

In this dissertation, for both the tasks of statistical predictive simulated player modeling and data-driven adaptable interactive narrative planner derivation, deep learning (Schmidhuber, 2015; LeCun et al., 2016; Goodfellow et al., 2016) is the major machine learning technique to be investigated and evaluated. Deep learning methods, in the machine learning family, are representation-learning methods with multiple levels of representation (LeCun et al., 2016; Goodfellow et al., 2016), in which higher level representation is usually composed of lower level representations with nonlinear transformation. In recent years, when the term deep learning is mentioned, it usually means deep learning in artificial neural networks. In this dissertation, the term deep learning is used as the synonym as deep learning in artificial neural networks. Deep learning has gained great breakthroughs in several artificial intelligence fields in recent years. Significant improvement over prior state-of-the-art, or even human-level performance have been achieved in subjects like image recognition (Krizhevsky et al., 2012; Farabet et al., 2013; Tompson et al., 2014; Szegedy et al., 2015; Simonyan & Zisserman, 2015; He et al., 2016; Huang et al., 2017), object detection, localization and segmentation (Girshick et al., 2014; He et al., 2014; Girshick, 2015; Ren et al., 2015; He et al., 2017), speech recognition (Jaitly et al., 2012; Sak et al., 2014; Deng et al.,
and generation (Oord et al., 2016; Ank et al., 2017; Wang, Skerry-Ryan, Stanton, Wu, Weiss, Jaitly, Yang, Xiao, Chen, Bengio & Le, 2017; Mehri et al., 2016; Sotelo et al., 2017), natural language processing and understanding (Kumar et al., 2016; Mikolov et al., 2013; Pennington et al., 2014; Hermann et al., 2015), machine translation (Bahdanau et al., 2014; Cho et al., 2014; Luong et al., 2015; Wu et al., 2016), reinforcement learning (Mnih et al., 2015; Silver et al., 2016; Silver et al., 2017), human intention and plan recognition (Min et al., 2014; Min et al., 2016; Min et al., 2017), and application in medicine and biology (Ciodaro et al., 2012; Helmstaedter et al., 2013; Leung et al., 2014; Xiong et al., 2015), etc. Specifically, for the tasks of statistical predictive simulated player modeling and data-driven interactive narrative planner derivation, deep learning offers significant potential because some neural network models properly fit the sequential pattern recognition task in the player behavior prediction and player—interactive narrative planner interaction. An effective machine learning model should be able to utilize the property that features at each time step is with the same format. In the neural network family, several types of models, like recurrent neural networks and convolutional neural networks, can effectively utilize this regionally repeated feature map to efficiently learn sequential patterns. In this dissertation, both recurrent neural networks (Siegelmann, 1992; Schmidhuber, 1990) and convolutional neural networks (LeCun et al., 1989; LeCun et al., 1990) will be deeply investigated for constructing high-fidelity statistical simulated players and effective data-driven interactive narrative planners.

1.3 Thesis Statement and Research Questions

In this dissertation, a statistical predictive simulated player model is proposed to imitate human player’s behavior while they play an interactive narrative-centered educational game.
Based on the proposed simulated players, various deep reinforcement learning techniques are investigated on their effectiveness in deriving adaptable interactive narrative planners to maximize player’s interaction outcomes. More specifically, focusing on the investigation of effective machine learning models and methods to achieve the previously proposed targets, the following thesis statement is studied in this dissertation:

*Deep learning offers effective models to predict human players’ interaction actions and interaction outcomes if proper neural network structures, optimization and regularization techniques are utilized. Deep reinforcement learning trains effective adaptable interactive narrative planners to personalize interactive narratives to individual players in educational games.*

The assessment of the statistical simulated player models majorly relies on their prediction performance. Because the way predictive simulated players generating synthetic interaction data is through estimating player’s future actions and interaction outcomes based on interaction history, the measurement is based on prediction accuracy and macro-average F1 score. Player action is going to be predicted and evaluated at each player interaction time step, while player outcome is estimated and evaluated at the end of the gameplay interaction. To assess the effectiveness of the proposed deep reinforcement learning-based adaptable interactive narrative planner, the metric of narrative planning policy value is adopted. The narrative planning policy value reflects the expectation of interaction outcomes between an interactive narrative planner adopting a specific narrative planning policy and the population of players. Narrative planning policy value can be estimated by averaging the interactive narrative planning interaction outcomes of the planner with simulated players for a certain
number of episodes. Policy value is a standard metric utilized in reinforcement learning to evaluate the effectiveness of derived policies.

Specifically, in this dissertation, deep learning is the major machine learning technique investigated and evaluated using these metrics. Although deep learning achieves great performances in plenty of fields as mentioned Section 1.2, it is not an easy-to-use off-the-shelf machine learning toolbox. To obtain good results in the task of interactive narrative adaptation, several technical aspects in constructing deep learning models, e.g. deep neural net structure, depth of network, regularization, optimization, and convergence property etc. are necessary to carefully discuss. With these major topics mentioned so far, the following research questions (RQs) and sub-questions are proposed and investigated in this dissertation:

• RQ 1: Is deep learning an effective choice to construct data-driven simulated player models?
  
  • RQ 1.1: Can deep learning outperform commonly utilized machine learning methods (baselines) in both tasks of player action prediction and player outcome prediction?
  
  • RQ 1.2: Can deep learning offer proper model to deal with sequential player—narrative planner interaction patterns in simulated player modeling?
  
  • RQ 1.3: Does deep neural network models outperform shallow models in constructing simulated player models?
  
  • RQ 1.4: What regularization methods are necessary to improve deep learning methods’ generalization performance in simulated player modeling?
• RQ 2: Can deep reinforcement learning derive effective interactive narrative planners in narrative-centered educational games to personalize narratives to individual players?
  • RQ 2.1: Is deep reinforcement learning a tractable method in training interactive narrative planners?
  • RQ 2.2: Can deep reinforcement learning outperform linear reinforcement learning method (baseline) in training adaptable interactive narrative planners?
  • RQ 2.3: What are proper neural network structures to form a reinforcement learning-based interactive narrative planner?
  • RQ 2.4: How does deep reinforcement learning-based interactive narrative planner perform in interaction with unfamiliar player population (generalization performance)?

The proposed interactive narrative adaptation work is investigated and analyzed on an interactive narrative-centered educational game—CRYSTAL ISLAND. One dataset collecting from multiple studies involving more than 400 players is utilized in simulated player modeling. The CRYSTAL ISLAND narrative context is also adopted in constructing the interactive narrative generation and adaptation environment. Performance of all modules mentioned above is finally evaluated using this CRYSTAL ISLAND testbed.

1.4 Organization

The remainder of this proposal is organized as follows. Chapter 2 provides background on the task of data-driven interactive narrative generation and adaptation. This includes a discussion of related work on interactive narrative generation utilizing various data sources, statistical simulated player/user modeling, deep learning models adopting sequential data, as well as deep reinforcement learning techniques in solving sequential decision making
problems. Chapter 3 presents the interactive narrative adaptation test bed, CRYSTAL ISLAND, which offers an open world game environment for player interaction and the interactive narrative context for the narrative planner’s adaptation. Chapter 4 describes the statistical predictive simulated player model proposed in this dissertation, and the deep neuron networks that can properly model player’s actions and outcomes for simulated players. Chapter 5 describes reinforcement learning-based interactive narrative planning investigated in this work. Specifically, the way to construct deep reinforcement learning models, techniques to represent policies, methods to stabilize training, and the strategies to control narrative planners’ exploratory behaviors are discussed. Chapter 6 presents evaluation results on the CRYSTAL ISLAND testbed with a corpus from multiple studies involving more than 400 players. Chapter 7 follows with a discussion of the experiment results, and Chapter 8 concludes the dissertation by revisiting the research questions proposed in Section 1.3, as well as discussing future work.
CHAPTER 2

BACKGROUND AND RELATED WORK

The proposed data-driven interactive narrative adaptation framework involves work in several active research fields: (1) interactive narrative generation and adaptation, which drafts and tailors stories with computational methods, (2) player simulation techniques, which establish simulated users to imitate human user behaviors, (3) deep learning techniques, which utilize deep neural networks to formulate machine learning models, (4) reinforcement learning techniques, which offer the framework to solve sequential decision making problems in uncertain environments. The remainder of this section describes background and related works from the aspects of these four research fields and attempts to clarify where the proposed framework stays in these mentioned research fields. Section 2.1 presents the previously studied computational methods in interactive narrative generation and adaptation. Section 2.2 describes data sources adopted in prior interactive narrative generation and adaptation works, with a focus on user simulation techniques utilized in this field and other similar fields. Section 2.3 briefly demonstrates the recent success of the deep learning techniques, which motivates the choice of the machine learning tools utilized in this dissertation. Section 2.4 depicts the recent advancement of the reinforcement learning technique, especially the branch of deep reinforcement learning research.

2.1 Interactive Narrative Generation and Adaptation

Interactive narrative, as the most compelling form of narrative intelligence, has gained plenty of attention in recent years (Riedl & Bulitko, 2012). A large amount of research works, utilizing various computational approaches in generating interactive narratives, have shown great achievements and promises. Intelligent computational methods have
been adopted to balance author’s intent and creative freedom in story generation, controlling non-player character’s behavior, and tailor interactive narrative to fit individual player’s personalized preferences (Riedl & Bulitko, 2012). A broad range of computational methods, including adversarial search (Bates, 1992; Weyhrauch, 1997; Lamstein & Mateas, 2004; Nelson & Mateas, 2005; Garber-Barron & Si, 2013; Li et al., 2013), STRIPS-planning (Porteous et al., 2015; Robertson & Young, 2015), and machine learning-based techniques (Lee et al., 2014; Yu & Riedl, 2014; Nelson et al., 2006; Roberts et al., 2006; Harrison & Riedl, 2016; Rowe et al., 2014; Rowe & Lester, 2015; Wang, Rowe, Mott, & Lester, 2016, Wang et al., 2017a, Wang et al., 2017b) have been developed for interactive narrative generation.

Adversarial search algorithms usually encode narrative content into a plot graph (Figure 2.1), in which arcs denote precedence constraints and graph nodes represent narrative events so that one narrative event can only occur if all events constrained to occur prior to it have already occurred. A complete story is then formed as a plot trajectory, in which a smart
experience manager would be able to visit favorable nodes while avoiding unfavorable ones for individual players. Bates first proposed the idea of search-based drama management (SBDM) in (Bates, 1992), with Weyhrauch further developing this work by applying it on a simplified version of the Infocom interactive fiction *Deadline* (Weyhrauch, 1997). Lamstein and Mateas (2004) implemented the first full SBDM in a real-time story world setting and evaluated its performance with human players. Nelson and Mateas (2005) expanded the prior SBDM systems and enabled it to process hundreds of plot points using Boolean concatenated ordering constraints on plot points on interactive fiction *Anchorhead*. Garber-Barron and Si (2013) exploited graph-based searching technique in an automatic storytelling system, and adapted story contents according to periodically estimated user preference. They also made progress on consistency and novelty in automatically generated storytelling. Li et al. utilized an innovative crowdsourcing framework to generate plot graphs (Li et al., 2013), so that plot structure could be automatically abstracted from a corpus of narratives, and new stories could be formed by sampling in a domain model.

To depict interactive narratives, STRIPS-planning techniques define explicit initial states and goal states, with a set of actions being able to change narrative state by imposing certain effects (post-conditions) under pre-conditions. A story would be formed by following a plan, which consists of a sequence of operations, and leads to a goal state from the initial state. Young et al. (2004) proposed a general plan-based narrative behavior generation architecture *Mimesis* in framing conventional game engines. This run-time narrative planner is able to specify a sequence of in-game goals, as well as maintain the coherence of these plans in face of unpredictable user activities. Riedl and Stern (2006) presented an interactive narrative system focusing on both narrative control and believable autonomous agent’s
behavior generation. They used a tiered re-planning approach to restore causal coherence by removing threatened causal link events and filling in new events. Kartal et al. (2013) used plan-based narrative generator in large story domains in which the number of optional actions could exceed 100. Monte Carlo tree search was adopted for searching through large domains, and a Bayesian story evaluation method was utilized to guide planning toward believable stories. Porteous et al. (2015) proposed an automatic method to generate an extended interactive narrative domain under planning framework. They used opposite actions and predicates to those forming the baseline and enabled the automated creation of alternative narrative contents. User evaluation demonstrated that the generated contents are understandable to humans. Another narrative-centered game engine has also been developed to support planning in the content management pipeline, which functions by layering a procedural content generation pipeline on top of an experience management framework (Robertson & Young, 2015). The so-called General Mediation Engine system, can produce an interactive narrative by instantiating, controlling, and destroying assets based on narrative state.

In the machine learning family, several approaches have been utilized in interactive narrative generation and adaptation. Lee et al., (2014) proposed a supervised learning framework to devise an effective narrative director agent that enables dynamical story event sequence generation according to players’ actions and needs. Dynamic Bayesian network is utilized to learn a narrative adaptation strategy from human “wizards”. The experiment results indicate that machine learning driven narrative director agent can yield models to positively influence players’ learning and gameplay experiences, and dynamic Bayesian network outperforms other baseline machine learning models, including naïve Bayes and
bigram model, in devising effective narrative director agents. Recommender system-based drama manager has also been adopted in interactive narrative adaptation and personalization, in which a personalized interactive narrative should be composed of a sequence of plots rationally recommended to individual players. The Prefix-Based Collaborative Filtering (PBCF) algorithm, as a personalized plot recommendation method, was proposed to compose stories using the story library Choose-Your-Own-Adventure (Yu & Riedl, 2014). Specifically, the PBCF algorithm recommends those stories with potentially high rates to players in accordance with their rates to prior partial stories. The empirical results show the PBCF algorithm can improve players’ narrative experiences over the designer’s storytelling intention alone and can deliver more personalized interactive narratives to players. Reinforcement learning (RL)-based methods have been studied in interactive narrative generation and adaptation tasks for a long time. Reinforcement learning is a common choice for interactive narrative generation and adaptation, generally because it models and solves sequential decision-making problems within uncertain environments (Littman, 1996; Sutton & Barto, 1998; van Otterlo & Wiering, 2012), which properly aligns with plenty of interactive narrative generation tasks. Further, RL supports encoding narrative quality in terms of reward functions, which guide the process of training an interactive narrative planner by optimizing a reward function determining target. Utilization of reward function allows the narrative quality measurement to follow a sparse quantitative format, which properly fits the interactive narrative generation and adaptation problems when player outcomes like engagement or learning performance are adopted in narrative evaluation. Nelson and Roberts et al. utilized temporal-difference methods to train a drama manager for a text-based interactive fiction, Anchorhead, using simulated user data and an author-
specified story evaluation function (Nelson et al., 2006). The empirical results demonstrate that RL performs as well as adversarial search methods in simple interactive narrative domains and could outperform adversarial search in more complex domains. Furthermore, they proposed the self-adversarial / self-cooperative exploration training method, which is able to exceed normal TD training on interaction with simulated user model. Roberts et al. (2006) applied target-trajectory distribution MDPs (TTD-MDPs) to generate interactive narratives according to an author-specified target distribution over possible stories. Instead of applying normal training in the RL framework, they represented trajectories of MDP states as new form of state representation in the TTD-MDP model and used standard linear algebra to derive policies if there exists one. They also proposed solutions for conditions when constraints occurred to prevent direct policy calculation using linear algebra. Thue and Bulitko (2012) used MDPs to model player behavior in a player-adaptive interactive narrative, dynamically augmenting story events based on estimates of player gameplay preferences. To describing player’s behavior, a reward estimator and a policy estimator were derived from expert-defined functions using a set of game features. In this way, game content would be adaptively adjusted according to the estimation of player’s behaving tendency. Harrison and Riedl (2016) proposed Quixote, a system that programs virtual agents in interactive narrative environments. In the Quixote system, a reward function is learned from human generating exemplar stories to encode desired virtual agent behavior. Then reinforcement learning is adopted with the derived reward function to control the actions of virtual agents to make them rationally interact within the narrative environment. Empirical results in a virtual narrative environment Robbery World demonstrated the effectiveness of the system.
Reinforcement learning methods get an especially large amount of attention in educational interactive narrative adaptation, generally because the highly uncertain student-narrative planner interaction and the adoption of learning outcomes in narrative evaluation fit properly into the reinforcement learning framework. However, it is not intuitive to adopt RL in educational interactive narrative generation. For the purpose of limiting the RL policy search space, decomposition-based modular reinforcement learning approaches (Singh, 1992; Russell & Zimdars, 2003) have been studied. Decomposition-based modular reinforcement learning divides the origin RL problem into multiple sub-problems, with each sub-problem being dealt with one RL agent. If necessary, an arbitrator module would be adopted to make final decisions utilizing decisions from each sub-RL agent (Russell & Zimdars, 2003). Rowe et al. presented the first example of a modular RL framework for interactive narrative adaptation by introducing the concept of adaptable event sequences (Rowe & Lester, 2015; Rowe et al., 2014), a decomposition-focused abstraction for recurring story units within interactive narratives. They used 12 separate Markov decision processes (MDPs) to describe narrative interactions of 12 types, in each of which one adaptable event sequence is constructed. In the experiment using an educational narrative-centered open-world game—CRYSTAL ISLAND, it is shown that modular RL deriving interactive narrative planners induce more effective narrative adaptation policies than heuristic-based policies and random policy, measured by policy value calculated following expected cumulative reward. Wang and Rowe et al. (2016) then extended this framework by exploring alternative decomposition structures for a modular RL-based drama management; Rowe et al. investigated only one type of decompositional representation based on author-specified adaptable event sequences (Rowe et al., 2014). Also, Wang, Rowe, Mott and Lester (2016) utilized importance sampling-based
off-line evaluation methods (Peshkin & Shelton, 2002; Mandel et al., 2014) to assess drama management policies induced from player interaction data, enabling policy evaluation without the requirement to collect additional data from new groups of human players. Based on a corpus collecting from more than 400 students’ gameplay, it is found that decomposition structure in modular RL has significant effect on narrative planning policy’s quality. In case with limited human player generating data, the modular RL agent with the optimal decomposition structure would statistically significantly outperform singular RL and fully-decomposed RL agents’ policies and the uniform random policy, although it is not clear if an optimal decomposition structure can be efficiently found.

2.2 Simulated Player Model

Player interaction data is the major information resource in data-driven interactive narrative generation and adaptation. In interaction narratives which endow players with the authority to explore the narrative environment and lead the unfolding of narrative branches freely, player interaction data can be the most important source to learn player’s behavior. However, collecting player data sometimes can be particularly time-intensive and resource-intensive, especially when formal measurement of player’s gameplay outcomes (e.g., learning outcome, engagement, etc.) is necessary to obtain. Researchers have utilized several methods to collect or synthesize player interaction data for data-driven interactive narrative planner derivation. Heuristic-based player interaction data generation is one of the major narrative generation data sources. For temporal-difference method-based drama manager derivation work (Nelson et al, 2006), synthetic simulated player data is utilized. Specifically, they assumed that player would behave either cooperatively or adversarially with the drama manager. The strategy helps the designer understand the effectiveness of the drama manager; however, this heuristic
might be too simple to reflect human players’ gameplay preferences in complex interactive narratives. Roberts et al. (2006) adopted a heuristic about pre-defined narrative interaction trajectory distribution in their study of how the TTD-MDP model generates interactive narratives. Compared to the self-cooperative / self-adversarial assumption, the narrative trajectory distribution heuristic is more powerful in describing human players’ interaction patterns with commonly seen large variance in its distribution; however, it is hard to properly define this distribution accurately even by experts. Also, for certain problems like interactive narrative adaptation for individual players, the objective of the interactive narrative planner is not manipulating the distribution of the narrative trajectories. This fact makes the TTD-MDP model less useful in solving these problems. In interactive narrative application domains like education, it is also difficult to directly build up a linear state transition model to describe the narrative interaction dynamics precisely. This property prohibits the general employment of TTD-MDP in interactive narrative adaptation problems in educational interactive narratives.

In recent years, Crowdsourcing has been an innovative and effective way to gather data for interactive narrative generation research. Li et al. (2013) proposed to adopt crowdsourcing to learn a domain model for automatic story generation. Utilizing Amazon Mechanical Turk (AMT), the authors trained a plot graph, which models logical event flow in the virtual story world as a set of precedence constraints between plot events, from a dataset of stories generated by AMT users. Then, innovative stories can be generated by iteratively sampling and adding executable event from the space of narratives. The strategy of utilizing crowdsourced data in story generation was then extended by Harrison and Riedl (2016) in their system called Quixote. Using AMT, they first collected a corpus of exemplar stories, with which Quixote automatically learned a reward function for each virtual agent.
Based on the learned reward function, virtual agents’ behavior can be rationally controlled by a reinforcement learning model. In a case study, they found that virtual agents’ behavior generated by *Quixote* matched with the behaviors from the training corpus in the interactive story environment *Robbery World*.

Player modeling is another option to describe players’ behavior and generate synthetic data of player interactions in interactive narratives. Thue et al. (2007) presented the interactive storytelling system PaSSAGE (Player Specific Stories via Automatically Generated Events) to dynamically select the content of an interactive story for individual players. PaSSAGE learns a model of player’s preferred style of story using a player model from Robin Laws’ rule (Laws, 2002; Peinado & Gervás, 2004). Robin Laws’ rule categories players into five types: fighters who prefer combat; power gamers who prefer gaining special items and riches; tacticians who prefer creative thinking; storytellers who prefer complex plots; and method actors who prefer taking dramatic actions. Based on player’s observed actions, PaSSAGE estimates one player’s style by adjusting weights assigned to each optional player type, according to which personalized interactive story content would be demonstrated to each player. Yu and Riedl (2014) also adopted Robin Laws’ rule in their prefix based collaborative filtering based interactive story generator’s verification. Based on the rule, simulated players were built to interact with the story generator. Every simulated player has a five-dimensional characteristic vector, with each entry in the vector specifying the corresponding player characteristic with respect to the five player categories respectively. Correspondingly, story prefixes are rated according to beliefs about how they match with Robin Laws’ player scheme, so story branches would be suggested to players with the highest probability to enjoy the story pattern.
It is also common to see corpus collected from studies involving human players utilized in data-driven interactive narrative generation and adaptation. Compared to synthetic player data generated following heuristics, human player generating data can describe player’s interaction patterns in more complicated narrative environment; compared to crowdsourcing data collection, gathering data in human players involving studies usually is more controllable, which makes it specifically fit the requirement of interactive narrative generation and adaptation research in education and training domains, considering pre-test and post-test are usually necessary in these domains. Although advantageous, organizing human players involved studies is usually time-intensive and resource-intensive, which usually prohibits player interaction corpus to be large. Lee et al. (2014) utilized human player generating corpus directly in their dynamic Bayesian network-based narrative director agent training in educational games. Further, the Wizard-of-Oz mechanism was adopted, so a human expert narrative director agent demonstrated how a narrative should be unfolded to each individual player. The dynamics Bayesian network driving director agent imitates human expert’s action in a supervised learning setting, which effectively utilized human player data, and simplified the narrative generation problem from sequential decision making to one-step decision making; however, this method introduces the risk of learning non-optimal narrative director agent interaction strategies in interactive narratives, especially due to the fact that a proper strategy to get out from “bad” interaction states might be rarely demonstrated by experts. Human player generating data has also been adopted in reinforcement learning-based interactive narrative planner derivation (Rowe et al., 2014; Rowe & Lester, 2015; Wang, Rowe, Mott, & Lester, 2016). Forming the interactive narrative generation and adaptation problems as reinforcement learning problems allows the searching
for the optimal narrative interaction policies under the sequential decision making framework, even though it can be challenging to properly train a reinforcement learning agent with corpus of limited size, which is not uncommon in human players involving studies. Rowe et al. (2014; 2015) built up Markov decision processes (MDPs) from the human player generating corpus, and then did training and evaluation on the same MDPs using the metric of expected cumulative rewards (Tetreault & Litman, 2006), in which case the generalizability of the trained reinforcement learning-based narrative adaptation policy cannot be evaluated. Wang, Rowe, Mott and Lester (2016) applied cross-validation to verify the effectiveness of modular reinforcement learning-based interactive narrative planners using importance sampling with human player generating data. However, statistical techniques, like importance sampling and weighted importance sampling, usually inherit large variance (Mandel et al., 2014), which can be problematic in certain cases, especially when narrative trajectories are long.

While data-driven simulated players have not been broadly studied in the field of interactive narratives, some research about simulated students in intelligent tutoring systems might shed some light on player simulation in educational interactive narratives. VanLehn et al. (1994) outlined the three practical applications of simulated students: simulated students provide opportunities to teachers for refining instructional skills; simulated students can perform as co-learners; simulated students can help formally evaluate learning materials. To date, the focus of research on simulated students in the field of intelligent tutoring systems majorly has been in the cognitive aspect of student learning (Rowe et al., 2017). Predictions of students’ problem-solving behaviors and learning outcomes have become the main research direction (Beck et al., 2000; Matsuda et al., 2015; Rosenberg-Kima & Pardos,
2015). Out of these works, the ADVISOR intelligent tutoring system proposed by Beck et al. (2000) solves a similar problem to the one studied in this dissertation. The ADVISOR system employed two machine learning driving agents, with the first agent being responsible for learning a model of how students perform when use the tutoring system in various contexts, and the second agent utilizing reinforcement learning to train a tutoring policy to teach the tutee simulated by the first agent. The architecture of ADVISOR is designed for the purpose of centralizing the reasoning of an ITS into a single component to offer customized tutoring to tutees. Experiments conducted with human students demonstrated the effectiveness of the ADVISOR architecture in deriving successful teaching policies.

Figure 2.2: Architecture of a spoken dialogue system with simulated users derived using machine learning approaches (Schatzmann et al., 2006).

Although data-driven simulated player models have not been thoroughly studied and utilized in fields of interactive narratives, it has been one of the major research interests in some other fields, for example spoken dialogue systems, for a long time (Schatzmann et al., 2006; Young et al., 2013). As claimed by Schatzmann et al. (2006), although the training of machine learning-based dialogue management strategies could in theory be done using either human users or corpora of human-computer dialogue, in practice the typically vast possible
dialogue space of states and policies cannot be thoroughly explored without the usage of automatic simulated users. Although traditionally, non-statistical user modeling techniques was the major research direction of user modeling in spoken dialogue systems, these techniques demonstrated their disadvantages of being highly complex to design in nature (Zukerman & Litman, 2001). On the other hand, user simulation systems using machine learning approaches to construct statistical user models have gained increasing attention in recent decades (Levin et al., 2000; Young, 2002). Machine learning driving simulated users are often utilized to imitate human users behavior, and they have been adopted effectively in training and evaluating dialogue strategies (policies) through trial-and-error interactions. Usually, utilization of statistical simulated users in spoken dialogue systems is of two phases (Figure 2.2). First, a simulated user model is trained using supervised learning approaches upon a dialogue corpus collected from human users. Second, a dialogue manager strategy would be learned with reinforcement learning methods in interactions between the dialogue manager and simulated users, when simulated users would behave by predicting and imitating human users’ preferences learned in the prior phase.

Several machine learning methods have been studied in statistical predictive user simulation modeling in spoken dialogue systems. Eckert and Levin introduced n-gram models to predict user actions given the dialogue history of the dialogue manager and user actions (1997; 1998). The n-gram models can be fully stochastic, and totally domain-independent in any spoken dialogue systems, although careful extra design may be necessary if logical constraints exist to prohibit some actions to be taken by users under certain conditions. Scheffler (2003) adopted graph-based models in simulated user modeling. The method maintains deterministic rules for user’s goal-dependent actions in specific dialogue
systems, as well as the probabilistic modeling convering the conversational behavior distribution learned from human user corpus. Bayesian networks were later utilized to model users’ interaction in dialogue systems (Pietquin & Dutoit, 2005; Pietquin & Beaufort, 2005). Compared to prior methods, Pietquin et al. proposed to train a Bayesian network model to learn user action distribution based on requested information, user’s goal and memory of dialogue interactions. They acknowledged the importance of understanding user state and its effect on dialogue behavior to accurately predict user’s actions. Georgila et al. (Georgila, Henderson & Lemon, 2005; Georgila, Lemon, & Henderson, 2005) utilized supervised learning techniques to predict user actions in dialogues. Specifically, user action prediction is formed as a classification problem, in which the probability of the user taking each action is estimated given a real-valued feature vector representing the dialogue interaction state between the dialogue manager and the user. Hidden Markov models have also been studied to simulated user-dialogue manager interactions, in which an Input-Output-HMM is employed (Cuayáhuitl et al., 2005). A set of state transition probabilities and a set of output probabilities are maintained to record the interaction, and then user response is predicted from user model probabilities.

2.3 Deep Learning

Deep learning techniques have gained significant interest in the recent decade in both academia and the artificial intelligence industry. This neural network based machine learning technique, as mentioned in Section 1, has improved the performance of the state-of-the-art solutions in multiple domains. In this subsection, the recent advancement and success of the deep learning techniques, and some successful applications of the deep learning methods in solving similar problems to the interactive narrative-centered problem are briefly discussed.
Although learnable neural network related research can be dated back to 1950s (Rosenblatt, 1958), the most recent popularity of deep neural network models started from around 2006. Even though neural network models, as universal function approximators (Csáji, 2001), have astonishing capability to fit very complex functions, training of deep neural networks is usually not intuitive. Hinton et al. (2006) proposed a layer-wised pretraining method, which can be utilized, as an unsupervised method, to initialize the network parameters in a deep belief network. This work represents the start of quick growth in research to design, train and apply deep neural network models in this recent deep learning upsurge.

In 2012, Krizhevsky et al. (2012) developed an 8-layer convolutional neural network model to compete in the Large Scale Visual Recognition Challenge (LSVRC) using the dataset Imagenet (Deng et al., 2009). After training on the 1.2 million high-resolution images from 1,000 classes, the proposed deep convolutional neural network (AlexNet) achieved a top-5 test error of 15.3%, which was much better than the prior state-of-the-art of 25.8%. Compared to prior methods (Perronnin et al., 2010), in which expert-designed image feature extractors (e.g., Fisher Vectors (Perronnin et al., 2010)) were utilized together with shallow machine learning models (e.g., support vector machine), the AlexNet does not use any human engineered visual features. Instead, the deep convolutional structure learns useful and representative visual features all by itself. AlexNet follows the way how convolutional network is adopted in prior image recognition systems like LeNet-5 (Lecun et al., 1998), however it constructs a much larger convolutional network in all dimensions. The stacked convolutional layers improve the model’s capability of extracting more abstract and more representative hierarchical features; the adoption of the non-linearity ReLU (Nair & Hinton,
2010) mitigates the gradient vanishing problem in training very deep networks; the utilization of the dropout regularization technique (Hinton et al., 2012; Srivastava et al., 2014) cleverly improves the model’s generalizability; and the usage of GPUs allows an obvious improvement in deep network training speed, which makes training and utilization of large and deep neural networks to be feasible. The AlexNet paper significantly affects the computer vision community, so that deep convolutional networks have almost become a standard choice for plenty of the modern computer vision tasks, including image recognition, object detection, localization and image segmentation. Following this direction of utilizing deep convolutional networks (CNNs) in computer vision systems, researchers have extended the scale of the AlexNet model further, so deeper and larger CNNs with various improvements were proposed in the later years and the state-of-the-art were boosted in a fairly quick speed. Taking the LSVRC as an example, we can briefly see what the research trend was in deep learning in the recent decade, and how it improved systems’ capability in the field. In 2014, two competitors into the LSVRC achieved dramatic performance and attracted attentions. GoogLeNet was proposed as a 22-layer deep CNN employing an inception module (Szegedy et al., 2014). Compared to AlexNet, it is a much deeper CNN model, which dramatically increases the ability of learning complex features. With the help of the inception modules, the scale of trainable parameters is decreased from about 60 million (AlexNet) to 4 million (GoogLeNet), that enables the quick training and easier deployment of GoogLeNet with a much more improved top-5 validation error of 6.7%. VGGNet was another successful competitor in that year, which employs a 19 weight layers CNN-based structure (Simonyan & Zisserman, 2015). Compared to prior models (e.g., AlexNet), VGGNet is constructed with a larger depth and smaller (3×3) convolution filters.
VGGNet demonstrated outstanding performance on the Imagenet dataset, with a top-5 evaluation error rate of 6.8%, and good generalizability on other datasets. In 2015, the novel deep residual network (ResNet) was proposed and employed in the LSVRC competition (He et al., 2016). ResNet innovatively uses skip connection in building very deep CNNs, so convolutional layers in ResNet does not just learn a hierarchical feature presentation, instead it learns a residual representation with respect to features in prior layers which have skip connections into the current layer. The usage of skip connection significantly increases the capability of training even deeper CNNs, so a 152-layer ResNet achieved a top-5 evaluation error rate of 3.57%, which surpasses human expert’s performance (Russakovsky et al., 2015) on recognizing image categories in the Imagenet dataset.

Although deep learning techniques have not been broadly studied in simulated player modeling and interactive narrative generation and adaptation problems, the success of deep learning in problems with sequentially formatted data, which is one important property shared by the problems studied in this dissertation and many other fields, has motivated our ambition to study the effectiveness of deep learning techniques in solving our specific interactive narrative-centered problems. Speech recognition problems usually takes advantage of the sequential patterns of speech signals in translating spoken dialogues into texts. Amodei et al. (2016) utilized the combination of recurrent neural networks (RNNs) and convolutional neural networks (CNNs) in designing an end-to-end speech recognition system for both languages of English and Mandarin. The end-to-end system architecture allows the abandonment of human expert’s knowledge in designing hand-engineered components in the speech recognition pipeline. The utilization of high performance computing (HPC) infrastructure shortened the training time of such large and deep NN models from weeks to
several days. As a result, the so-called deep speech 2 system achieved competitive results to human workers’ transcription quality. Wang and Skerry-Ryan et al. (2017) applied complex deep learning techniques in end-to-end text-to-speech synthesis work. Similar to the previously introduced deep speech 2 system, the so-named Tacotron system uses <text, audio> pairs directly in a speech generation model training with no need to any human expert’s extra knowledge into the system. Tacotron also utilizes both CNN modules and RNN modules. Specifically, attention RNN (Xu et al., 2015; Mnih et al., 2014) is employed in dealing with the sequential data extracted from each frame. Experiment results demonstrated that Tacotron outperforms a production parametric system in terms of naturalness. Kumar et al. (2016) applied deep learning into another field with commonly seen sequential data: natural language processing. They proposed the dynamic memory network, which also adopts RNNs and attention mechanism (Xu et al., 2015; Mnih et al., 2014) in a question answering system. Specifically, the applied attention mechanism permits the model to condition its attention on specific inputs and the result from the previous related question answering iterations. This end-to-end model obtained state-of-the-art performance on several types of tasks and databases, including question answering, text classification for sentiment analysis, and sequence modeling for part-of-speech tagging. Machine translation is another domain in which deep learning techniques assist in solving the complex language translation problem with sequentially formatted data. Wu et al. (2016) implemented the Google’s Neural Machine Translation (GNMT) model, which consists of a deep LSTM network with 8 encoder layers and 8 decoder layers. Residual connection and attention connections are built from the decoder network to the encoder. To improve handling of rare words, they also divided words into a limited set of common sub-words units for both input and output.
GNMT achieved competitive results to the state-of-the-art at that time on English-to-French and English-to-German benchmarks.

Adoption of deep learning techniques on narrative-centered educational games has also shown great promise. Min et al. (2014) proposed to use an unsupervised pre-training technique—stacked denoising autoencoder (Vincent et al., 2008), to assist in training multiple layer fully connected neural networks for predicting human player’s goals. The authors claimed that with no expert knowledge built into the goal recognition system, the stacked denoising autoencoder based goal recognition system could outperform the prior state-of-the-art, which was a Markov logic network-based model (Ha et al., 2011), in all three metrics of accuracy rate, prediction convergence rate and prediction convergence point. This work was then extended by the same group of authors, who experimented with long short-term memory (LSTM) models on the same task (Min et al., 2016). The sequences of player interaction data were directly used as input into an LSTM network, which properly encode player interaction history using gated recurrent LSTM unit with shared parameters along time axis. Besides, a player action embedding representation method was introduced, which utilizes a similar definition of word embedding from the natural language domain (Bengio et al., 2003). The experiment results indicate that, by utilizing the LSTM model, the player goal recognizer can obtain statistically significant improvement on player goal recognition accuracy, compared to prior models including Markov logic networks and feedforward neural networks pre-trained with stacked denoising autoencoders. These two papers specifically demonstrate the impressive potential of deep learning techniques in problems within an interactive narrative-centered domain. Human player’s interaction with the
narrative planner, although potentially with large variance in the distribution, could be properly handled with deep learning techniques.

2.4 Reinforcement Learning

As a framework to solve sequential decision making problems, reinforcement learning has been broadly studied and successfully exploited in several domains in recent years. The considerably increasing popularity and advancement of the deep learning techniques significantly boosts the development of reinforcement learning research, so even human-level or super-human performance has been achieved by reinforcement learning methods in several fields (e.g., playing video games, playing go game, controlling humanoid robot et al.). In this subsection, we’ll briefly cover some recent success and advancement of reinforcement learning, especially deep reinforcement learning approaches. Because the prior utilization of reinforcement learners in the task of interactive narrative generation and adaptation has been discussed in Section 2.1, these applications will not be covered in this subsection anymore.

Before the most recent thrive of deep learning, reinforcement learning has attained some success in different fields. Besides the usually used simple testbeds for reinforcement learning algorithms, for example grid-world navigation, cart-pole balancing, inverted pendulum and mountain car, reinforcement learning has been applied in solving some real-world problems. Kohl and Stone (2004) proposed to use policy gradient reinforcement learning to control the locomotion of quadrupedal robots to implement fast walk. After training for about three hours, the locomotion policy induced by the policy gradient method can drive the robot to walk faster than any previous known gait for the robot. Ng et al. (2006) used reinforcement learning to fly an autonomous helicopter. Autonomous helicopter flight
is regarded as a hard problem because of the highly stochastic and nonlinear dynamics in the flight environment. The authors utilized supervised learning to identify a model of the helicopter’s dynamics, with which reinforcement learning is adopted in training a policy which learns to control the helicopter’s locomotion, including difficult inverted hovering. Reinforcement learning also have demonstrated its capability in learning strategies to play some board games. Temporal difference learning has been adopted successfully to play games like backgammon (Tesauro, 1995) and checker (Schaeffer et al., 2001). Abe et al. (2004) investigated the usage of reinforcement learning in marketing. In their work, they described a novel solution that treats problems of cross channel integration and customer life time value modeling in one unified MDP framework. And the result demonstrated a 7% to 8% increase in the store profits by exploiting a mailing policy induced by reinforcement learning.

In recent years, deep reinforcement learning, which utilizes deep neural networks as function approximators in the reinforcement learning framework, has attained great attention and achieved human-level or even super-human performance in several problems which were treated as very challenging problems. The great power of deep neural networks significantly enhances the capability of reinforcement learning model’s capability in extracting meaningful representations, which allows the reinforcement learning models to learn to interact with the environment using high-dimensional raw features (e.g., pixel values in Atari games). However, at the same time, utilization of deep learning also introduces challenges in training the reinforcement learning agent in a stable way to obtain effective policies in reasonable training time.
Video games, for example Atari 2600, have become one of the standard testbeds for reinforcement learning research in recent years (Bellemare et al., 2013). Video games offer a cheap, repeatable and challenging task for reinforcement learning agents. It is cheap and repeatable because video game testbeds, like the Arcade Learning Environment (ALE), can generate a large amount of training trajectories with little expense. The problem of designing video game play agent is also challenging, because some games have complex design for emitting rewards, and state feature can be in very high dimension (210×160 color video at 60Hz frame rate) if raw pixel values were used as input. Mnih et al. (2015) proposed the deep Q-network (DQN) technique, which is the earliest success of utilizing deep reinforcement learning techniques in solving Atari 2600 gameplay problems with human-level performance. They employed a CNN module to extract effective hierarchical features from the raw pixel value inputs. And the output of the neural network is the Q-values for each discrete game actions matching to controls that players can make from the game controller. Q-learning with the same set of configurations is adopted to learn a gameplay policy for each of the 49 Atari games. In this work, they pointed out that the major problem of utilizing deep neural networks in reinforcement learning is the stabilization of the training process, and they innovatively proposed to utilize two techniques—experience replay and fixed target network to stabilize training. In the experiment, they compare the performance of the DQN policies against professional human game testers. The result demonstrates that for 75% of the 49 games, the DQN-based policies achieved performance that is comparable or superior to human players. This work was an astonishing achievement at that time, and significantly motivated the research of deep reinforcement learning. Then, several following papers appeared and succeed in improving the DQN method to make the training process faster and
the trained policy better. Van Hasselt et al. (2015) extended the DQN method with the Double Deep Q-networks (DDQN) approach, in which different Q functions for action selection at next time step and action value estimation at next step are utilized. Compared to the DQN method, DDQN mitigates the problem of overoptimistic value estimation in Q-learning. Schaul et al. (2015) also extended the DQN method by introducing prioritized experience replay, which is a variance of the experience replay technique utilized in the DQN work. Compared to normal experience replay, in which every state transition tuple stored in the database will be uniformly sampled in the learning process, prioritized experience replay set a weight to every state transition tuple according to estimated learning progress (an estimation of the TD-error used in Q-learning), so that each tuple would be sampled following the probability parameterized by the weights. The experiment result of prioritized experience replay turned out to speed up the training, as well as generate a better policy. Wang, Schaul, Hessel, Van Hasselt, Lanctot and De Freitas (2016) further improved the DQN method and proposed Dueling DQN, which applies advantage learning using Q-networks. With a slight change on the structure of Q-network, the dueling Q-network estimates state values and action advantage values separately. This specific network structure allows the general state value estimation across actions. Empirical results demonstrate that the dueling network generates better policies for playing Atari 2600 games than DQN does. Mnih et al. (2016) proposed another idea of stabilizing the training process of deep Q-networks. Instead of using experience replay, which usually requires large memory space to record past state transition tuples, asynchronous architecture is proposed to train Q-networks. With the asynchronous setting, multiple threads would be initialized and adopted in parallel to train the Q-network, which means that multiple RL agents could train the Q-network
together using shared Q-network parameters. This proposed Q-network implementation architecture is able to significantly speed up the training of Q-networks, as well as generate policies with comparable quality.

Simulated robot control environment (e.g., MuJoCo) is another commonly utilized deep reinforcement learning testbed. Different from video game problems in which actions for the RL agents are usually discrete, in simulated robot control problems, RL agent’s actions are usually continuous, which makes the original deep Q-network from Mnih et al. (2015) not intuitively compatible. To solve this problem, several policy gradient or actor-critic methods based approaches have been proposed. Asynchronous advantage actor-critic method (A3C) was proposed by Mnih et al. (2016). Experiment results indicate that A3C can generate effective policies to do continuous control on the testbed of MuJoCo. Trust region policy optimization (TRPO) method was introduce by Schulman et al. (2015), which is a practical implementation of the theoretically effective natural policy gradient optimization method. TRPO constrains the changes of the RL policy in each iteration, so the new policy would not be significantly different from old policy, which helps to derive good policies more efficiently. Schulman et al. (2017) then introduced proximal policy optimization (PPO), which approximates trust region policy optimization. PPO modifies the form the target function used in actor-critic RL, so that a threshold controlling how much the policy can change in one iteration of optimization is introduced. PPO offers a simple implementation of TRPO only using the first-order derivatives from the neural network function approximator. Wang, Bapst, Heess, Mnih, Munos, Kavukcuoglu and De Freitas (2017) introduced the method of actor-critic with experience replay (ACER). The innovative techniques proposed in this work, including the truncated importance sampling with bias correction, stochastic
dueling network architecture, and an efficient implementation of TRPO by maintaining an average policy network to represent a running average of past policies which is used to constrain policy update, have enabled ACER to perform well on both Atari gameplay and continuous control problems. Wu et al. (2017) introduced another trust-region method for deep reinforcement learning, which is named as actor-critic using Kronecker-Factored Trust Region (ACKTR). The Kronecker-factored curvature approximation method is used to do natural policy gradient optimization in neural network based actor-critic reinforcement learning. The empirical results also show the effectiveness of this method in both Atari gameplay and MuJoCo continuous robot control.

Besides the above achievements, deep reinforcement learning has also been adopted in playing very challenging board games. Silver et al. (2016) proposed AlphaGo, which is a deep reinforcement learning framework heavily utilizing Monte Carlo tree search technique. AlphaGo learns from expert human players first, and then learn to play against itself to generate good game play strategies partially utilizing the knowledge learned from human player. In 2016, AlphaGo beat the 18-time go game world champion Lee Sedol in a five-game challenge, which demonstrated the great power of the AlphaGo system that was never achieved before. Silver et al. (2017) then extended the work by introducing AlphaGo Zero, which no longer learns from human experts at all. They use a neural network to predict AlphaGo’s own move selection and also the winner of the game. Only using self-play, AlphaGo Zero achieved superhuman performance and won 100-0 against the previous version AlphaGo.
CHAPTER 3

CRYSTAL ISLAND INTERACTIVE NARRATIVE

This section presents the testbed utilized in this data-driven interactive narrative adaptation research, which is an interactive narrative-centered open-world educational game—CRYSTAL ISLAND. The following aspects of the CRYSTAL ISLAND will be introduced. First, the backstory and the open-world game environment of the CRYSTAL ISLAND interactive narrative will be described. Second, we will present the selected features and player actions in player interaction representation used in the interactive narrative adaptation framework. Then, adaptable narrative events designed in the CRYSTAL ISLAND interactive narrative will be introduced. Finally, the corpus investigated in the research will be described.

3.1 CRYSTAL ISLAND Backstory and Game Environment

CRYSTAL ISLAND is an interactive narrative-centered open-world educational game. It offers a rich, large virtual 3D environment that features eighth grade microbiology course content following the North Carolina curriculum standard (Figure 3.1). With one major subject of

![Figure 3.1: CRYSTAL ISLAND interactive narrative.](image-url)
extensive empirical investigation, CRYSTAL ISLAND has been found to provide substantial learning and motivational benefits to students (Rowe et al., 2011). The backstory of the CRYSTAL ISLAND interactive narrative begins with a group of scientists who work on a remote island and are bothered by an infectious disease outbreaking on the island. The player of CRYSTAL ISLAND adopts the role of a visiting investigator whose mission is to save the group of scientists by identifying the infectious source and proposing proper treatment. In the broad virtual environment, the player could explore by visiting the infirmary, living quarters, dining hall and a virtual laboratory. Non-player characters (NPCs) with identities of camp nurse, research scientists, camp cook and custodian, microbiology experts, and lab technician can communicate with the player to reveal story background, their physical conditions, microbiology knowledge and other story-related hints. Virtual books and posters covering relevant microbiology concepts offer players resources to learn the knowledge that is useful to solve the mystery. A virtual chemical lab supplies equipment to test contaminated food. In-game quizzes will come up when certain conditions are satisfied to assist players to remember and understand the knowledge deeply. To solve the mystery in the game, players need correctly finish a worksheet on which the source and treatment of the infectious disease need to be precisely identified.
3.2 CRYSTAL ISLAND Player Interaction Representation

Player’s interaction in the CRYSTAL ISLAND game environment are recorded with information in three categories: players’ gameplay action sequences, players’ traits, and players’ interaction history with the narrative planner. In this interactive narrative adaptation research, we concentrate on 15 player actions, which can basically describe the way how a player explores the game environment, unfolds the narrative during gameplay and conducts learning related activities. These actions are concretely illustrated in Table 3.1.

The actions listed here contain all major actions that can describe the amount of narrative that has been unfolded, as well as the amount of curriculum related contents that have been revealed. More importantly, using this set of player actions, narrative adaptation triggering opportunities can be deterministically determined. This means in the simulated player model, one set of rules for defining the narrative adaptation opportunities can be easily defined, so that we can clearly and accurately depict how players and interactive narrative planner interact in the simulation mode. Table 3.1 also shows the player action frequency is
listed. This frequency statistic is counted from the corpus collected in human players involved studies, which will be introduced in later subsection.

<table>
<thead>
<tr>
<th>Goals</th>
<th>Note</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Poster-Read</td>
<td>Reading time longer than 2 seconds</td>
<td>0.2218</td>
</tr>
<tr>
<td>Book-Read</td>
<td>Reading time longer than 5 seconds</td>
<td>0.0710</td>
</tr>
<tr>
<td>Talk-to-Kim</td>
<td>Dialogue branch into “pathogen branch”</td>
<td>0.0221</td>
</tr>
<tr>
<td>Ask-Teresa-Symptom</td>
<td></td>
<td>0.0342</td>
</tr>
<tr>
<td>Ask-Bryce-Symptom</td>
<td></td>
<td>0.0400</td>
</tr>
<tr>
<td>Talk-to-Quentin</td>
<td></td>
<td>0.0622</td>
</tr>
<tr>
<td>Talk-to-Robert</td>
<td></td>
<td>0.0364</td>
</tr>
<tr>
<td>Talk-to-Ford</td>
<td></td>
<td>0.0333</td>
</tr>
<tr>
<td>Take-Quiz</td>
<td></td>
<td>0.1311</td>
</tr>
<tr>
<td>Test-Object-in-Lab</td>
<td></td>
<td>0.2401</td>
</tr>
<tr>
<td>Submit-Worksheet</td>
<td>Incorrect submission</td>
<td>0.0489</td>
</tr>
<tr>
<td>Bryce-Reveal-Branch-1</td>
<td>Whether player triggers the action</td>
<td>0.0176</td>
</tr>
<tr>
<td>Quentin-Reveal-Branch-1</td>
<td>Whether player triggers the action</td>
<td>0.0110</td>
</tr>
<tr>
<td>Kim-Let-Quentin-Reveal-1</td>
<td>Whether player triggers the action</td>
<td>0.0059</td>
</tr>
<tr>
<td>Game-End</td>
<td></td>
<td>0.0245</td>
</tr>
</tbody>
</table>

The actions listed here contain all major actions that can describe the amount of narrative that has been unfolded, as well as the amount of curriculum related contents that have been revealed. More importantly, using this set of player actions, narrative adaptation triggering opportunities can be deterministically determined. This means in the simulated player model, one set of rules for defining the narrative adaptation opportunities can be easily defined, so that we can clearly and accurately depict how players and interactive narrative planner interact in the simulation mode. Table 3.1 also shows the player action frequency is
listed. This frequency statistic is counted from the corpus collected in human players involved studies, which will be introduced in later subsection.

With this set of player actions, we build the player interaction feature set, which can be used to trace how a player plays the game and how the interactive narrative planner tailors the narrative for each individual player. This feature set contains 21 features, fourteen out of which are from the player action information as listed in table 3.1. From the 1st player action to the 11th player action listed in Table 3.1, we use player action counts as feature values; for the 12th to the 14th actions in Table 3.1, a binary indicator is utilized to denote whether the player has triggered that event to happen at least one time in the gameplay. Game ending player action is not counted in the feature set.

Three player trait features are adopted. They indicate the player’s gender (binary value), player’s gaming frequency (integer value, 1-5), and player’s pre-test score (integer value, 1-19). There are other 4 features storing interactive narrative planners’ decision history, which have real values ranging in [0,1]. The meaning of these four features will also be described in the following subsection.

Standardization preprocessing of these 21 features is adopted in both simulated player modeling and adaptable interactive narrative planner derivation. The standardization process rescales the values of all features, so each feature should have zero mean and unit variance after preprocessing.

Because of the educational objective of CRYSTAL ISLAND, we utilize normalized learning gain (Marx and Cummings, 2007) to assess the quality of players’ interactive narrative experiences. Players typically complete pre-tests and post-tests when participating in classroom studies with CRYSTAL ISLAND. Normalized learning gain (NLG) is the
normalized difference between player’s post-test score and pre-test score, which is demonstrated in Equation 3.1.

\[ NLG = \frac{score_{post} - score_{pre}}{score_{max} - score_{pre}} \] (3.1)

In our analysis, we group player experiences into two categories: high NLG and low NLG. Players with NLG scores above or equal to the median value are in the high NLG group, and players with NLG scores below the median are labeled with low NLG. Although NLG is adopted in this study, other experience metrics (e.g., engagement questionnaire scores) could also fit into this interactive narrative adaptation framework.

### 3.3 Narrative Adaptation in **CRYSTAL ISLAND**

In the **CRYSTAL ISLAND** interactive narrative, several places are adaptable. It means NPC’s dialogue contents, feedbacks to player’s certain behaviors, and extra in-game functionalities can be dynamically controlled and adjusted to individual players. In this dissertation, we study four adaptable events. These four adaptable events can be triggered by certain player actions, and all affect the unveiling of curriculum content to players in the narrative:

- **Teresa Symptoms Event** is triggered each time the player inquiries about the sick scientist Teresa’s symptoms. The interactive narrative planner selects one of three conversational responses for Teresa: providing minimal detail, moderate detail, or maximal detail about her symptoms. In player interaction feature set, these three actions are encoded as real numbers, with 0 representing the event not being triggered yet; 0.33 representing providing minimal detail; 0.67 representing providing moderate detail; 1 representing providing maximal detail. The interactive narrative planner’s most recent decision is stored in the player interaction feature set (described in Section 3.2).
• **Bryce Symptoms Event** is triggered each time the player inquires about the sick NPC Bryce’s symptoms. The interactive narrative planner has two optional responses: providing minimal detail or moderate detail about his symptoms. In player interaction feature set, these two actions are encoded as real numbers, with 0 representing the event not being triggered yet; 0.5 representing providing minimal detail; 1 representing providing maximal detail. The interactive narrative planner’s most recent decision is stored in the player interaction feature set (described in Section 3.2).

• **Diagnosis Feedback Event** is triggered when a player submits an incorrect diagnosis worksheet. The interactive narrative planner selects one of three options for the camp nurse’s feedback: providing minimally detailed, moderately detailed, or maximally detailed feedback. In player interaction feature set, these three actions are encoded as real numbers, with 0 representing the event not being triggered yet; 0.33 representing providing minimal detail; 0.67 representing providing moderate detail; 1 representing providing maximal detail. The interactive narrative planner’s most recent decision is stored in the player interaction feature set (described in Section 3.2).

• **Knowledge Quiz Event** is triggered when a player converses with a subset of the NPCs. The interactive narrative planner optionally delivers, or does not deliver, an in-game quiz about relevant microbiology concepts. In player interaction feature set, these two actions are encoded as real numbers, with 0 representing no quiz delivered; 1 representing one set of quizzes delivered. The interactive narrative planner’s most recent decision is stored in the player interaction feature set (described in Section 3.2).
The appearance of these four adaptable events follows a clear rule in the game. For the Teresa symptoms event and Bryce symptoms event, they are triggered when the player adopts the Ask-Teresa-Symptom action or Ask-Bryce-Symptom action respectively. The diagnosis feedback event will be triggered when the submit-worksheet player action is taken. Triggering rule of the knowledge quiz event is a little more complex. When the player talks with NPC Quentin, or Ford, or Robert for the first time, the knowledge quiz event will be triggered to determine if the player will or won’t receive a knowledge quiz. All these rules are clearly and strictly followed in the simulated player modeling design. The purpose of adapting interactive narratives in CRYSTAL ISLAND is to improve players’ interaction outcomes and/or experiences. Specifically, because of the educational nature of CRYSTAL ISLAND, in this study we utilize normalized learning gain as a metric to evaluate the narrative adaptation effect in the game. Normalized learning gain has been presented in Section 3.2.

3.4 CRYSTAL ISLAND Interactive Narrative Corpus

In the research of interactive narrative adaptation, the human player interaction corpus for CRYSTAL ISLAND was generated from two human subjects involved studies (one with 300 students and one with 153 students), which were conducted in two public middle schools. Students played the game until they solved the mystery, or 55 minutes had elapsed, whichever occurred first. During the studies, a uniform random narrative planning policy was deployed to interact with human subjects. All adaptable events in CRYSTAL ISLAND were designed to ensure the coherence of the narrative when the random policy was adopted. The game logged all player actions, triggered adaptable events, and random interactive narrative planner responses. In addition, several questionnaires were administered prior to, and immediately after, students’ interactions with CRYSTAL ISLAND. These questionnaires were
used to obtain information about students’ individual traits, curricular knowledge, and engagement with the narrative environment. From all these information, we extract the data of player narrative interaction in CRYSTAL ISLAND following the interactive feature set described in prior subsections. Besides, all students’ scores of their pre-test exams and post-test exams were collected for calculating students’ learning outcomes. Specifically, in this educational game, the learning outcome serves well as a measurement of the player’s achievement for their gameplay, which is also the metric we utilize in our interactive narrative planner derivation for its performance optimization.
CHAPTER 4

STATISTICAL PREDICTIVE SIMULATED PLAYER MODEL

This section presents a statistical predictive simulated player model that was created for the interactive narrative adaptation research. The following aspects of simulated player modeling will be described. First, we introduce the general framework of the statistical simulated player model. Second, we will introduce neural network models specifically designed to model sequence data, which is aligned with the temporal player interaction data in interactive narratives. Third, we describe long short-term memory networks (LSTMs), a family of recurrent neural networks that effectively address the vanishing gradient problem. Fourth, we describe methods to construct deep recurrent neural networks, including stacked deep LSTM, Grid LSTMs, and deep recurrent highway networks. Fifth, deep convolutional neural networks in simulated player modeling is introduced. Finally, we describe several regularization techniques for deep neural network models that we utilize in simulated player modeling.

4.1 Simulated Player Modeling in CRYSTAL ISLAND

In interactive narratives with open-world virtual environments such as CRYSTAL ISLAND, players actively drive the narrative forward by performing actions in the virtual world. Usually, open-world environments afford many possible narrative trajectories—players may follow different sequences of conversing with NPCs, completing in-game sub-tasks, or interacting with virtual objects—and each player is likely to experience his or her own unique trajectory, albeit with similarities to peers’ experiences.

To simulate player behavior in CRYSTAL ISLAND, we devise a model that predicts the next player action distribution at each player interaction time step using prior gameplay
history, player trait data and narrative planner’s prior decisions as input. In addition, the player simulation model predicts the player’s experiential outcome distribution at the conclusion of the narrative episode. As in the CRYSTAL ISLAND interactive narrative, in which the open-world interaction environment affords a large space for players to freely explore the game environment and spread out the narrative without much constraints, usually players’ interaction patterns could diverse dramatically. One example in CRYSTAL ISLAND is that, some players might be interested in reading some virtual books and posters with curriculum content to gain curriculum knowledge before start to thoroughly explores the environment and find evidences of the disease breakout, while other players might prefer to collect suspicious items and conduct laboratory test to identify the disease source in the early stage of gameplay. Further, because learning outcome (measured by normalized learning gain) is the adopted metric for interactive narrative quality evaluation in the CRYSTAL ISLAND study, it’s almost impossible to find clearly deterministic rule even for experts to tell what type of player interaction patterns could result in good learning outcome. So, in this research, the statistical predictive simulated player model is employed. For both the player actions and player outcome, the goal of simulated player modeling is to derive a function to approximate the distribution of players’ behavior at each interaction time step, and the distribution of players’ normalized learning gain after the gameplay interaction ends.

Simulated player model is derived for imitating human players’ gameplay process. Specifically, with simulated players, data-driven interactive narrative planners can be trained and evaluated in both off-line and on-line modes conveniently. In this work, simulated player model is utilized in the following way to imitate human player’s gameplay: an initial simulation state is generated by sampling from the human player initial states’ probability
distribution (induced from human players’ corpus). The sampled initial player state should only indicate what type of player she is, by setting player trait feature values, and zeroing out all player action feature values and narrative planner past decision feature values. Next, the simulated player model is used to determine the distribution over likely player actions at the next player interaction time step. One action is sampled according to this distribution, and the player simulation’s current state is updated according to the effects of the synthetic player action. If the player action triggers an adaptable event, then a narrative adaptation decision by the narrative planner occurs, once again updating the simulation’s current state. This process continues until a game-ending player action is generated. Afterward, the simulated player’s experiential outcome is predicted.

We frame both player action prediction and outcome prediction as classification problems. Because player actions in CRYSTAL ISLAND can be represented in terms of a discrete player action set, player action prediction can be formalized as a multi-class classification problem. In CRYSTAL ISLAND, following the introduction in Section 3.2, we concentrate on 15 types of player actions (including the game-ending action), which collectively capture the different ways players explore CRYSTAL ISLAND’s interactive narrative. Also, as described in Section 3.2, because the interactive narrative testbed was designed for educational purposes, we focus on two types of player outcomes: high learning outcomes (represented with high normalized learning gain) and low learning outcomes (represented with low normalized learning gain). A binary divide on players’ learning outcomes forms the prediction into a binary classification problem. This design is adopted for concerns from several aspects. First, predicting players’ learning states using gameplay features, simple player trait features, and narrative planning decision features in an
educational interactive narrative is a very difficult problem, especially considering the human subjects generating corpus is not large. A binary split simplifies the problem so useful interaction patterns can probably be extracted. Second, binary divide using median learning outcome (normalized learning gain) value makes each class with similar amount of samples. An unbiased corpus makes it easier to train an effective player outcome predictor. Third, a binary divide is also a conventional way to preprocess player outcome data in educational interactive narrative settings (Rowe & Lester, 2015; Rowe et al., 2014; Wang, Rowe, Mott, & Lester, 2016).

The input features to the player simulation model are designed to represent key player attributes, including how the player interacts with the virtual environment as well as how the interactive narrative planner adapts events to shape the player’s experience. This set of 21 features has been defined and concretely introduced in Section 3.2.

Figure 4.1: Example of interactive narrative planner action timeline and player action timeline.

The interaction of simulated players and the narrative planners is illustrated in Figure 4.1. In Figure 4.1(b), the simulated player is demonstrated as sampling and adopting a series of player actions $u_t$ along the player action timeline. A certain subset of player actions (e.g.,
and \( u_7 \) in Figure 4.1(b)) would trigger narrative adaptation opportunities following the game rule. Then an interactive narrative planner would pick a narrative adaptation action \( a_{t_t}^{e_t} \) from optional adaptations for adaptable event type \( e_t \) to affect the narrative. One thing to notice in this design is that, player actions in simulated player model and narrative planner’s adaptation actions from interactive narrative planners are depicted in different granularity. It is acceptable to model player’s behavior (actions) in a finer granularity because simulated player model is formed under a supervised learning framework, which is relatively easier to train. Interactive narrative planner’s behavior, when trained under the reinforcement learning framework, would be hard to optimize with long interaction sequences and sparse reward signals. So, in this dissertation research, I adopt a coarsely grained description for interactive narrative planner’s adaptation (compared to player action description’s granularity). Because players are the active drivers of the interaction process, it is natural to integrate the rules of how narrative adaptation is triggered. Also, extra rules might be integrated to prohibit certain player interaction trajectories to be sampled, which introduces more prior knowledge in generating synthetic simulated player interaction trajectories. This means compared to training an adaptable interactive narrative planner directly from human generating corpus, training a simulated player model using the corpus and logical game rules might improve the induction of effective adaptable data-driven interactive narrative planners.

### 4.2 Neural Network Models for Sequence Data Modeling

To solve a classification problem as defined in Section 4.1 for predicting players’ actions and outcomes while they play an interactive narrative-centered game, there are multiple choices of classification models to adopt, for example logistic regression, random forest, etc. For
gameplay predictions, one important property of the interaction data is that they are sequences following the same format along the time axis. Although multiple methods, including sliding window method, recurrent sliding windows, hidden Markov model and conditional random fields et., could be utilized in dealing with machine learning problems utilizing sequence data (Dietterich, 2002), deep learning methods offer an especially effective set of models in solving the simulated player modeling problem with sequence data. Several neural network models, including recurrent neural networks (RNNs) and convolutional neural networks (CNNs) have demonstrated impressive performance in processing and predicting sequence data in multiple fields including language modeling, machine translation and speech generation (Kumar et al., 2016; Luong et al., 2015; Wu et al., 2016; Arik et al., 2017; Wang, Skerry-Ryan, Stanton, Wu, Weiss, Jaitly, Yang, Xiao, Chen, Bengio, & Le, 2017). Compared to other methods, deep learning methods give the advantage that no prior expert knowledge is required in forming an effective feature extractor. A properly designed neural network (NN) model should be able to extract useful sequential patterns and encode historical information all by itself, even though the choice of potential neural network structures still needs some human expertise and extra experiments.

Recurrent neural networks (RNNs) dedicate an effective NN-based option for modeling sequence data (Graves et al., 2013). The major difference of RNNs from vanilla fully connected neural network is that a recurrent layer has a connection to itself, whose functionality is to deliver the output of the recurrent layer from current time step as an input to the next time step, so that an encoded history by the recurrent layer could propagate through the network along the time axis. The structure of a vanilla RNN model is shown in Figure 4.2.
In Figure 4.2, the middle layer represents a recurrent layer, in which a self-pointed connection is constructed. For the ease of reading, the RNN model could be unfolded along the time axis, illustrating the way how recurrent layer propagates history information along time axis (right part in Figure 4.2). If we use $s_t$ to represent the output of the recurrent layer at time step $t$, the calculation of $s_t$ follows:

$$s_t = f(Ux_t + Ws_{t-1})$$  \hspace{1cm} (4.1)

in which $x_t$ is the input into this recurrent layer at time step $t$, $s_{t-1}$ is the output of this recurrent layer from the previous time step. $U$ and $W$ are trainable parameters in the network, and $f$ is an activation function, usually can be sigmoid function or tanh function. In Figure 4.2, $o_t$ is the output of the whole RNN. In case it is a one recurrent layer NN model, the output can be a linear function of $s_t$, or with an extra non-linearity conducted upon it. As demonstrated in Equation 4.1, a recurrent network encodes its history information in $s_{t-1}$, which is a representation from the recurrent hidden layer containing information from all prior time steps. In this case, the input of the RNN model can only takes information from the current time step, and don’t have to employ a sliding window which may increase the dimension of input feature set.

Convolutional neural networks (CNNs) process sequence data in a different way compared with RNNs. Instead of keeping an encoded hidden representation for interaction
history, CNNs utilize a series of observations across multiple timesteps in inference. The core idea in CNN is that convolutional kernels apply the same processing on the input features at each time step, so that invariant patterns along the time axis can be identified. The domain of simulated player modeling in interactive narrative environment properly fits this framework. Details of utilizing CNN models in simulated player modeling will be covered in Section 4.5.

4.3 Long Short-Term Memory Network for Sequence Data Modeling

Although RNNs contribute well to sequence data modeling, vanilla RNNs are not commonly utilized in real world problems. The major reason is that vanilla RNNs are not good at processing long dependencies (Bengio et al., 1994). RNNs use back propagation through time (BPTT) in its optimization, which uses chain rule to propagate error from later time step backwards to early time step (opposite to the direction of how information is propagated in prediction). The chain rule multiplies the gradient of error w.r.t RNN parameters along the time axis, so when the sequence is very long, the potential appearance of several small gradient will make the error hard to propagate backwards for a long distance. The so-called gradient vanishing problem prohibit vanilla RNNs to learn well on sequences with long dependences. To solve the gradient vanishing problem, long short-term memory network (Hochreiter & Schmidhuber, 1997) was proposed.

Long short-term memory (LSTM) network is a special kind of RNN. It uses gate structures to control the way how information flows in the recurrent layer in an RNN. Specifically, we adopt an implementation of LSTM network in (Graves et al., 2013), which is one commonly utilized LSTM implementation:

\[ i_t = \sigma(W_{xi}x_t + W_{hi} h_{t-1} + W_{ci} c_{t-1} + b_i) \]  

\[ (4.2) \]
\[ f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \]  \hspace{1cm} (4.3) \\
\[ c_t = f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \] \hspace{1cm} (4.4) \\
\[ o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \] \hspace{1cm} (4.5) \\
\[ h_t = o_t \tanh(c_t) \] \hspace{1cm} (4.6)

Equations 4.2 to 4.6 concretely describe how LSTM works. In an LSTM cell, three gate structures are adopted to control how sequence information flows. Input gates (Equation 4.2), forget gate (Equation 4.3) and output gate (Equation 4.5) are all trainable gate structures. Input from current time step \( x_t \), hidden value from prior time step \( h_{t-1} \) and cell value from prior time step \( c_{t-1} \) are features to determine the behavior of the three gates. Weight parameters \( Ws \) and bias parameter \( bs \) (Equation 4.2, 4.3, 4.5) are learned to properly define the weights to inputs at current time step \( x_t, h_{t-1} \), encoded information from prior partial sequence \( c_{t-1} \), and encoded information at current time step \( c_t \), for their contribution to the final output of the LSTM cell \( h_t \). The cell activation value \( c_t \) performs as an encoding of the information gathered from current input and all compressed history information.

LSTM has demonstrated its power in plenty of sequence modeling problems (Graves et al., 2013). For simulated player modeling, LSTM dedicates an off-the-shelf model to properly encode the interaction sequence between players and the narrative planner. So, without storing explicit interaction history sequence, LSTM is able to describe this sequential interaction by properly extracting sequence correlation patterns, and appropriately modeling long-term dependences, which might be important in predicting players’ future behaviors and interaction outcomes.
4.4 Deep Recurrent Neural Networks for Sequence Data Modeling

An important property of neural network models is that multiple hyperparameters can be tuned, and hyperparameter tuning may have significant effects on neural network models’ performance. The depth of a neural network is such a hyperparameter, which significantly affects the complexity of the NN function approximator, as well as the number of learnable parameters to train. In recent years, both theoretical analysis (Bengio, 2009; Srivastava et al., 2015) and empirical evidence (Zilly et al., 2017; Kalchbrenner et al., 2016; He et al., 2016) support the idea that deeper NN models have clear advantages over shallow NN models in forming effectively learners. In this section, the topic of constructing deep RNN models to effectively utilize sequence data in simulated player modeling is discussed. Besides the deep LSTM model, which is a straightforward extension over the shallow LSTM network covered in the prior section, two successfully applied deep RNN models, Grid LSTM model and Recurrent Highway Network models, are introduced.

Deep LSTM models are a simple extension to the single LSTM layer NN model. Lower layer LSTM unit propagates its output $h_t$ (as in Equation 4.6) not only to itself at the next timestep, but also the upper layer LSTM unit at the same timestep as the input $x_t$ (as in Equation 4.2, 4.3, and 4.5). A stacked multilayer LSTM network forms a hierarchical representation of the sequence data, as well as keeps encoded historical information naturally in each of the LSTM layer’s hidden units, so that the more abstracted sequence feature from higher level LSTM unit would be able to improve the learning task effectively. It is important to note that in deep LSTM networks, the cell activation output $c_t$ (as in Equation 4.4) in an LSTM unit is not propagated upwards to higher layers. This is an important difference from the Grid-LSTM model that will be introduced later in this section. Another important factor
about deep LSTM model is that, stacking multiple layers of NNs, especially for networks with complex structures (e.g., LSTM unit with multiple gate units), will hugely increase the number of trainable parameters, correspondingly increase the difficulty of training. Thus, proper regularization technique is usually adopted, which will be covered in next sub-section.

Grid-LSTM, as an extension of the vanilla multilayer stacked LSTM network, is proposed to construct easy-to-train deep LSTM networks (Kalchbrenner et al., 2016). Although deep NN models are more powerful in approximating any complex functions, the increase of NN model depth raises the difficulty in effectively training the model. The Grid-LSTM model innovatively utilizes gate units in forming multilayer LSTM models. The gate units employed in an LSTM unit in vanilla LSTM networks improve the model’s capability of remembering long term dependency and reduces the difficulty of gradient vanishing in back propagation through time. Grid-LSTM adopts this idea in constructing stacked multilayer LSTMs. Along the dimension of NN layers, Grid-LSTM builds cell activation output and normal hidden output, just as how vanilla LSTM uses them along the sequence dimension. Usually, a separate set of trainable parameters will be adopted in learning the two sets of outputs. However, we can also let the Grid-LSTM unit shares trainable parameters along different propagation dimensions to prohibit the number of parameters grow too quick (Kalchbrenner et al., 2016). With the extra gate units in learning hidden representations along the depth dimension, Grid-LSTM is supposed to construct deeper LSTM models that are relatively easier to train.
The third deep RNN model that is investigated in this dissertation in for sequence data (especially player interaction sequence) modeling is recurrent highway networks (RHNs). Recurrent highway networks (RHNs) construct recurrent NNs using highway layers (Zilly et al., 2017). A highway layer calculates its output as a weighted summation of an input’s non-linear transformation and the input itself. This design has proven effective for building deep and trainable NNs (Srivastava et al., 2015). Specifically, RHNs increase network depth by utilizing a high recurrence depth in the recurrent step transition: at each transition time step, multiple highway layers (yellow blocks in Figure 4.4) are connected. A 3-layer RHN with recurrence depth of 3 is depicted in Figure 4.4. The output of the adopted RHN model at recurrence depth \( l \) and time step \( t \) on any RHN layer (the output of any yellow block in Figure 4.4) is calculated as follows:

\[
s_t^l = h_t^l \cdot r_t^l + s_{t-1}^l \cdot (1 - r_t^l)
\]

in which \( h_t^l \) is a nonlinear transformation of \( s_{t-1}^l \) with \( s_0^l \) being the recurrent output from time step \( t-1 \), and \( r_t^l \) is the transform gate that controls the contribution of \( h_t^l \) into the output \( s_t^l \). Calculation of \( h_t^l \) and \( r_t^l \) is as follows:
where $x^t$ is input into one RHN layer at time step $t$, $\mathbb{I}$ is the identity function, and $W, R, b$ are trainable weights and biases in RHN.

![RHN model for player action prediction.](image)

**Figure 4.4: RHN model for player action prediction.**

### 4.5 Deep Convolutional Neural Networks for Sequence Data Modeling

Convolutional neural networks (CNNs) are another major branch of NN models that are investigated in this dissertation. The major computational process adopted in a CNN model is described as “convolution,” which sums up the values of a subset of features by adopting weights in the convolution kernel. Convolution kernels usually have a size smaller than the input size, and the convolution process slides the kernel on the input feature set to conduct dot products centered at certain positions in the input. In this way, convolution can process and extract regional invariant features that could appear at different positions in the input feature set with properly defined convolution kernels. Training of a CNN model learns meaningful kernel values, so representative invariant features to describe the original input can be extracted. Usually CNN models also adopt pooling modules, which are non-learnable...
layers mainly utilized for down-sampling purposes. Since player interaction data usually does not have very high dimension. In this dissertation pooling is not adopted in CNN models.

Because player outcomes are typically measured only after the conclusion of an interactive narrative experience, outcome data tends to be sparser than player action data. To address this problem, it is especially interesting to investigate the effectiveness of convolutional NN-based (CNN) models for player outcome prediction. CNN-based player outcome prediction models can detect repeatable patterns from each gameplay time step and eliminate the constraint imposed by recurrent NN models of having to extract patterns using the exact input sequences.

CNN models can be leveraged in simulated player modeling because the same feature set is usually utilized at each gameplay time step to record the player’s behaviors, thereby enabling convolution to be naturally applied along the time axis. This strategy has been used in machine translation tasks (Gehring et al., 2017). One example of utilizing CNN models in simulated player modeling is the CNN-based player outcome prediction model that is experimented in this dissertation, which has two convolution layers stacked above the input layer (Figure 4.5).

As noted in Section 4.4, depth is also a vital property of CNN models, which directly determines the capability of the model in approximating function. To make deep CNN models easy to train, the residual learning method in (He et al., 2016) is studied and utilized in this dissertation. Residual structure in NNs builds a direct path from input to output so that each layer with a residual connection learns a residual function with reference to the layer’s input, which has proven effective in training deep CNN models. Compared with the highway unit studied in Section 4.4, the major difference is that highway units learn a gate structure to
assign weights to lower layer raw input and non-linearly transformed values, while residual learning does not learn a gate structure but directly propagates lower layer values into upper layers, which makes the structure of the network simpler, and empirically to be more efficient in learning very deep CNN models (He et al., 2016). Specifically, we adopt a residual structure that directly adds raw input into the output of each of the two convolution layers. In this way, the output of convolution layer \( l \) is \( o_l = \text{conv}(i_l) + i_l \) in which \( i_l \) represents the input into convolution layer \( l \).

![Figure 4.5: CNN model for player outcome prediction.](image)

### 4.6 Regularization Techniques in Neural Network Models

As a universal function approximator, neural network models can be ultimately significantly powerful and complex by increasing the size of hidden layers or stacking a larger number of hidden layers in a NN model. Although more complex models can potentially extract more complicated nonlinear patterns, the probability of overfitting also grows dramatically along with the increasing model complexity. To effectively resist overfitting, properly
regularization techniques should be adopted, especially for the problem of simulated player modeling, in which the human generating corpus is usually small.

In this dissertation, we will cover the successfully utilized regularization techniques in neural network model training in the problem of simulated player modeling and RL-based interactive narrative generation.

4.6.1 Dropout

Deep neural network models are powerful function approximators, which also introduces the problem that overfitting could easily happen with a large number of trainable parameters in deep NN models. Besides regular and universal regularization techniques (e.g., L1-norm, L2-norm, etc.) that are also broadly investigated with other machine learning models, dropout technique (Srivastava et al. 2014), as a broadly adopted NN specific regularization tool, has achieved particular success and extensive usage in plenty of domains (He et al., 2016; Zilly et al., 2017).

From the perspective of implementation, dropout is a very easy to use technique. For each layer of a NN model, the dropout technique randomly turns off the output of a specific proportion of neurons during training, in which the proportion hyperparameter is usually called the dropout rate. In this case, only a subset of hidden neurons would contribute their values to upper layers during the training period of the model, while the dropout mask that controls which subset of neurons could propagate their values to upper layers, is resampled with a specific dropout rate for each single training data point. At test time, resampling of dropout mask is prohibited, and all neurons would contribute their output to the calculation of the upper layers, with their values usually weighted by the dropout rate to control the magnitude of the values from a single layer. According to (Srivastava et al. 2014), dropout
prevents neurons on the same layer from co-adapting too much. During training, a randomly selected subset of neurons is kept, while the full set of neurons are used at test set. Thus, dropout serves also as an ensemble technique, in which an exponentially large smaller network is trained and an ensemble of all these smaller learners are utilized during evaluation. Dropout has demonstrated impressive regularization effects for training large neural networks.

Despite the great regularization effect in training NN models, the original dropout technique is not an off-the-shelf tool for training recurrent neural networks. Directly utilizing dropout in an RNN model with the dropout mask resampled every time usually will not generate a model with convincing results (Zaremba et al., 2014), with the potential explanation that noise accumulated along the time axis in an RNN model would be amplified for long sequences. Previously, researchers proposed to only adopt dropout layer-wisely, and not do dropout along the time axis for RNN models (Pham et al., 2014). In (Zaremba et al., 2014), a new variational inference-based dropout technique is introduced. This dropout technique utilities a fixed dropout mask on each recurrent unit for each unique sequence. It resamples the dropout mask when processing the next sequence. So, the same set of features will be kept for each sequence during training. This technique has shown better results than classic dropout techniques in recurrent NN regularization. In this dissertation, variational inference-based dropout is utilized for regularizing all RNN models to achieve better generalization performance.

4.6.2 Mixture of Experts

Mixture of experts (ME) method is an ensemble method that has seen common usage and easy configuration in neural network-based machine learning models. ME is established
based on the divide-and-conquer principle in which the original problem space is divided into several smaller sub-spaces, and a group of sub-learners are assigned to solve the problem in the sub-spaces. The assignment of sub-learners into each problem sub-space is supervised by a gating structure, whose behavior can also be learned (Masoudnia & Ebrahimpour, 2014).

![Figure 4.6: Mixture of Experts model structure (Masoudnia & Ebrahimpour, 2014).](image)

The structure of an ME neural network model could be illustrated in Figure 4.6. From the input layer, multiple parallel sub-learners could be constructed. These sub-learners can be fully connected multi-layer perceptrons, or other NNs with complex modules like CNN. A gating network is also built upon the input layer. With the same input, the gating network learns a proper weighting mechanism on the output of each sub-learner (e.g., $MLP_i$ in Figure 4.3), so that the final output of the whole ME network is the weighted summation of the outputs of each sub-learner:

$$o_{ens} = \sum_i o_i g_i$$  \hspace{1cm} (4.10)  

$$o_i = f(W_{o_i}x + b_{o_i})$$  \hspace{1cm} (4.11)  

$$g = f(W_{g}x + b_{g})$$  \hspace{1cm} (4.12)

in which $o_i$ and $g$ are the output of the $i$th sub-learner and the gating network. $g_i$ is the $i$th item in the $g$ output which represents a weight to $o_i$ in the final output $o_{ens}$ calculation.
Equations 4.11 and 4.12 match the situation when fully connected NNs are used as sub-learners, which is the use case in this dissertation. But a more general ME network can be easily constructed by substituting the linear transformation in Equations 4.11 and 4.12 with other calculations (e.g., convolution in CNN models). Usually, a softmax layer is stacked on top of the gating network, so each of its outputs (weights) would be properly ranged in (0,1).

Figure 4.5 illustrates an example of adopting a “mixture of experts” technique in constructing player outcome predictors. In this case, the CNN based lower layers are shared by each logistic regression predictor, for the purpose of keeping the size of trainable parameters in control. A combination of multiple logistic regression predictor as well as a gate module are used as an ensemble of multiple weak classifiers to predict the player outcome type from interaction data.

4.6.3 Multi-Task Learning

Multi-task learning is another broadly utilized regularization method in deep learning. According to Caruana (1997), multi-task learning is an inductive transfer mechanism whose principle purpose is to improve the model’s generalizability performance. With multi-task learning, a machine learning model can utilize domain-specific information contained in the training data across multiple tasks. When multiple tasks are correlated, this technique has the potential to improve the model’s performance on each task.

Multi-task learning can be a highly promising technique in the task of simulated player modeling in games. With a limited sized corpus usually adopted in simulated player modeling problem, the multi-task mechanism is possible to boost the model’s performance by utilizing information from multiple domains. Examples of adopting multi-task learning in simulated player modeling could include the followings: Player action prediction and player
outcome prediction in simulated player modeling are distinct but related tasks. In interactive narrative personalization, players with positive experiential outcomes may exhibit different patterns of actions than players with negative experiential outcomes. We could exploit this relationship by utilizing a multi-task NN architecture for player simulation, which integrates player action prediction and player outcome prediction into a single model. Another thought, exploiting the system design pattern that player’s outcome prediction is formed as a binary classification problem, urge to conduct both the tasks of predicting the discrete player outcome type, as well as predicting the real-valued player outcome score, in a single prediction model. Although these two tasks are framed through a shared data stream, distinct errors would be filled into the optimization pipeline due to the difference in loss function design in a classification setting and a regression setting. So, it is potentially helpful to construct a more powerful simulated player model, especially for the task of player outcome prediction, considering the sparser dataset used in the player outcome prediction task.

Multi-task learning is fairly easy to implement in deep neural network models. Because deep neural networks learn multi-level hierarchical features across multiple hidden layers, it is possible for multi-task deep neural networks to utilize shared lower layers with separate output layers targeting individual tasks. This architecture enables the model to leverage cross-task information by sharing abstract features learned in shared hidden layers in neural networks. During training, errors from each output layer are back-propagated into the shared lower layers. Consequently, extraction of hierarchical features in the shared hidden layers is determined by multiple outputs in a multi-task NN architecture. Figure 4.7 illustrates a multi-task LSTM model for predicting player actions and player outcomes simultaneously. Two tasks of player action and player outcome predictions both contribute
parameter gradients in optimization for the shared hidden LSTM layer. In case these two tasks are highly correlated, it is potential that the performance on each task will be properly boosted by utilizing patterns learned also from the other task. Figure 4.7 demonstrates a specific structure to construct multi-task neural network models. However, this regularization technique is very generalizable. In case the player outcome type (classification) and player outcome score (regression) tasks are formed together, we only need to substitute the player action predictor output layer correspondingly. Hidden layer design is also generalizable, so the LSTM hidden layer in Figure 4.7 can be displaced by any other neural network structures, if they are rational choices in a specific problem domain. Figure 4.5 also demonstrates the utilization of multi-task learning outcomes design within CNN models. In that case, a player outcome score predictor (regressor) and a player outcome category predictor (classifier) are adopted and different errors from different loss functions (mean squared error function and cross entropy loss function) help the training of the shared layer parameters, so the learned model would be beneficial in both tasks.

Figure 4.7: Multi-task LSTM network implementation of a simulated player model for interactive narrative planning.
In this section, we present reinforcement learning-based interactive narrative planning. Specifically, the approach of utilizing reinforcement learning (RL) methods in deriving adaptable interactive narrative planners for the CRYSTAL ISLAND interactive narrative is described. The organization of this section is as follows: in Section 5.1, we introduce a reinforcement learning framework that can be utilized in interactive narrative generation and adaptation. In Section 5.2, Q-network-based interactive narrative planning is discussed. Several important factors in constructing Q-networks are presented, including the stabilization technique used to help the convergence of training, recurrent Q-networks used for modeling sequential interaction patterns in reinforcement learning, and dueling structures in Q-networks to implement advantage learning. In Section 5.3, actor-critic reinforcement learning methods are introduced. Specifically, asynchronous advantage actor-critic method and proximal policy optimization are explained with their application in interactive narrative planning. In Section 5.4, the exploratory strategies of reinforcement learning methods are analyzed and compared. It is expected that reinforcement learning-based interactive narrative planner’s generalizability could be highly correlated with the exploratory strategies the agent adopts.

5.1 Reinforcement Learning Framework for Interactive Narrative Planning

For interactive narrative planning problems, RL provides a natural computational framework because of its capacity to model sequential decision-making tasks in uncertain environments with delayed rewards, i.e., player outcomes. We represent interactions between a narrative
planner and a player as a series of stochastic state transitions, which are influenced by the narrative planner’s run-time adaptations, which drive players’ interactive narrative outcomes.

More formally, when a player conducts a series of player actions and triggers an adaptable event $e$ at interactive narrative planning time step $t$, the interactive narrative planner chooses an action $a^e_v$ from a discrete action set $A^e = \{a^{e_1}, a^{e_2}, ..., a^{e_m}\}$ of event $e$. Decisions about adaptable narrative events are driven by planning policy $\pi$ and the current narrative interaction state $s_t \in S = (o_{t-n+1}, ..., o_t)$, in which $o_t$ is the observation at interactive narrative planning time step $t$, and $n$ is the number of observations encoded in the state representation. The interactive narrative environment proceeds to the state $s_{t+1}$ and reward signal $r_t$ is administered according to a narrative experience quality metric. Training RL-based interactive narrative planners gradually adjusts the interactive narrative planning policy $\pi$ in order to optimize the expected discounted cumulative reward $R_t = \sum_{t'=t}^{\infty} \gamma^{t'-t} r_{t'}$ obtained by the interactive narrative planner, where the discount factor $\gamma \in [0,1]$. The output of RL training is an optimal policy $\pi^*$, which encodes the optimal narrative planning action to perform in each state $s$.

In CRYSTAL ISLAND, the RL narrative planner utilizes 25 features for each observation $o_t$, of which 21 real-valued features are the ones used to represent a simulated player’s gameplay state (described in Section 3.2), and 4 additional features form a one-hot encoding for the adaptable event index (described in Section 3.3). Among the four adaptable events, in total 10 optional discrete narrative adaptation actions exist (described in Section 3.3). For these discrete actions defined for the narrative planner, the narrative planner will select an optional action from the subset of actions that fit the adaptable event type, which can be told from the adaptable event index. A discount factor $\gamma = 0.99$ is set to slightly
encourage quick learning for players. The structure of the whole data-driven interactive narrative planning system can be illustrated in Figure 5.1.

Figure 5.1: Deep reinforcement learning-based interactive narrative personalization framework.

As shown in Figure 5.1, the deep RL interactive narrative planning framework has a 4-tier structure. We illustrate the architecture with respect to two principal phases: policy learning and policy evaluation. In the Dataset Tier, human player game interaction and questionnaire data were collected from two studies involving in total 402 students, with all incomplete records eliminated. In the Player Simulation Tier, using features defined in Section 3.2, one player simulation module learns to predict players’ next actions at each player interaction time step, and the other module learns to predict the players’ learning outcomes at the end of narrative interaction. During the RL-based interactive narrative planner’s training period, the simulated player’s behavior, as input, is sent into the RL Tier, where a deep RL agent (performs as an interactive narrative planner) is trained to improve the interactive narrative planning policy in an optimization process. During evaluation, the
interactive narrative planner from the Narrative Adaptation Tier utilizes and assesses the optimized policy by interacting with simulated players trained on the test set. Multiple simulated player models can be trained and employed in both interactive narrative planner training and evaluation, especially for the purpose of assessing the derived interactive narrative planning policy’s generalizability. The specific design of the experiment will be introduced in the following section.

5.2 Q-Network-Based Interactive Narrative Planning

In this subsection, one family of value function-based deep RL methods—Q-network, is presented, especially with its application in adaptable interactive narrative planner derivation. Value function-based RL approaches induce the optimal policy for RL agents by maintaining and optimizing the estimation of expected discounted cumulative rewards the RL agent can obtain from each state, or state-action pair, and then following a specific policy in the interaction with the RL environment. An (locally) optimal policy can be derived from the estimated state or state-action pair values for that (locally) optimal policy. A deterministic form of optimal policy selects actions greedily under each state according to action function values, and a stochastic optimal policy generates action selection probability distribution in proportional to the action value estimation. Specifically, Q-network is a neural network-based implementation of the broadly utilized off-policy model-free RL method, Q-learning (Watkins & Dayan, 1992). The off-policy property allows Q-learning to learn the value function of an effective policy by observing RL agent’s interaction history under a sampling policy, which can be a simple and exploratory policy in the problem domain. State-action pair values are optimized according to Equation 5.1, with $\alpha$ representing the learning rate, until approximation of the Q function converges.
\[ Q_\pi(s, a) = Q_\pi(s, a) + \alpha [r_{s,s'} + \gamma \max_{a'} Q_\pi(s', a') - Q_\pi(s, a)] \]  

(5.1)

In Equation 5.1, the \( Q_\pi(s, a) \) represents the estimated expected cumulative reward by performing action \( a \) at state \( s \) and then follows policy \( \pi \). All the other symbols in Equation 5.1 have the same meanings as they’ve been defined in Section 5.1. Q-learning is guaranteed to converge to the optimal policy in case a linear function approximator is utilized and proper learning rate is adopted. Even with non-linear function approximator, Q-learning usually also induces effective policies. When the learning process converges, with the optimal \( Q^*(s, a) \) values, the optimal deterministic policy could be derived as:

\[ \pi^*(s) = \arg\max_a Q^*(s, a) \]  

(5.2)

Figure 5.2(a) illustrates one type of Q-network implementation structure, which utilizes multiple steps of observation in state representation, and fully connected layers as hidden layers. The structure of Q-networks can vary according to the specific properties of the problem and the adopted state feature representation. For example, CNN layers can be used to construct the lower layers of the Q-network, in case input features are time sequences or image data. Specifically, for interactive narrative interaction data, as utilized in this dissertation, either fully connected or CNN layers can be used to construct the interactive narrative planning Q-network.

5.2.1 Deep Reinforcement Learning Stabilization

Utilizing NNs enables RL interactive narrative planners to extract complex nonlinear interaction patterns. However, the training process of deep RL agents (e.g., Q-networks) can be unstable, or even diverge (Tsitsiklis & Roy, 1997) because of the correlations in the sequence of neighboring interactions (Mnih et al., 2015). To stabilize the deep RL interactive narrative planning policy training, three techniques are investigated in this dissertation. The
first stabilization method is experience replay (Lin, 1993). By applying experience replay, instead of updating Q functions directly using the instant experience $e_t = (s_t, a_t, r_t, s_{t+1})$ from the observation of real-time RL agent and RL environment interaction, all interaction experiences are stored into a dataset, then a minibatch of experiences are randomly sampled from the dataset at each training step. This approach breaks the correlations in neighboring interactions by rearranging sequences of experiences used in RL agent training.

The second stabilization technique applied in this dissertation is separate target network (Mnih et al., 2015). With this method, instead of employing a single NN to calculate both $Q(s_t, a_t)$ and the temporal difference target $r_t + \max_a' Q(s_{t+1}, a')$ in Q-learning, a separate target network is copied from the training Q-network every certain steps and held fixed between individual updates. The target network is only used in temporal difference target estimation. By freezing the target network for certain steps, a less frequently changing target function is obtained, so the training process can be stabilized. Using $\theta$ and $\theta^-$ to represent the training Q-network parameters and separate target Q-network parameters respectively, Q-learning can be executed with any gradient descent-based optimization method using the gradient of the loss function $L(\theta)$ as shown in Equation 5.3 and 5.4. Note that we remove the subscript for time step $t$, in $s$, $a$ and $r$, to make the expectation not specific to a certain time.

$$\nabla_\theta L(\theta) = \mathbb{E}_{s,a,r,s'}[(Q(s,a;\theta) - y)\nabla_\theta Q(s,a;\theta)]$$ (5.3)

$$y = r + \gamma \max_{a'} Q(s',a';\theta^-)$$ (5.4)

The third way we exploit to stabilize Q-network training is the asynchronous gradient descent optimization (Mnih et al., 2016). The core idea of implementing an asynchronous RL interactive narrative planner architecture is that, instead of using one interactive narrative
planner to interact with one player and learn from the interactions, multiple RL interactive narrative planners can be set up to interact with multiple players in parallel. By accumulating gradients of loss from each RL interactive narrative planner with respect to the shared Q-network parameters, the training of the Q-network can be stabilized using all current experiences without the need to build the experience replay dataset. In the implementation of asynchronous gradient descent optimization for Q-network-based interactive narrative planner, multiple threads can be utilized in parallel, with each thread maintains one copy of the Q-network model and interacts with a separate simulated player. While each Q-network from a separate thread generates unique gradient of the loss function, all threads share the weights of the Q-network. Just as how mini-batch learning is used in supervised learning, multiple samples from parallel threads offer a better estimation to the loss function gradient because of the decreased variance of the gradient approximation, in which way the whole optimization process can be stabilized.

5.2.2 Recurrent Q-Network

Although the player–interactive narrative planner interaction has often been assumed Markovian in previous RL-based interactive narrative personalization work (Rowe et al., 2014; Rowe & Lester, 2015; Wang, Rowe, Mott, & Lester et al., 2016), this assumption might be too strong, considering the features researchers normally utilize in describing this interaction process. The usual feature set representing player’s interaction process is limited in its representation capacity, so it is hard to maintain a proper representation meaningfully describing players’ interaction history. This situation gets even worse when player’s learning outcome is utilized as the metric to evaluate the quality of the interaction and the generated interactive narrative. With these limitations, it is almost impossible to extract Markovian
state representations in depicting the player—interactive narrative planner interaction. To model interactions in such a partially observable environment, a common solution is to stack a fixed length of observation history into the RL state representation, usually represented as $s_t = (o_{t-n+1}, \ldots, o_t)$. However, there are other ways to concisely encode long-term history, such as the adoption of RNNs. Specifically, the recurrent Q-network embeds an RNN layer (e.g., LSTM layer) within the Q-network (Hausknecht & Stone, 2015), which enables the Q-network to effectively preserve a long-term memory in the narrative but maintains a compact state representation as $s_t = o_t$. This structure may help in situations when long interaction patterns influence interactive narrative quality severely, i.e., late stage interactive narrative planning decisions being strongly affected by early stage events.

Compared with the non-recurrent Q-network models with stacked observation history representation $s_t = (o_{t-n+1}, \ldots, o_t)$, recurrent Q-network has a lower dimension in the Q-network input, which affects the scale of policy searching for the RL-based interactive narrative planner training. On the other hand, utilizing recurrent module in Q-network does not have to be positively effective, because recurrent modules (e.g., LSTM modules) usually introduce relatively a large number of trainable parameters into Q-networks, which makes the training even harder. Hausknecht and Stone also find that it can be harmful if error information from very far future is utilized in back propagation through time (BPTT) in recurrent Q-network training, which empirically demonstrates that extra efforts are needed to tune recurrent Q-networks. Interactive narrative adaptation is a problem in which the state transition distribution and reward distribution can be of large variance. This property enhances the necessity to study a proper way to encode interaction history in Q-network-based method. Figure 5.2(c) illustrates one implementation of the recurrent Q-network.
structure. As demonstrated, the recurrent Q-network uses an LSTM layer as a hidden layer to encode historical representation. At the same time, the input layer only need deal with single time step observation as state representation, which makes the input dimension lower than it for the original Q-network (Figure 5.2(a)).

5.2.3 Dueling Q-Network

Because state and policy spaces in interactive narrative personalization problems are often vast, utilizing advantage learning (Harmon et al, 1995) by evaluating state values and advantage values independently can be valuable for learning an effective policy. Advantage learning extended value function-based RL by introducing the advantage function, which measures the advantage of taking a specific action at a state over the estimated state value under a policy $\pi$:

$$Q^\pi(s,a) = V^\pi(s) + A^\pi(s,a)$$  \hspace{1cm} (5.5)

so, in advantage learning, the RL agent keeps the estimations of both $V^\pi(s)$ and $A^\pi(s,a)$. By adopting advantage learning, state value $V^\pi(s)$ function gives the RL agent an estimation of the “goodness” of a state, even without all actions being thoroughly explored. This property is especially useful in cases when in certain states the RL agent’s specific action choice have bare influence for the interaction outcome. Thus, in early stages in interactive narrative planner training, advantage learning might be able to guide the interactive narrative planner to avoid “bad states” early and explore “good states” more thoroughly.

Under the Q-network structure, this notion can be implemented using a dueling Q-network (Wang, Schaul, Hessel, Van Hasselt, Lanctot, & De Freitas, 2016). As shown in Equation 5.6, one action’s Q values can be represented by the summation of a state value V and an action advantage value A, parameterized by the trainable parameters in the neural
network. In Equation 5.6, $\theta'$ represents parameters of the Q-network below the dueling layer, and $\alpha, \beta$ represent parameters in the advantage stream and state stream, respectively (as shown in Figure 5.2(b)). A merging module sums the output from these two streams to obtain the same output format as other Q-networks.

$$Q(s, a; \theta', \alpha, \beta) = V(s; \theta', \beta) + A(s, a; \theta', \alpha)$$  \hspace{1cm} (5.6)

Although Equation 5.6 looks straightforward to implement, there is one potential problem in the dueling Q-network training if directly adopting advantage learning as it is shown in Equation 5.6. Equation 5.6 is not identifiable considering that the function $V$ and $A$ cannot be uniquely identified given the target value of $Q$. This lack of identifiability could negatively influence the performance of the dueling Q-network. Future, we adopt a slightly modified form of Equation 5.6 from (Wang, Schaul, Hessel, Van Hasselt, Lanctot, & De Freitas, 2016), in which we maintain an averaged advantage values across all optional actions at state $s$:

$$Q(s, a; \theta', \alpha, \beta) = V(s; \theta', \beta) + \left(A(s, a; \theta', \alpha) - \frac{1}{|A|} \sum_{a'} A(s, a'; \theta', \alpha) \right)$$  \hspace{1cm} (5.7)

In this case, although the meaning of advantage value is changed a little bit, this form of advantage learning could still enable the RL agent to get early estimation of state values and stabilize the training of the dueling Q-network. Figure 5.2(b) illustrates one potential implementation structure of the dueling Q-network used in interactive narrative planner training. As claimed for other Q-network structure implementation, it is also true that the dueling Q-network can have different implementation structures. The duel stream structure is compatible with plenty of other neural network layers if properly configured.
Besides value function-based reinforcement learning methods, actor-critic is another category of broadly utilized RL methods in modern RL research (Mnih et al., 2016; Schulman et al., 2017; Wang, Bapst, Heess, Mnih, Munos, Kavukcuoglu & De Freitas, 2017; Wu et al., 2017). The major idea of actor-critic RL is that, an actor module and a critic module are employed together to learn effective policies for the RL agents. An actor module maintains an explicit representation of the RL agent’s policy, which is the probability distribution of the agent taking each action in discrete action case, or the direct action output value in case of continuous action setting. The critic module, on the other hand, learns to estimate a value function of state values, or state-action values, or action advantage values, to efficiently formulate the optimization target, which drives the training of the actor module. Actor-critic methods offer another option in training RL agents for adaptable interactive narrative planner derivation. Several properties make it especially interesting to investigate its effectiveness in deriving high-quality and generalizable narrative planning policies. Actor-critic methods

Figure 5.2: Neural network structures of (a) Q-network, (b) Recurrent Q-network, and (c) Dueling Q-network.
maintain stochastic formed policies, which might be useful for interactive narrative adaptation, considering a deterministic policy may not be able to lead the agent to properly interact within the partially observable complex interactive narrative space. It’s also interesting to consider the difference of exploratory strategies utilized in value function-based RL methods and actor-critic RL methods. The exploration behavior of RL agents significantly affects the efficiency of the agent’s policy learning, as well as the capacity of the RL agent to finally learn an (locally) optimal policy.

In this dissertation, two broadly utilized actor-critic methods are investigated: Asynchronous Advantage Actor-Critic (A3C) method (Mnih et al., 2016) and Proximal Policy Optimization (PPO) method (Schulman et al., 2017). In A3C, training focuses on increasing the value of a target function as defined in Equation 5.8, in which $A^\pi_t$ is the advantage value function which can be estimated from samples using Equation 5.9. In contrast to A3C, PPO implements trust region update (Schulman et al., 2015) by utilizing a special target function as defined in Equation 5.10, so that policy updates are constrained within a threshold controlled by a hyper-parameter $\epsilon$ to improve training robustness. In Equation 5.10, $rt_t(\theta)$ represents the ratio of probability to obtain the transition sequence under the current policy $\pi_\theta$ and the probability under the sampling policy.

$$T_t = \log \pi(a_t|s_t; \theta)A^\pi_t$$ \hspace{1cm} (5.8)

$$\widehat{A}^\pi_t = \sum_{t'=t}^{\infty} \gamma^{t-t'}rt_t - V(s_t; \theta)$$ \hspace{1cm} (5.9)

$$T_t = \mathbb{E}[\min(rt_t(\theta)A^\pi_t, clip(rt_t(\theta), 1-\epsilon, 1+\epsilon)A^\pi_t)]$$ \hspace{1cm} (5.10)

5.4 Exploratory Strategy in Interactive Narrative Planning

High-fidelity simulated player models enable the investigation of the generalizability of narrative planning policies, which measures the policy’s effectiveness when interacting with
unfamiliar player populations. A narrative planning policy’s generalizability is essential to consider because the deployment environment of an interactive narrative planner is often challenging to control, and it can differ substantially from the training environment. For interactions with player populations who have different gameplay preferences, a generalizable narrative planning policy should generate meaningful and adaptive narratives.

In this dissertation, one major research interest is to investigate the effects of an RL-based narrative planner’s exploration-exploitation strategy on the narrative planning policy’s generalizability. The exploration-exploitation strategy balances an RL agent’s preference for gathering more information about the environment or utilizing its current knowledge to drive decision making. This problem is central in interactive narrative adaptation because of the pervasive high uncertainty in the interplay between human interaction and narrative adaptation. With insufficient exploration, the high variance of the state transition distribution and reward distribution may lead a narrative planner to learn a risky suboptimal policy. Without sufficient exploitation, convergence to (locally) optimal policies may not be achieved. This problem becomes even more important when narrative planners are expected to interact with unfamiliar player populations that exhibit different gameplay patterns. To explore this problem, the exploratory strategies of the RL methods investigated in the above subsections are studied and compared.

Value function-based RL methods usually adopt the exploratory strategy of $\epsilon$-greedy. In $\epsilon$-greedy, the RL agent picks the action $\text{argmax}_a Q(s, a; \theta)$ with probability $1 - \epsilon$ for exploiting the learned knowledge, or it selects a random action with probability $\epsilon$ for exploring alternative choices. Usually, a decaying schedule is set for the $\epsilon$ parameter, so the
agent explores the environment more often during its early learning stages, and it exploits
learned knowledge more often later in learning as it has gained experiences.

Because actor-critic methods keep an explicit representation of the policy \( \pi(a|s; \theta) \), in standard use their exploration strategy is to sample actions following the probability distribution in \( \pi \) so that more promising actions will be selected more often for exploitation, while other actions also have chances to be adopted for exploration. The difference in the target functions of A3C and PPO not only distinguishes their updating processes, but it also affects their exploration strategies. Because PPO prevents its updating step from being very large, policies of PPO agents change more slowly, which offers PPO agents opportunities to exploit the already learned knowledge more often, and allows PPO policies to converge easier. On the other hand, because A3C’s policy updating can be significantly influenced by each single noisy transition sequence in any training stage, A3C is encouraged to avoid early convergence to (local) optima and explore other choices more often, which holds promise for interactive narrative planning with unfamiliar players.
CHAPTER 6

EVALUATION

To evaluate the performance of the proposed data-driven adaptable interactive narrative planning framework, the following studies using the CRYSTAL ISLAND interactive narrative testbed have been conducted. The CRYSTAL ISLAND corpus described in Section 3.4 serves as the data source to derive the statistical simulated player model, based on which the RL-based interactive narrative planner is then induced. Specifically, in this dissertation we study if neural network models boost the performance of statistical simulated player modeling and adaptable narrative planning in educational interactive narratives. In this section, the performance of statistical simulated player models in interactive narratives is first evaluated. Second, based on the derived simulated player model, the effectiveness of the adaptable interactive narrative planners induced using reinforcement learning methods is accessed. Details about how the model configuration affects the proposed model’s performance are also demonstrated in the results.

6.1 Statistical Simulated Player Model Evaluation

As described in Section 4, in this dissertation statistical simulated player modeling in narrative interactions is formulated as a classification problem, which is composed of tasks including player action prediction and player outcome prediction. We evaluate the performance of simulated player models on the metrics of player action and outcome prediction accuracy rates, and macro-average F1 scores. In the CRYSTAL ISLAND dataset, 402 human players generated a total of 16,313 player actions spanning 15 player action types, of which the most commonly exhibited action type (testing object in virtual lab) consists of 24.01% of all player actions. With a median split on the player learning outcome metric of
NLG, 200 out of 402 players are labeled as high NLG players, and all the other players are labeled as low NLG players. Thus, in this set of experiments, player action prediction is cast as a 15-class classification problem, while player outcome prediction is a binary classification problem (high NLG player against low NLG player).

Before evaluating the neural network models in simulated player modeling, five commonly utilized non-NN baseline models are assessed on both tasks of player action prediction and player outcome prediction. These baselines include logistic regression (Logistic), support vector machine with a linear kernel (SVM), random forest (RF), gradient boosting (GB) (Friedman, 2002) and naïve Bayes models. All of these baseline models are implemented with Scikit-Learn (Pedregosa et al., 2011), and all hyperparameters are tuned using grid search.

For player action prediction, five-fold cross validation is conducted on the two metrics of accuracy and macro-averaged F1 score across all methods. For those non-NN baseline models, grid search hyperparameter tuning finds the linear kernel function as the best match for SVM player action predictor, out of choices including polynomial kernel and radial basis function kernel. The error penalization hyperparameter for the SVM model is set to 0.5. The random forest player action predictor sets a max tree depth of 10, number of estimators of 20, and the gradient boosting player action predictor sets a max tree depth of 5, number of estimators of 5.

In Table 6.1, the six non-deep neural network baseline models’ performance on player action prediction are listed. From the results it can be seen that logistic regression and multilayer perceptron (with 2 hidden layers) are the two best performing models in player action prediction based on the two metrics. Multilayer perceptron achieves the highest
averaged prediction accuracy of 0.3248 over the five-fold cross validation, and there is statistical difference between the performance of multilayer perceptron and logistic regression under paired t-test (t=3.3848, p=0.0277). Logistic regression achieves the highest macro-average F1 score of 0.1774 over the five-fold cross validation, and there is no statistical difference between the performance of multilayer perceptron and logistic regression under paired t-test (t=1.6907, p=0.1662).

Table 6.1: Player action prediction performance of non-deep learning models.

<table>
<thead>
<tr>
<th></th>
<th>Logistic</th>
<th>SVM</th>
<th>RF</th>
<th>GB</th>
<th>Naïve Bayes</th>
<th>MLP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pred. Acc.</td>
<td>0.3135</td>
<td>0.3118</td>
<td>0.3196</td>
<td>0.3225</td>
<td>0.0780</td>
<td><strong>0.3248</strong></td>
</tr>
<tr>
<td>Macro F1</td>
<td><strong>0.1774</strong></td>
<td>0.0653</td>
<td>0.1510</td>
<td>0.1772</td>
<td>0.0720</td>
<td>0.1698</td>
</tr>
</tbody>
</table>

Then, experiments are conducted with neural network-based models. Several NN models are formed for predicting players’ actions. These include a shallow LSTM network with one hidden layer and 64 hidden neurons; a deep LSTM network with 4 hidden layers and 21 hidden neurons in each hidden layer; a Grid LSTM network with 4 hidden layers and 21 hidden neurons in each hidden layer; a recurrent highway network (RHN) with 3 hidden layers, 3 steps recurrence depth, and 21 hidden neurons per hidden layer (as shown in Figure 4.4); as well as a convolutional neural network (CNN) with 1 convolution layer using kernel size 1×21, convolution step size of 1, and 2 time-frame inputs. For all the following neural network-based player action and outcome prediction experiments, Adam (Kingma & Ba, 2015) is employed for NN optimization. A dropout rate of 0.1 is adopted for all the models. The number of training epochs in each round of cross-validation is determined using a separate validation set, which is later merged back into the training set for the final evaluation. All the model size related hyperparameters are tuned using random search.
According to results demonstrated in Table 6.2, using evaluations from five-fold cross-validation, the Friedman statistical test finds significant differences in player action prediction accuracy rates, $\chi^2(5)=25.0$, $p<0.001$, and macro-average F1 scores, $\chi^2(5)=23.6$, $p<0.001$, across the six models. The deep RHN model outperforms all the other action prediction models, including the shallow RNN-based state-of-the-art (shallow LSTM) (Wang & Rowe et al., 2017) by large margins, with the Wilcoxon post-hoc analysis yielding a $p$ value of 0.043 for each pairwise comparison on both prediction accuracy and macro-average F1 score metrics. The fact that all three deep recurrent NN models (RHN, deep LSTM, and Grid LSTM) outperform the shallow LSTM model on both metrics with $p$ values of 0.043 under pairwise Wilcoxon tests suggests that depth in recurrent NNs is an important factor in player action predictors. Further, all three deep recurrent NNs outperform the CNN model with $p$ values of 0.043 from pairwise Wilcoxon tests, which indicates that deep recurrent NNs provide an advantage over CNNs in player action prediction. A possible explanation may be that the exact gameplay history sequence provides considerably more information, which contributes to accurate player action prediction. To investigate the effects of regularization technique on training the deep RHN model, we train other RHNs either without dropout or only adopting a normal dropout layer on the output of the RHN model. In

<table>
<thead>
<tr>
<th>Model</th>
<th>Pred. Acc.</th>
<th>Macro F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLP</td>
<td>0.3248</td>
<td>0.1698</td>
</tr>
<tr>
<td>Shallow LSTM</td>
<td>0.3304</td>
<td>0.2361</td>
</tr>
<tr>
<td>Deep LSTM</td>
<td>0.4306</td>
<td>0.3108</td>
</tr>
<tr>
<td>Grid LSTM</td>
<td>0.4126</td>
<td>0.3065</td>
</tr>
<tr>
<td>RHN</td>
<td>0.4435</td>
<td>0.3218</td>
</tr>
<tr>
<td>CNN</td>
<td>0.3605</td>
<td>0.2655</td>
</tr>
</tbody>
</table>

Table 6.2: Player action prediction performance of deep learning models.
these cases, the action prediction accuracy rates drop from 0.4435 to 0.4112 and 0.4137, respectively.

For player outcome prediction, similar experiments and comparisons are conducted. First, the same set of non-NN baseline models are assessed for the player outcome prediction task. Model hyperparameters are also tuned using grid search, with the difference that the error penalization hyperparameter for the SVM model is set to 1, the random forest player outcome predictor’s number of estimators is set to 5, and the gradient boosting player outcome predictor’s number of estimators is set to 10.

Table 6.3: Player outcome prediction performance of non-deep learning models.

<table>
<thead>
<tr>
<th></th>
<th>Logistic</th>
<th>SVM</th>
<th>RF</th>
<th>GB</th>
<th>Naïve Bayes</th>
<th>MLP-Mul</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pred. Acc.</td>
<td>0.5673</td>
<td>0.5422</td>
<td>0.5623</td>
<td><strong>0.5721</strong></td>
<td>0.5596</td>
<td>0.5622</td>
</tr>
<tr>
<td>Macro F1</td>
<td><strong>0.5712</strong></td>
<td>0.5386</td>
<td>0.5616</td>
<td>0.5700</td>
<td>0.5569</td>
<td>0.5459</td>
</tr>
</tbody>
</table>

In Table 6.3, the six non-deep neural network baseline models’ performance on player outcome prediction are listed. In the player outcome prediction results, it can be seen that logistic regression and gradient boosting are the two best performing models based on the two metrics. Gradient boosting achieves the highest averaged prediction accuracy of 0.5721 over the five-fold cross validation, although there is no statistical difference on prediction accuracy between gradient boosting and logistic regression under two-tailed paired t-test ($t=0.2098$, $p=0.8390$). Logistic regression achieves the highest macro-average F1 score of 0.5712 over the five-fold cross validation. There is also no statistical difference between the performance of gradient boosting and logistic regression under two-tailed paired t-test ($t=0.0479$, $p=0.9630$) on macro-average F1 score.
Experiments are then conducted with deep neural network-based models as well. Several NN models are formed for predicting players’ outcomes. These include a shallow LSTM network with one hidden layer, 64 hidden neurons and multi-task output structure predicting player action type and player outcome type at the same time (labeled as LSTM-Mul in Table 6.4). A 1-layer RHN with recurrence depth of 5 is a deep recurrent model. A 2-layer CNN model with mixture of experts (using 23 experts) and multi-task regularization (labeled as CNN in Table 6.4, outputting player outcome type and player outcome score together) is implemented as shown in Figure 4.5. CNN-noME is the same network as the CNN model but without mixture-of-experts regularization, and CNN-noMul eliminates the multi-task output module from the CNN model. All CNN-based outcome predictors have convolution layers of size 21 with convolution kernels of size 1×21, convolution step size of 1, and 2 time-frame inputs. All these models use Adam as optimizer and adopt the dropout rate of 0.1.

<table>
<thead>
<tr>
<th></th>
<th>GB</th>
<th>LSTM-Mul</th>
<th>RHN</th>
<th>CNN</th>
<th>CNN-noME</th>
<th>CNN-noMul</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pred. Acc.</td>
<td>0.5721</td>
<td>0.5871</td>
<td>0.5997</td>
<td>0.6096</td>
<td>0.5772</td>
<td>0.5971</td>
</tr>
<tr>
<td>Macro F1</td>
<td>0.5700</td>
<td>0.5909</td>
<td>0.5997</td>
<td>0.6121</td>
<td>0.5798</td>
<td>0.5974</td>
</tr>
</tbody>
</table>

Table 6.4: Player outcome prediction performance of deep learning models.

CNN models outperform all the other models on both metrics of player outcome prediction accuracy and macro-average F1 score with five-fold cross-validation (Table 6.4). The Friedman test indicates that there are no statistically significant differences between these models in the outcome prediction accuracy ($\chi^2(5)=9.0$, $p=0.110$) and macro-average F1 score ($\chi^2(5)=7.8$, $p=0.167$). However, we find that CNN models outperform the gradient boosting baseline in all the five rounds of evaluations in cross-validation on both metrics,
which is not achieved by the LSTM-Mul or RHN models. This observation indicates that in contrast to player action prediction, keeping track of the exact sequence of gameplay history may not be necessary in player outcome prediction, which allows CNN models to perform better than recurrent neural networks. The result that CNN models achieve higher average performance over CNN-noME and CNN-noMul suggests that both mixture-of-experts and multi-task learning techniques have good regularization effects on training CNN-based player outcome predictors.

6.2 Reinforcement Learning-Based Interactive Narrative Planner Evaluation

Building on the results described in Section 6.1, we train high-fidelity deep RHN-based and deep CNN-based simulated player models to evaluate the effectiveness and generalizability of narrative adaptation policies derived by RL methods. Three experimental conditions are employed. In Condition 100-100, RL narrative planners are trained using the full CRYSTAL ISLAND corpus-based simulated player model and then evaluated on the same simulated player model. In Condition 50-100, the CRYSTAL ISLAND corpus is randomly divided into two halves, and a simulated player model is trained on each half. Then two RL narrative planners are trained using each half corpus-based simulated player model and evaluated on the full corpus-based simulated player model. In Condition 50-50, a two-fold cross-validation-like evaluation is conducted, in which RL narrative planners are trained using a training set-based simulated player model and evaluated on a test set-based simulated player model. In this way, the evaluation narrative planning environment in Conditions 100-100, 50-100 and 50-50 becomes increasingly unfamiliar to the trained narrative planners. For each narrative interaction step, a reward of 1 is assigned only at the conclusion of the interactive narrative if the simulated player reaches a high NLG outcome; otherwise, the reward is
always 0. All evaluations are conducted by allowing the derived narrative planners to interact with high-fidelity simulated player models for 5,000 episodes. Policies are evaluated by policy value, which is the average score the narrative planner receives for each narrative adaptation episode, ranging in [0,1].

For each RL method in each condition, three policies are generated with different training steps after training converges, which is implemented as following: during RL agent training, after each certain episodes of interaction and learning, the partially trained RL agent will interact with the simulated player model in an evaluation mode (optimization is paused at this period of time) for a certain number of episodes, so the value for this currently achieved policy can be roughly estimated through this quick evaluation interaction. This is a common technique to demonstrate how the RL agent training goes. Specifically, for the training process of an RL-based interactive narrative planner, this quick evaluation of a partially trained policy is very hard to be accurate and reliable, majorly because (simulated) human players’ behavior can be highly stochastic. This means either the RL agent training could not be easily converge if not properly controlled, or the RL function approximator (for the value function or policy function) would converge within a small space of functions instead of a specific RL function or policy. This property raises the question of when to stop the RL agent training to get the optimal interactive narrative planning policy. The method adopted in this dissertation is that, after observing the estimated RL policy value has converged in a range, we save the partially trained RL agent periodically during training and use the 3 policies with the highest quick evaluation values during the training period. This reflects if a well-trained RL-based interactive narrative planner can be reliably obtained.
In the first set of experiments, RL methods’ exploratory strategies, especially their effect on the policies’ generalizability is investigated. The asynchronous Q-network (AQN), asynchronous advantage actor-critic (A3C) method, and proximal policy optimization (PPO) method are employed. For A3C and PPO methods, a stochastic policy using the direct output of the methods and a greedy policy, which always adopts the action with the largest probability from $\pi(a|s; \theta^*)$, are both evaluated. The $\varepsilon$ value in $\varepsilon$-greedy of AQN is set to decay from 1 to 0.01 linearly in 75% of the training steps, which makes AQN operate in an exploration mode initially and in an exploitation mode in the later stages of training. Because of PPO’s optimization, it also takes on an exploitation mode more prominently than A3C. A3C is the most exploratory RL method in this experiment. For all AQN, A3C and PPO models utilized in the experiment, we adopted a 2-layer convolutional network-based structure with residual connections. Input into the deep RL network contains observation features across three timesteps. The convolution kernel has size of $1 \times 21$, and convolution step size of 1. Optimization of these deep RL networks are done with Adam.

<table>
<thead>
<tr>
<th>Cond.</th>
<th>Random</th>
<th>AQN</th>
<th>A3C</th>
<th>A3C-Greedy PPO</th>
<th>PPO-Greedy</th>
</tr>
</thead>
<tbody>
<tr>
<td>100-100</td>
<td>0.5531</td>
<td><strong>0.6255</strong></td>
<td>0.6064</td>
<td>0.6031</td>
<td>0.6061</td>
</tr>
<tr>
<td>50-100</td>
<td>0.5531</td>
<td><strong>0.5864</strong></td>
<td>0.5769</td>
<td>0.5802</td>
<td>0.5653</td>
</tr>
<tr>
<td>50-50</td>
<td>0.5315</td>
<td>0.5038</td>
<td>0.5510</td>
<td><strong>0.5516</strong></td>
<td>0.4756</td>
</tr>
</tbody>
</table>

It is found that the exploratory strategy of RL methods heavily affects the effectiveness, and particularly the generalizability, of interactive narrative planners (Table 6.5). When the evaluation environment is the same as the training environment (Condition
100-100), all RL methods derive effective narrative planning polices, and the more exploitative methods (AQN and PPO) are usually superior to the more exploratory method (A3C). These behaviors are what one would expect because exploitation helps the narrative planner to converge to (local) optimal policies easier. When the evaluation environment becomes more challenging and more unfamiliar to the RL narrative planners (Conditions 50-100 and 50-50), their performance drops with the degradation speed reflecting their degree of exploration. For less exploratory methods (AQN and PPO), performance drops faster, and they even derive policies worse than random in Condition 50-50. Meanwhile, the more exploratory method (A3C) is able to generate effective narrative planning polices (better than random policy), even when the evaluation simulated player model is quite different from the training simulated player model. These results indicate that the greater emphasis on exploration helps A3C avoid relying too much (and too early) on the highly noisy early narrative adaptation experiences, and A3C’s preference for exploring more thoroughly in uncertain environments boosts its ability to derive more generalizable narrative adaptation policies for unfamiliar players.

Table 6.6: Interactive narrative planning policy evaluation – value function-based RL methods.

<table>
<thead>
<tr>
<th>Cond.</th>
<th>AQN</th>
<th>Duel AQN</th>
<th>Recurrent AQN</th>
<th>DQN</th>
<th>Linear Q-Stabilized</th>
<th>Linear Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>100-100</td>
<td>0.6255</td>
<td>0.5923</td>
<td><strong>0.6505</strong></td>
<td>0.6078</td>
<td>0.6348</td>
<td>0.5798</td>
</tr>
<tr>
<td>50-100</td>
<td>0.5864</td>
<td>0.5606</td>
<td><strong>0.5864</strong></td>
<td>0.5840</td>
<td>0.5861</td>
<td>0.5744</td>
</tr>
<tr>
<td>50-50</td>
<td>0.5038</td>
<td>0.4905</td>
<td>0.4971</td>
<td><strong>0.5235</strong></td>
<td>0.5114</td>
<td>0.5006</td>
</tr>
</tbody>
</table>

In the second set of experiment, more value function-based RL methods are investigated and compared. As covered in Section 5, besides the AQN method adopted in the
prior set of experiment, other value function-based RL methods including Duel-AQN, Recurrent AQN, Deep Q-network with experience replay (DQN), a linear Q-learning model with asynchronous gradient descent and fixed target function stabilization (Linear Q-Stabilized), and a normal Q-learning model with a linear function approximator but no stabilization technique (Linear Q) are studied. Results are shown in Table 6.6.

Results from Table 6.6 offer clues about performance of different value function-based RL methods. Multiple interesting conclusions can be made from this set of the results, together with the results already illustrated in Table 6.5. First, the conclusion about the correlation between RL method’s exploratory strategy and the generalizability of the derived policy can be further strengthened by integrating all the results from Table 6.5 and Table 6.6. For all the methods listed in Table 6.6, they utilize the same $\epsilon$-greedy exploratory strategy during training. And the decaying schedule for the $\epsilon$ parameter is also the same for all these methods. So, all value function-based RL methods (AQN, Duel AQN, Recurrent AQN, DQN, Linear Q-Stabilized, and Linear Q in Table 6.5 and 6.6) that are tested in the experiment have the same exploratory strategy.

Interestingly, all these value function-based RL methods have the same trend of their performance in the three experiment conditions (condition 100-100, 50-100, 50-50) compared to the other two actor-critic methods (A3C and PPO) with different exploratory strategies. Because of the adopted $\epsilon$-greedy strategy, which prefers exploration in early stage of training and heavy exploitation in later stage of training, all value function-based RL methods can generate effective interactive narrative planning policies (better than random policy evaluated in policy value) in condition 100-100, 50-100, and struggles to generate effective policies in condition 50-50 when evaluation environment is very different from
training environment. This enhances the conclusion that was made previously in this section that a more exploitative (i.e., less exploratory) RL method is good at learning an effective policy in an less changing interaction environment, but not good at deriving an effective policy in a more changing interaction environment, probably because the agent tries to rely on previously learned experience too much and too early, which prohibits the improvement of its generalizability.

Comparing the performance of AQN and DQN methods in Table 6.6, it can be seen how effective different deep RL stabilization techniques are. The only difference of the AQN and DQN methods in this experiment is that AQN utilizes asynchronous gradient descent stabilization method, and DQN utilizes experience replay method (Section 5.2.1). At the same time, both AQN and DQN use fixed target to further improve the training stabilization. From the results it can be found that both asynchronous gradient descent stabilization and experience replay stabilization work effectively in training RL-based interactive narrative planners. Although in the experiment result, AQN generates higher policy value than DQN in condition 100-100, extra caution is needed to make the conclusion about which stabilization technique is better in the interactive narrative planning scenario. This is because there are many hyperparameters to tune both of these stabilization methods. For example, the number of parallel threads used in asynchronous gradient descent, the size of the experience replay database, and the size of the minibatch to feed into the RL agent training in experience replay are all possible to have effects on their performance. Due to the limited time to conduct all the studies, in this dissertation these hyperparameters have not been thoroughly tuned. So, the general idea to be concluded from this pair of comparison is that both types of deep RL stabilization techniques work for RL-based interactive narrative planner training. Also, when
the asynchronous gradient descent method was proposed (Mnih et al., 2016), it is their major contribution that this method is fairly faster than experience replay in training an effective deep RL agent, instead of training a finally better performing deep RL agent. In this experiment, we do find asynchronous gradient descent is faster than experience replay in training an RL agent, especially if GPU-based parallelization is not adopted with experience replay.

The comparison of the AQN, Recurrent AQN, and Linear Q-Stabilize methods reveals for effects of different neural network structures in effectively encoding the interaction history in state representation. AQN, as introduced above, utilized a two-layer CNN network to encode 3 timesteps of observation in state representation. Recurrent AQN uses a one-layer fully connected network to encode one timestep observation for each state representation. Linear Q-stabilize method uses a linear function approximator with stacked 3 timesteps of observation as RL agent state representation. From the results in Table 6.6, it can be seen that the Recurrent AQN performs best in condition 100-100 and 50-100. Its performance is especially better than the other two methods in condition 100-100, probably because the recurrent layer is especially good at tracking interaction sequences in learning an effective RL policy. Interesting, it can also be found that the Recurrent AQN’s performance degrades fastest from condition 100-100, to 50-100 and 50-50, compared against the other two methods. Possibly, the reason is that the recurrent layer learns a specific interaction sequence pattern, which might not be very useful and reliable in a changed interaction environment in conditions 50-100 and 50-50. Interesting, from the comparison it can be seen that CNN modules do not have advantage over linear function approximators in RL-based interactive narrative planner training.
From the results of Duel AQN in Table 6.6, it can be seen that adding the dueling module in RL-based interactive narrative planning agent does not help its performance in all the three experiment conditions. While the exact reason of the failure of the Duel AQN is hard to analyze, one potential explanation could be related to the domain property of interaction narrative planning. In (Wang, Schaul, Hessel, Van Hasselt, Lanctot, & De Freitas, 2016), the authors claimed that in their experiment testbed—Atari games, it was found that the advantage values of actions for a given state were usually very small relative to the magnitude of the Q values, which indicates that for a large portion of states, the goodness of being in these states is fairly more important than specific actions to take under these states. In this setting, maintaining an estimation of state value across actions is more meaningful and helpful in learning effective RL policies. However, this property might not be true in interactive narrative planning. It is understandable that for a large portion of states in interactive narratives, it is hard to tell if these states are either “good” or “bad”. Unfolding of a narrative usually follows a generally similar way, which makes more and more details and contents available to the players, and correspondingly change the interaction state representation. And it might be the case that for a lot of narrative interaction states, difference of performing distinct actions is more obvious than it is in testbeds like Atari games.

To study the effectiveness and the necessity of utilizing deep RL methods in interactive narrative planning, compared to non-deep RL methods, we demonstrate the comparison between the Linear Q-Stabilized and Linear Q methods in Table 6.6. It is necessary to clarify that the study of deep RL does not only focus on integrating various neural network-based function approximators into the RL framework. The most recent trend
of deep RL research started by solving the problem of neural network-based function approximators in RL make the training hard to converge (Mnih et al., 2015). For this reason, to investigate whether the cutting-edge deep RL techniques really help the training of RL-based interactive narrative planners, I compare the method of Linear Q-Stabilized, which is Q-learning with linear function approximator utilizing asynchronous gradient descent and fixed target function stabilizers, as well as the Linear Q method, which is a normal Q-learning method with linear function approximator without any stabilization techniques used in deep RL implementation. From results in Table 6.6, it can be seen that in all three experiment conditions, the Linear Q-Stabilized method outperforms the Linear Q method. As a further evidence of discussion, I should say that the Linear Q method has no problem of converging during training in our experiment. However, it looks like extra stabilization techniques adopted in even the linear Q-learning method helps to derived more effective interactive narrative planners.

For all the results demonstrated in Table 6.5 and 6.6, the largest coefficient of variation of any item in the two tables is 0.061, which is from the three sampled policies for each method under each experiment condition as explained above. This is an indication that reliable, converged policies can be derived using any of these RL methods that have been investigated in this dissertation.
CHAPTER 7

DISCUSSION

7.1 Constructing High-Fidelity Simulated Player Models

In this dissertation, a statistical predictive simulated player model was constructed and evaluated. The simulated player model is constructed with two modules, which separately predict player’s gameplay actions and outcomes in an interactive narrative-centered game. Both player gameplay action prediction and player outcome prediction problems are cast as classification problem. Player’s gameplay history, player’s traits and interactive narrative planner’s past decisions are utilized to form a feature set to represent the gameplay interaction process in simulated player models. Two evaluation metrics are employed to evaluate the performance of the simulated player models using different machine learning methods on both tasks. The simulated player model studied in this dissertation not only serves as player model to describe player’s gameplay behavior, but also provides a testbed for adaptable interactive narrative planning study that is also conducted in this dissertation research.

Modeling player’s behavior in interactive narrative-centered games is an essential and hard problem. Building a statistical predictive model helps researchers to gain better understanding of player’s interaction process, as well as interaction results. For player action prediction, one major challenge is that probability distribution of actions is imbalanced. This is the major reason why macro-averaged F1 score is one of the metrics utilized in this work. We have not utilized special strategies to deal with the problem in our player modeling work because the costs of making an incorrect prediction of player’s action across multiple classes do not vary in this specific game domain. Also, one can expect that the future gameplay
interaction would demonstrate similar patterns of player action probability distribution for other populations of players. However, extra care is necessary when dealing with special player actions in the game domain. One example is the game-ending action. In case game-ending action is one action that is modeled by the player simulated model, and the simulated game interaction would be ended only if the simulated player adopts the game-ending action, it is possible to introduce the problem that the simulated gameplay will never terminate because the simulated player would never adopt the game-ending action according to the action prediction results. There are multiple potential solutions to deal with this problem. One solution that is adopted in this work is when certain amount of player gameplay actions are simulated and generated (indicating the simulated player has played the game for a certain amount of time), the probability of emitting the game-ending action would be significantly increased to make sure the game ends after a long-term game play. This setting is designed due to the fact that in the CRYSTAL ISLAND study, human players were given at most 55 minutes to play the game. Even if the player cannot finish the gameplay, the study and the gameplay would end. In cases where certain player gameplay actions are more important to predict correctly (certain actions are costlier for incorrect prediction), researchers can consider using techniques like oversampling to deal with the imbalanced corpus problem.

From the experiment results demonstrated in this dissertation, it can be seen that the task of modeling player’s gameplay outcome is especially hard in simulated player modeling. Even though the problem is simplified into a binary classification problem, it can be seen from the results (Table 6.3, 6.4) that this problem is even harder than player action prediction, even though player action prediction is a 15-class classification problem (Section
After some quick empirical analysis of the results, we find that during evaluation, the predictor would sometimes make very high confidence (upon the output of the softmax classifier) to make an incorrect prediction of player’s outcome. A potential reason is that the set of features utilized in this player outcome predictor is not accurate enough to reflect player’s outcome patterns. With the current feature representation, it is possible to see multiple players going through similar interaction process but achieve different interaction outcomes. To solve this problem, more careful feature engineering should be a potential solution. Min and Mott et al. (2017) had conducted experiment of extending player modeling using multimodal data, which can be a promising direction to effectively extend this type of work. Also, it is found that in player modeling problems, the prediction of lower level concepts that are more related to the game rule can be easier than the prediction of higher level concepts that are not quite related to the game rule. This can be seen from the fact that player’s gameplay actions (Wang, Rowe, Min, Mott & Lester, 2017a; Wang et al., 2018), short-term interaction goals (Min et al., 2017) are easier to predict than player’s learning outcomes (Wang, Rowe, Min, Mott & Lester, 2017a; Wang et al., 2018).

Another thing worthy to discuss in simulated player modeling is the machine learning model selection. From the results in the Section 6 (Table 6.1-6.4), some general and interesting conclusion can be drawn. Besides the naïve Bayes model, which performs poorly on both player action and outcome predictions, probably because of the extremely strong independent features assumption, all other non deep NN-based machine learning models’ performance do not differ much. It is the case when RNN models or CNN models are utilized, the performance of player action and outcome predictions could be obviously improved. According to the results, we would argue that because of the sequence modeling
ability of deep RNN and CNN models, the performance of deep NN-based simulated players can be significantly improved over baselines. It is true that there exist other models (e.g., hidden Markov model) which are able to deal with sequence data, it is not the focus of evaluating all of these models in this dissertation. Compared to probabilistic graphical models (e.g., HMM), an obvious advantage of RNN and CNN models is that much less feature engineering work is necessary. However, probabilistic graphical models also take the advantage of easiness in interpretation.

7.2 Deriving Interactive Narrative Planners

A significant contribution of this dissertation is the introduction of statistical predictive simulated player models that are utilized in interactive narrative planners’ training and evaluation. Although statistical predictive simulated players are not broadly utilized either in the interactive narrative community, or in the educational game community, it has advantage over other training and evaluation setups from perspectives of both feasibility and cost. A statistical predictive simulated player model is learned from human player generating data, so ideally it should reflect human players’ interaction pattern. This is an advantage over assumption-based simulated players, with which the performance of simulated players is mainly determined by human designed rules. Also, statistical simulated player model is feasible considering it is easier to control the stochasticity of the simulated player’s behavior in the process of sampling simulated player’s actions and outcomes. This property is beneficial because human player’s behavior is highly stochastic. The ease of controlling the stochasticity in simulated player’s behavior sets up a more realistic experiment environment. Simulated player model also allows creating different simulated player pattern reflecting different populations’ gameplay preferences in training and evaluation, which permits the
study of the generalizability of the derived interactive narrative planning policy. This is also the reason of creating three different experiment conditions in the experiment setup (Section 6.2). Simulated players offer an almost zero-cost test environment before the interactive narrative planner is deployed back to the system and interact with human players. So, this is a safe and inexpensive way for thoroughly studying the effectiveness of interactive narrative planners in games.

In addition to the conclusions discussed in Section 6.2, it is interesting to investigate the performance of reinforcement learning-based interactive narrative planner’s performance from a high-level perspective. The most noticeable conclusion in the experiment results (Section 6.2) is that the experiment condition (which determines the similarity between training environment and evaluation environment) hugely affects the performance of the RL-based interactive narrative planner. It can be seen when the training and evaluation environment is the same, RL-based interactive narrative planner is able to improve the policy value by almost 10%, which indicates about 10% more players could have their interaction outcomes improved from low NLG class to high NLG class. This is an impressive result, considering the simulated player model generates simulated player outcomes in a highly stochastic way. However, when training and evaluation environments are different, the performance of the RL-based interactive narrative planners degrades dramatically, even would reach to the situation that the derived planner performs worse than random planner. This conclusion serves as a very important indication that data-driven interactive narrative planner should be cautiously deployed to interact with new player populations. Even though effective interactive narrative planners can be trained, it does not have to be effective when new players use them. Of course, it does not mean data-driven interactive narrative planner
derivation is not reliable. But it does reveal the truth to researchers and designers of interactive narrative systems or game adaptation systems that enough human player interaction data should be collected and utilized in training data-driven interactive narrative planners or game content managers to improve the possibility that the derived agent would be generalizable. Also, finding meaningful features in describing this interaction is another way to improve the generalizability.

Another important contribution of this work in this dissertation is constructing a model-based reinforcement learning framework to derive interactive narrative planners. The process of training simulated player models can be seen as building a Markov decision process (MDP) for a reinforcement learning agent. Training player action predictors and player outcome predictors correspond to learning the state transition function and reward function in building up an MDP. From this perspective, the whole work in this dissertation can be understood as building an MDP from human player generating corpus and then derive (locally) optimal policies from the MDP model. In this case, the major difference of this work from the prior model-based reinforcement learning works for interactive narrative planning (Rowe et al., 2014, Rowe & Lester, 2015) is that the framework from this dissertation permits a broader range of model-free reinforcement learning methods (e.g., Q-learning, actor-critic reinforcement learning) to be adopted, which bring more feasibility in reinforcement learning method selection.

7.3 Limitations

This dissertation has presented a set of experiments in simulated player modeling. The conclusions presented in Section 6, Section 7.1, and Section 7.2 should be considered in light of the limitations of this work.
First, it will be important to determine how effective the derived interactive narrative planners are with human players. An important question to consider before running a human player involved study is determining how many human players are sufficient to yield accurate estimates of the quality of each derived policy? The simulated player model would offer some empirical evidence, but more rigorous estimation is also necessary to get a reliable study result. This offers an important direction for future work.

Second, it can be seen from the experimental results that human players’ interaction process cannot be very accurately represented using the current set of features. This can be attributed to the fact that the player outcome prediction accuracy is relatively low, which indicates that high performing players and low performing players cannot be effectively distinguished using the current features, even though this set of features is broadly used in describing player’s gameplay interactions. It is effective at predicting more game-rule related features (e.g., player actions, player short-term goals), but not sufficiently effective at predicting players’ outcomes. It is possible that adding multimodal data could improve the performance of the predictors.

Third, to derive a generalizable interactive narrative planning policy that is able to be deployed in the game for future use with new players, it may be the case that the current dataset is not large enough. Thus, collecting a larger corpus might be necessary to derive reliable and generalizable interactive narrative planners.
CHAPTER 8
CONCLUSIONS AND FUTURE WORK

Interactive narrative adaptation is a promising and challenging task in open-world educational games. The capability of interactive narrative planners to personalize players’ narrative interaction experiences would significantly affect players’ learning outcome. In this dissertation, a data-driven interactive narrative adaptation framework has been proposed. This framework is constructed from two components: 1) a statistical predictive simulated player model that learns to imitate human players’ gameplay behavior and estimate human players’ learning outcome, and 2) a reinforcement learning-based interactive narrative planner learns to tailor narrative content to fit individual player’s needs for improving their learning. Specifically, neural network models have been broadly investigated for both components of the data-driven interactive narrative adaptation framework. In the experimental evaluations, the proposed neural network models have achieved high performance in both simulated player modeling and reinforcement learning-based adaptive interactive narrative planning. Leveraging the experiments with the CRYSTAL ISLAND interactive narrative, this dissertation has explored the utilization of deep recurrent neural networks and deep convolutional neural networks on sequence pattern extraction in simulated player modeling, as well as investigating the effects of deep reinforcement learning methods in adaptable and generalizable interactive narrative planner derivation.

8.1 Research Questions Revisited

With the experimental results from Section 6 and discussion from Section 7, we now revisit the research questions posed in Section 1.3. This serves as an examination of whether the
proposed methods for data-driven simulated player modeling and interactive narrative planning problems are effective and generalizable.

• **RQ 1: Is deep learning an effective choice to construct data-driven simulated player models?**

Yes. Empirical evidence from our experiment with the CRYSTAL ISLAND dataset demonstrates the deep learning models, especially deep RNN and deep CNN models are more effective in constructing data-driven simulated player models, measured on metrics of prediction accuracy and macro-average F1 score, on both tasks of player gameplay action prediction and player outcome prediction.

• **RQ 1.1: Can deep learning outperform commonly utilized machine learning methods (baselines) in both tasks of player action prediction and player outcome prediction?**

Yes. From the experimental results (Section 6) it is found that certain neural network structures, including RNNs and CNNs, are especially effective in dealing with sequence data in simulated player modeling and outperform non-neural network models by a large margin. More specifically, it is found that the deep recurrent highway networks and deep convolutional networks with residual connections are the best performing models in our experiment for player action prediction and player outcome prediction tasks, respectively.

• **RQ 1.2: Can deep learning offer proper model to deal with sequential player—narrative planner interaction patterns in simulated player modeling?**

Yes. It is found from the experimental results that the major reason that deep learning can outperform other models in simulated player modeling is that
certain neural network models are especially good at learning patterns from sequence data. By comparing the performance of multilayer perceptron and some non-neural network models, like logistic regression and gradient boosting, it can be seen that multilayer perceptron does not have obvious advantage, especially for the task of player outcome prediction, in which case the other non-neural network-based models could achieve better performance. However, with proper neural network structures like recurrent layers or convolution layers added in the neural network models, their performance on both player action prediction and player outcome prediction can be improved.

• **RQ 1.3: Does deep neural network models outperform shallow models in constructing simulated player models?**

Yes. From the experimental results, it is found that depth of neural network models in simulated player modeling is one of the most important factors in determining the effectiveness of player behavior predictors. This phenomenon is significant in player action prediction, in which a 3 hidden layer deep recurrent highway network with a recurrence depth of 3 is the best preforming model. Also, all the 3 deep RNN models outperform the shallow LSTM model, which is also a good indicator that depth is an important feature in building effective simulated player models. It is also noticeable that increasing neural network depth by stacking hidden layers might not be the most effective way. Besides the adoption of proper regularization methods, techniques like adopting highway units and residual connections are also critical.
RQ 1.4: What regularization methods are necessary to improve deep learning methods’ generalization performance in simulated player modeling?

It is found that regularization is especially important when deep neural network models are utilized. In the experiment it can be seen that the dropout method, mixture of experts, and multi-task learning are specifically helpful in simulated player modeling. In player action prediction, variational inference-based dropout boosts the performance of recurrent highway networks over regular dropout technique, or the case dropout is not utilized. For player outcome prediction, multi-task learning and mixture-of-experts techniques both improve the performance of convolutional network-based models.

RQ 2: Can deep reinforcement learning derive effective interactive narrative planners in narrative-centered educational games to personalize narratives to individual players?

Yes. From the experimental results, it can be seen that deep reinforcement learning can derive effective interactive narrative planning policies which perform better than a uniform random yet rational policy for interactive narrative adaptation. In the three experiment conditions investigated, all deep and non-deep reinforcement learning methods can derive effective policies in conditions when training simulated players are the same or similar to evaluating simulated players (condition 100-100, 50-100 in Section 6). In the condition where training environment is very different from evaluating environment (condition 50-50 in Section 6), only some deep reinforcement learning method (A3C) can derive effective policies (better than random policy), while other reinforcement learning methods do not outperform the random policy.
• **RQ 2.1: Is deep reinforcement learning a tractable method in training interactive narrative planners?**

Yes. One apparent difficulty in studying and utilizing deep reinforcement learning methods is to converge the training process of deep reinforcement learning agents, so effective polices can be generated. The most recent advancement of deep reinforcement learning techniques focuses on proposing new methods in stabilizing reinforcement learning agent training, which demonstrated good performance. These stabilization techniques are utilized in this dissertation research, and the results are very promising. All the stabilization techniques adopted in this study show good performance. Effective policies can be derived using deep reinforcement learning. Meanwhile, in the experiment, it is also seen that linear Q-learning’s performance can also be boosted by these stabilization techniques.

• **RQ 2.2: Can deep reinforcement learning outperform linear reinforcement learning method (baseline) in training adaptable interactive narrative planners?**

Generally, the answer is yes. A regular Q-learning model with linear function approximator and not adopting any stabilization techniques is tested as a baseline in the experiment. In experiment condition 100-100, all deep RL-based methods outperform this linear Q-learning baseline while the linear Q-learning baseline still beats the random policy. In condition 50-100, all value function-based deep RL methods except for the Duel AQN outperform the linear Q-learning baseline. As discussed in Section 6, the Duel AQN does not perform well in all three conditions in the value function-based RL family.
The PPO method does not outperform the linear Q-learning baseline in condition 50-100. The explanation can be that the exploratory strategy affects PPO’s performance in such condition. In condition 50-50, all value function-based RL methods cannot derive better than random policies, while A3C can still derive effective policies. So, generally, it can be concluded that deep RL has advantage over commonly used linear RL (Q-learning) method in interactive narrative planner derivation.

• **RQ 2.3: What are proper neural network structures to form a reinforcement learning-based interactive narrative planner?**

According to the experiment results, it is found that the proper neural network structures to construct a deep reinforcement learning model in interactive narrative planning should depend on the experiment conditions. It can be seen in condition 100-100 and 50-100, when the training environment is the same or similar to the evaluating environment, recurrent structure demonstrates advantage over convolutional structure and fully connected structure in constructing a neural network-based RL agent. This advantage degrades quickly with the increase of the difference between the training and evaluating environment. Interestingly, we have not seen advantage of convolutional neural networks over linear function approximators in constructing neural network-based RL agent for interactive narrative planning. All the above conclusions are based on comparison of results of value function RL-based interactive narrative planner derivation.
• RQ 2.4: How does deep reinforcement learning-based interactive narrative planner perform in interaction with unfamiliar player population (generalization performance)?

Deep reinforcement learning methods can derive generalizable interactive narrative planners. The generalizability of deep RL-based policies is correlated with the exploratory strategies of the deep RL methods. More exploratory RL methods can derive more generalizable policies; however, its lack of exploitation makes it less effective in learning policies for evaluation environments that are similar to the training environment.

8.2 Summary

In this dissertation, data-driven interactive narrative planning has been investigated. Using the educational interactive narrative testbed, CRYSTAL ISLAND, we have created statistical predictive simulated player modeling and reinforcement learning-based adaptable interactive narrative planner derivation frameworks. In particular, deep learning models in both aspects of the research have been explored and evaluated using a corpus of human gameplay data.

Experiments suggests the following findings. First, it has been demonstrated that neural network-based models have advantages over several non-neural network-based commonly utilized classification models in both tasks of player action prediction and player outcome prediction. Further analysis demonstrates that specific neural network structures, i.e., recurrent neural network module and convolutional neural network module, significantly enhance the model’s capability in learning player interaction patterns from sequence data, so that the simulated player model’s prediction performance can be significantly improved over non-neural network baselines. Second, it has been found that the depth of neural network
models and the regularization techniques significantly contribute to the performance of the simulated players. Also, it is important to note that specific techniques such as the utilization of highway units and residual connections have significant effects on deep neural networks for interactive narrative. Based on these findings, the analysis of player’s gameplay interaction behavior reveals that with the usage of the selected feature set, players’ interaction behavior is highly stochastic. As an important limitation, this finding suggests that interactive narrative designers consider utilizing multimodal data in feature representations, as well as collecting large datasets in studying human player’s interaction patterns.

Based on the high-fidelity simulated player model, reinforcement learning-based adaptable interactive narrative planning has been investigated. Multiple deep reinforcement learning methods and Q-learning baselines with linear function approximators are used in interactive narrative planner training. Three distinct experiment conditions were used, in which evaluating simulated players can behave the same, similar to, or very different from the training simulated players. In such a setting, the effectiveness of reinforcement learning-based interactive narrative planners, including their capacity to learn generalized policies into unfamiliar interaction environments can be tested. From the experimental results, two major conclusions can be drawn. First, reinforcement learning’s exploratory strategy is strongly correlated with the generalizability of the trained interactive narrative planning policy. More exploitative RL methods (value function-based RL methods with \(\varepsilon\)-greedy exploratory strategy) derives higher performing policies in experiment conditions where evaluating simulated players behave either the same or similar to the training simulated players. More exploratory methods (A3C), in experimental conditions where evaluating simulated players behave quite differently from training simulated players, is still able to derive effective
interactive narrative planning policies (better than random yet rational policy) while more exploitative methods do not yield effective policies in these challenging conditions.

Second, there are interesting contrasts between the effectiveness of different deep reinforcement learning models. One interesting finding is that recurrent Q-network outperforms other neural network-based RL models including convolutional neural network-based and fully connected neural network-based RL models, as well as linear Q-learning models in experiment conditions where the evaluating simulated players behave the same or similar to training simulated players. Also, recurrent Q-network’s performance degrades quickly in the experiment condition where evaluating simulated players behave very differently from training simulated players. This is understandable, considering recurrent networks learns patterns from exact interaction sequences, which might not be generalizable in different simulated player populations. In the experiments, using stabilization techniques that are broadly adopted in deep reinforcement learning is one of the most effective factors determining the performance of RL methods in interactive narrative planning. Linear Q-learning method is also improved by adopting stabilization techniques. This observation is consistent with our understanding that in highly stochastic interaction environments such as interactive narrative environments, stabilization of RL agent’s training is of great importance.

In sum, adaptive interaction narrative planning presents significant challenges, and data-driven methods offer a promising approach. Deep learning methods offer considerable promise for both simulated player modeling and interactive narrative planning. These can be used to develop a deeper understanding of gameplay interaction patterns, as well as to derive adaptable interactive narrative planners.
8.3 Future Work

Beyond the results and conclusions drawn from the prior discussions, several directions for future work should be explored. First, the current feature set representing player interactions offers gameplay and player traits features, which might be a helpful for gameplay-related predictions and adaptations, but is not sufficient for player outcome-related predictions and adaptations. To improve the performance of the framework on player outcome tasks, it can be valuable to further extend current work by utilizing multimodal features (e.g., player eye-gaze features) in player outcome-related tasks. It is expected that additional related features, perhaps in conjunction with larger datasets, can effectively boost the performance of both simulated player models and adaptable interactive narrative planners.

Second, exploring graphical models for sequential data offer an important direction for future work. Considering that one of the major advantages of neural network models is their ability to effectively learn interaction patterns from sequence data, it will be important to investigate how graphical models can contribute to modeling sequential gameplay data.

Third, in this dissertation, human players were not involved in the final round of interactive narrative planning evaluation. In the future, it will be important to conduct studies in which human players actually interact with induced interactive narrative planners, particularly with regard to adaptation. This approach can also shed light on the generalizability of policies for interactive narrative.
REFERENCES


Gervás, P. (2009). Computational Approaches to Story-telling and Creativity. *AI Magazine*, 1


Intelligence and Interactive Digital Entertainment, 99-104.


Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G.,


Wang, P., Rowe, J., Min, W., Mott, B., & Lester, J. (2017a). Interactive narrative
personalization with deep reinforcement learning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, 3852-3858.


Young, S. (2002). Talking to machines (statistically speaking). In *Seventh International Conference on Spoken Language Processing*. 


