MIN, WOOKHEE. Generalized Goal Recognition Framework for Open-World Digital Games. (Under the direction of Dr. James C. Lester.)

Recent years have seen a growing interest in player modeling for open-world digital games. With the objective of creating player-adaptive gameplay, a central problem of player modeling is goal recognition, which aims to recognize players’ intentions from observable gameplay behaviors. Player goal recognition offers the promise of enabling games to dynamically adjust challenge levels, perform procedural content generation, generate interactive narratives, and create believable non-player character interactions. However, player goal recognition in open-world digital games poses significant challenges because of the inherent complexity in players’ gameplay behaviors attributable to a vast number of possible paths to achieve goals. Devising effective goal recognition models that deal with highly noisy action sequences is key to the success of goal recognition in open-world digital games.

This dissertation presents GOALIE, a generalized goal recognition framework featuring goal recognition models and multidimensional model evaluation. This framework supports (1) devising goal recognition models using corpora of goals and sequences of player actions executed to achieve goals for a given open-world digital game and (2) performing multidimensional evaluations of induced goal recognition models in order to select the most reliable model that will eventually operate at run-time to enable player goal-directed game adaptations. We use GOALIE to investigate a set of machine learning techniques that effectively model sequential patterns and cyclical relationships that pervade open-world digital games. Specifically, we use GOALIE to evaluate two deep learning-based techniques, including long short-term memory networks (LSTMs) and $n$-gram encoded feedforward
neural networks pre-trained with stacked denoising autoencoders along with two probabilistic graphical model-based approaches, including linear-chain conditional random fields (CRFs) and Markov logic networks. The most reliable goal modeling approach is identified through GOALIE’s multidimensional evaluation framework with a specific focus on measuring models’ predictive accuracy and early prediction capacity.

Results show that GOALIE serves as a generalized goal recognition framework that is scalable to the two open-world educational games. Of the goal modeling approaches investigated, LSTMs featuring distributed action representation learning achieve both the highest predictive accuracy and n-early convergence rates across the two corpora, while CRFs achieve the best standardized convergence points across the two data corpora. Notably, in addition to the LSTM model’s superior performance, it also automatically extracts predictive features in the form of an end-to-end trainable deep learning model, and thus provides an effective, scalable, and reliable solution to player goal recognition for open-world digital games.
Generalized Goal Recognition Framework for Open-World Digital Games

by
Wookhee Min

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

2016

APPROVED BY:

Dr. James C. Lester
Committee Chair

Dr. Bradford W. Mott

Dr. Eric N. Wiebe

Dr. Min Chi
DEDICATION

Dedicated to my parents and wife who have shown trust, support, and love,

and my son who has brought joy to my life.
BIOGRAPHY

Wookhee Min was born in Seoul, South Korea. As a K-12 student in the public school system in Gangnam in Seoul, he developed a deep interest in mathematics. With his parents’ support, he was recognized with several awards in mathematics competitions. These achievements paved a way to join the engineering school of Yonsei University in Seoul in 2000.

Wookhee entered the Department of Computer Science at Yonsei University after his first year. Computer science was a perfect fit for his interests, and he became fascinated with challenging problems in algorithms, security, cryptography, and artificial intelligence. During his college years he served in the Republic of Korea Army as a member of Korean augmentation to the United States Army. He worked for Special Operations Theater Support Elements – Korea in Yongsan, South Korea. In 2002 he was honorably discharged as a sergeant and was awarded the Army Commendation Medal.

After graduating from Yonsei University, in 2006 Wookhee joined Samsung Electronics, where he continued to work on security and AI-centered projects. He implemented an affect-based natural language processing engine, which was a core component of a text-to-animation system. After four and half years at Samsung, in 2011 he decided to pursue graduate study in computer science at North Carolina State University. At NC State he studied advanced AI techniques and investigated a broad range of AI problems in user modeling, intelligent game-based learning environments, and natural language processing.
ACKNOWLEDGMENTS

I would like to sincerely thank my academic advisor, James Lester, for his insightful advice on investigating the areas that have underpinned my dissertation work, including intent recognition, natural language processing, intelligent tutoring systems, and deep learning, among others. His kind and encouraging words have always kept me motivated during my graduate career. I also want to thank Brad Mott. Brad has provided me with careful feedback for an enormously wide range of questions regarding development, research, and even decision-making in my personal life. I am also grateful for Eric Wiebe’s insightful advice, which has substantially broadened my views to learning and educational applications. It has been a great pleasure to work and write research papers together throughout the ENGAGE and AIM projects, which have occupied much of my graduate work. Finally, I am very grateful to have had Min Chi as a member of my dissertation committee. I thank her not only for technical feedback and encouraging comments, but also for her warm regard for my family.

I want to thank Jon Rowe for his careful review on the broad range of work we have collaborated on and for his family’s kindesses to me. I appreciate Rob Taylor’s help and support whenever I have encountered challenges in my development work. Andy Smith and Pengcheng Wang are my great academic colleagues as well as dearest friends at school, who have made my graduate life full of excitement and fun. I have very much enjoyed my work with my colleagues on the ENGAGE project, including Kristy Boyer, Phil Buffum, Kirby Culbertson, Megan Frankosky, and Fernando Rodríguez, and I am grateful to continuing
collaboration on the AIM project with Lydia Pezzullo, Jen Tsan, Alex Vail, Chi-Han Wang, and Joseph Wiggins. To my two former colleagues who graduated some time back but who still keep in close touch with me, Eun Young Ha and Seung Lee, I very much appreciate their kindness. I also want to thank three former, more recently graduated colleagues, Alok Baikadi, Joe Grafsgaard and Sam Leeman-Munk. It has been fortunate to get to know them and great pleasure to work with them. I also wish to thank Collin Lynch for his kindness to my family and me. My chats with him have always been uplifting and motivating. Also, I thank my colleagues in the Center for Educational Informatics and friends for their help and support over the course of my graduate career.

Finally, I want to thank my parents, parents-in-law, sisters, and brothers-in-law for their persistent encouragement and continuous love for me over this five-year journey, while I stay more than 20 hours away from them. With their trust and love, I was able to embark on a long journey to pursue a Ph.D. at NCSU. I especially thank my mother-in-law and parents who came to Raleigh and helped to care for my little one. With all of their help we are raising a delightful boy. Finally, I sincerely thank my lovely wife, Yeojin, for her support, selfless devotion, trust and love, and my dearest son, Ewan, who has brought an enormous amount of joy and happiness to my life.
# TABLE OF CONTENTS

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

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

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

1.1 Thesis Statement and Hypotheses........................................................................... 8

1.2 Contributions......................................................................................................... 13

1.3 Organization.......................................................................................................... 15

CHAPTER 2 RELATED WORK ...................................................................................... 17

2.1 Plan, Activity, and Intent Recognition.................................................................... 17

2.1.1 Plan Recognition............................................................................................... 18

2.1.2 Goal Recognition............................................................................................. 27

2.1.3 Activity Recognition....................................................................................... 32

2.2 Deep Learning...................................................................................................... 33

2.2.1 Overview of Deep Learning............................................................................ 33

2.2.2 Deep Feedforward Neural Networks.............................................................. 34

2.2.3 Recurrent Neural Networks........................................................................... 38

2.2.4 Deep Learning in Vision, Language and Speech ........................................... 44

2.3 User Modeling in Intelligent Game-Based Learning Environments....................... 46

CHAPTER 3 GOAL RECOGNITION-BASED OPEN-WORLD DIGITAL GAMES......... 51

3.1 An Overview of Goal Recognition-Enabled Digital Games................................... 51

3.2 Generalized Observable Action Learning for Intent Evaluation (GOALIE) ....... 55

3.2.1 GOALIE: Goal Recognition Model Training.................................................. 56

3.2.2 GOALIE: Goal Recognition Model Evaluation Metrics................................ 69

CHAPTER 4 GOAL RECOGNITION DATA CORPORAS............................................. 73

4.1 Crystal Island: Outbreak......................................................................................... 74

4.1.1 Backstory and Game Play.............................................................................. 74
4.1.2 Study Description .................................................................................................................. 76
4.1.3 Corpus .................................................................................................................................. 77
4.2 Crystal Island: Uncharted Discovery ......................................................................................... 79
4.2.1 Backstory and Game Play .................................................................................................... 80
4.2.2 Study Description .................................................................................................................. 82
4.2.3 Corpus .................................................................................................................................. 83

Chapter 5 Sequence Model-Based Goal Recognition ......................................................................... 87
5.1 Data Representation Methods: One-Hot Encoding and Distributed Representations .............. 88
5.2 Long Short-Term Memory Network-Based Goal Recognition .................................................... 91
5.2.1 LSTM-Based Goal Recognition Model Architecture ............................................................ 93
5.2.2 LSTM-Based Goal Recognition Model Configurations ......................................................... 97
5.3 Linear-Chain Conditional Random Field-Based Goal Recognition .............................................. 98

Chapter 6 Evaluation ................................................................................................................... 102
6.1 Results for Crystal Island: Outbreak ....................................................................................... 103
6.2 Results for Crystal Island: Uncharted Discovery .................................................................... 110

Chapter 7 Discussion .................................................................................................................. 117
7.1 Discussion on Empirical Results for Crystal Island: Outbreak ................................................. 117
7.2 Discussion on Empirical Results for Crystal Island: Uncharted Discovery ................................ 120
7.3 Limitations ............................................................................................................................... 123

Chapter 8 Conclusions and Future Work .................................................................................... 129
8.1 Hypotheses Revisited .............................................................................................................. 131
8.2 Future Work .............................................................................................................................. 134
8.3 Concluding Remarks ............................................................................................................... 139

References ..................................................................................................................................... 143
LIST OF TABLES

Table 4.1: OUTBREAK: Distributions of goals ...............................................................79
Table 4.2: UNCHARTED DISCOVERY: Quest descriptions ........................................83
Table 4.3: UNCHARTED DISCOVERY: Distributions of goals ......................................86
Table 6.1: Accuracy rates of goal recognition models for the OUTBREAK dataset ..........105
Table 6.2: Accuracy rates, convergence points, standardized convergence points,
        convergence rates, and 1-early convergence rates of the highest performing goal
        recognition models for the OUTBREAK dataset ..................................................106
Table 6.3: Wilcoxon signed rank statistical test results with respect to the accuracy rate for
        the OUTBREAK dataset ......................................................................................107
Table 6.4: Precision, recall, and F1 scores of the distributed LSTMs and CRFs for the
        OUTBREAK dataset ..........................................................................................109
Table 6.5: Accuracy rates of goal recognition models for the UNCHARTED DISCOVERY
        dataset ................................................................................................................112
Table 6.6: Accuracy rates, convergence points, standardized convergence points,
        convergence rates, and 1-early convergence rates of the highest performing goal
        recognition models for the UNCHARTED DISCOVERY dataset ..........................113
Table 6.7: Wilcoxon signed rank statistical test results with respect to the accuracy rate for
        the UNCHARTED DISCOVERY dataset ................................................................114
Table 6.8: Precision, recall, and F1 scores of the distributed LSTMs and CRFs for the
        UNCHARTED DISCOVERY dataset ....................................................................115
LIST OF FIGURES

Figure 2.1: A conceptual illustration of layer-wise stacked denoising autoencoders ..........37
Figure 2.2: Recurrent neural networks ........................................................................38
Figure 2.3: Different classification tasks using recurrent neural networks .................40
Figure 2.4: A conceptual illustration of an LSTM memory block that features three gating units and a memory cell ........................................................................42
Figure 2.5: Two examples of LSTM to solve a classification task illustrated in 
Figure 2.3C .............................................................................................................44
Figure 3.1: Open-world digital games enriched with the goal recognition feature ..........52
Figure 3.2: Bayesian network goal recognition model .................................................60
Figure 3.3: Graphical representation of the Markov logic network ................................63
Figure 3.4: N-gram encoded feedforward neural network ...........................................66
Figure 4.1: CRYSTAL ISLAND: OUTBREAK educational game ...............................75
Figure 4.2: Map of the CRYSTAL ISLAND: OUTBREAK research camp ...............76
Figure 4.3: CRYSTAL ISLAND: UNCHARTED DISCOVERY educational game .........80
Figure 4.4: Virtual tablet’s IslandPedia app and problem-solving app .......................82
Figure 5.1: (A) An illustration of LSTMs for goal recognition and (B) a distributed action representation generated from a discrete action representation ........................................94
Figure 5.2: A factor graph representation of linear-chain CRFs .................................99
Figure 5.3: (A) A linear-chain CRF for goal recognition and (B) action properties converted to one-hot vectors .........................................................................................100
CHAPTER 1
INTRODUCTION

Recent years have seen the emergence of digital games that feature open-world environments that offer players significant autonomy over the goals they pursue and the plans they use to achieve their goals (Squire, 2008). In contrast to linear games, which prescribe a particular sequence of gameplay objectives for players to accomplish, the non-linear goals, story plots, and non-deterministic paths to achieve in-game objectives that pervade open-world digital games promote human players’ engagement and support increased replayability. However, open-world digital games pose significant challenges for game designers. Open-world games’ emphasis on player autonomy is at odds with game designers’ focus on crafting coherent storylines and well-paced gameplay; it is difficult for game designers to craft compelling stories if they do not know, in advance, what actions the player is going to take next (Min, Mott, Rowe, Liu, & Lester, 2016; Riedl & Bulitko, 2013).

To address these challenges, player modeling, computational modeling of players’ multidimensional characteristics in digital games, has been widely investigated in the context of Game AI (Charles & Black, 2004; Smith, Lewis, Hullett, Smith, & Sullivan, 2011; Yannakakis et al., 2013; Yannakakis, 2012). Player modeling enables digital games to dynamically adapt their gameplay attuned to players’ cognitive, behavioral, and affective states. With real-time identification of accurate player models during the gameplay, player modeling has demonstrated significant promise to support player-adaptive games through interactive narrative (Riedl & Bulitko, 2013; Yu & Riedl, 2015), game balancing (Booth,
2009; Gilfeather, 2009; Lopes & Bidarra, 2011), procedural game content generation (Jennings-Teats, Smith, & Wardrip-Fruin, 2010; Shaker, Togelius, & Nelson, 2015), and adaptive pedagogical planning in educational games (Baikadi, Rowe, Mott, & Lester, 2014; Ha, Rowe, Mott, & Lester, 2011; Min et al., 2015; Min, Rowe, Mott, & Lester, 2013; Min, Ha, Rowe, Mott, & Lester, 2014; Min, Mott, et al., 2016; Mott, Lee, & Lester, 2006).

Two high-level approaches to player modeling have been examined: model-based approaches (top-down) and model-free approaches (bottom-up) (Yannakakis et al., 2013). In contrast to model-based approaches that are grounded in an existing theoretical framework (e.g., emotion theories, usability theory, cognitive theory), model-free approaches employ data-driven player modeling, which is often based on supervised machine-learning techniques (Drachen, Canossa, & Yannakakis, 2009; Mott et al., 2006; Thue, Bulitko, Spetch, & Webb, 2010; Weber & Mateas, 2009). Following the model-free player modeling approach, player models have been devised using a broad range of multimodal game interaction data streams (i.e., data-driven approach) from behavioral gameplay data (Min, Ha, et al., 2014), to physiological signals such as skin conductance and blood volume pulse (Martinez, Bengio, & Yannakakis, 2013), to bodily expressions such as body movement and posture (Savva, Scarinzi, & Bianchi-Berthouze, 2012) and facial expressions (Shaker, Asteriadis, Yannakakis, & Karpouzis, 2011).

Along this line, some recent commercial games such as Minecraft (Bergensten, 2011), Diablo (Blizzard North, 1997), and Civilization IV (Firaxis Games, 2005) have incorporated procedural content generation (e.g., map, level, character generation), with the
aim of reducing the cost of content creation and foster player engagement through dynamic content generation (Shaker et al., 2015; Yannakakis et al., 2013), while many games have utilized domain-specific rules (e.g., scripts, finite state machines) rather than devising sophisticated player models (Yannakakis, 2012). Even though these efforts have shed light on practical utilizations of player modeling in targeted domains, scalable solutions applicable to various open-world digital games have not been widely investigated.

As a core player modeling functionality, goal recognition has been the subject of increasing attention (Ha et al., 2011; Kabanza, Filion, Benaskeur, & Irandoust, 2013; Min, Mott, et al., 2016; Mott et al., 2006). Goal recognition, a restricted form of plan recognition that infers an observed agent’s plans or goals (Baker, Saxe, & Tenenbaum, 2009; Carberry, 2001; Geib & Goldman, 2009; Kautz & Allen, 1986), focuses on identifying the agent’s (e.g., human player in digital games) concrete objectives through observed sequences of low-level, primitive actions in virtual game environments.

Goal recognition in open-world digital games offers the promise of enabling games to dynamically adjust challenge levels, perform procedural content generation, and create player goal-directed believable non-player character interactions, in conjunction with other player modeling techniques. As a promising application of goal recognition, in intelligent game-based learning environments, which integrate adaptive learning technologies with rich digital game environments, goal recognition has demonstrated significant promise for enabling environments to dynamically respond to the player’s intentions, delivering tailored learning interactions, provide personalized feedback, and redirect students’ goals if they are not
conducive to learning (Alvarez, Sanchez-Ruiz, Cavazza, Shigematsu, & Prendinger, 2015; Baikadi et al., 2014; Ha et al., 2011; Lee, Liu, & Popović, 2014). In addition to runtime game and curricular adaptations, goal recognition can also facilitate the game design revision process offline, by analyzing players’ inferred goals along with the game and curriculum designers’ intentions in a post-hoc manner (Ha et al., 2011).

This dissertation investigates goal recognition in open-world digital games with fully observable, exploratory player behaviors. Goal recognition in these environments exhibits considerable uncertainty. A sequence of actions is often explainable by multiple goals, while there are a vast number of non-deterministic paths to achieve a goal. Goals are not explicitly presented to the players. In cases where players have little or no prior experience with a game (e.g., educational games administered in classrooms), players are likely to explore the game world (e.g., conversing with non-player characters, triggering game world events) in order to identify goals, rather than perform actions in order to achieve a specific gameplay objective. It is also possible that players unintentionally achieve goals through exploratory actions, abandon goals with little warning, or adopt new goals based upon recent or prior events. Thus, goal recognition in open-world digital games is characterized by considerable uncertainty, and goal recognition models must be able to operate robustly in the face of highly noisy, idiosyncratic sequences of low-level player actions.

The exploratory nature in open-world digital games has characterized that goals and actions are not only sequential (Min, Mott, et al., 2016), but also form a cyclical relationship in which players’ previously achieved goals may inform their subsequent actions, and their
current actions may influence their upcoming goals (Ha et al., 2011). Consequently, modeling sequential patterns collectively from a series of player actions and previously achieved goals is crucial for the success of goal recognition (Min, Mott, et al., 2016). As well as player gameplay logs, computational goal modeling can also utilize all available observable features such as events that players have encountered, environments that they observed, multimodal data streams that players have revealed in forms of physiological signals (e.g., electrocardiography, galvanic skin response) and bodily expressions (e.g., gaze, facial expression, posture, gesture), and static player profiles (e.g., personality and gender). Among these features, goal recognition investigated in the dissertation focuses on players’ observed gameplay behaviors, which are sequences of actions and previously achieved goals that are commonly obtainable in various open-world digital games.

Adopting the model-free player modeling approach, the dissertation presents a data-driven goal recognition framework that is scalable to multiple open-world digital games. The purpose of adopting this framework is to create a goal modeling solution that generalizes to multiple open-world digital games and to identify reliable goal recognition models that robustly operate in the face of highly noisy sequences of low-level player actions for any given game. Reliable goal recognition models should not only correctly predict player goals on overall sequences of actions, but also make early predictions (i.e., making consistently correct predictions as early as possible) and fast predictions since run-time game adaptation is a central objective of goal recognition (Blaylock & Allen, 2003).
Devising reliable computational models is a key affordance of the framework. Without an accurate goal recognition model in place, implementing a player goal-adaptive game is infeasible. This dissertation investigates two deep learning techniques (Bengio, 2009; LeCun, Bengio, & Hinton, 2015; Schmidhuber, 2014) for training computational goal recognition models. Deep learning is a family of machine learning techniques grounded in artificial neural networks, which is capable of extracting hierarchical representations through multi-level abstraction of the training data. Deep learning forms the basis for state-of-the-art techniques for a broad range of classification tasks associated with computer vision, speech recognition, and natural language processing (LeCun et al., 2015), as well as plan and goal recognition (Bisson, Larochelle, & Kabanza, 2015; Min, Mott, et al., 2016).

This dissertation examines two deep learning techniques to devise reliable goal recognition models: n-gram encoded feedforward neural networks (FFNNs) pre-trained with stacked denoising autoencoders (SDAEs) (Vincent et al., 2007) and long short-term memory networks (LSTMs), a type of gated recurrent neural networks (Gers, Schmidhuber, & Cummins, 2000; Graves, 2012; Hochreiter & Schmidhuber, 1997). SDAE conducts an initialization of weight parameters of deep neural networks in an unsupervised manner, which has been empirically found to help find a region of the parameter space that can reach a better local optimum in a non-convex optimization graphs and support improved generalization in a variety of benchmark tasks (Erhan et al., 2010; Vincent, Larochelle, & Lajoie, 2010). LSTMs are a variant of recurrent neural networks (RNNs) that are specifically designed for sequence labeling of temporal data. LSTMs have achieved high predictive
performance in various sequence labeling tasks, often outperforming standard recurrent neural networks by leveraging a longer-term memory than standard RNNs, preserving short-term lag capabilities, and effectively addressing the vanishing gradient problem (Graves, 2012).

In addition to two deep learning-based approaches, the dissertation also investigates two probabilistic graphical model-based approaches including linear-chain conditional random fields (CRFs) and Markov logic networks (MLNs). CRFs are a class of undirected probabilistic graphical models and is categorized as a discriminative machine learning technique (Sutton & McCallum, 2012). Linear-chain CRFs, a type of CRFs, consider the current and previous labels along with the predictive features, in which a weight for each feature function is learned from data. CRFs have yielded encouraging results in a broad range of structured prediction tasks in natural language processing such as text normalization (Chrupała, 2014), part-of-speech tagging (Huang, Xu, & Yu, 2015), and named entity recognition (Huang et al., 2015) as well as computer vision and bioinformatics by effectively modeling contextual relationships characterized in data (e.g., a sequence of words in a sentence, neighboring pixels in an image) (Sutton & McCallum, 2012). MLNs are a statistical relational modeling technique that formulates a first-order logic-based knowledge base into a Markov network (Richardson & Domingos, 2006). The induced Markov network contains nodes for possible groundings of predicates included in a set of first-order logic formulae, while a feature is defined per grounding of a logic formula.
1.1 Thesis Statement and Hypotheses

This dissertation presents GOALIE (Generalized Observable Action Learning for Intent Evaluation), a generalized goal recognition framework featuring valid multidimensional model evaluation, and illustrate applications of GOALIE based on two testbed open-world educational games. GOALIE consists of two key functionalities: (1) offline goal recognition model training and (2) offline multidimensional goal recognition model evaluation. In the following paragraphs, we describe the requirements for GOALIE, with a focus on applications in open-world digital games.

First, GOALIE should provide a scalable solution to devising goal recognition models for a wide range of open-world digital games. GOALIE takes as input only a set of goals and a corpus containing player action sequences performed to achieve the goals, which are generally available in most open-world digital games. An action is composed of a type, a location where the action occurs, narrative states (i.e., players’ narrative progress in the gameplay), and a sequence of previously achieved goals. In this context, the task of goal recognition is to classify player goals (i.e., labels) based on the observed action sequences (i.e., explanatory variables). It should be noted that the set of explanatory variables is not constrained to the observed action sequences, but can be extended according to the characteristics of the game being investigated.

To test GOALIE’s generalizability, the dissertation investigates multiple data corpora generated by public school students who interacted with two open-world educational games: Crystal Island: Outbreak (Rowe, Shores, Mott, & Lester, 2011) and Crystal Island:
UNCHARTED DISCOVERY (Baikadi et al., 2014; Lester et al., 2014). CRYSTAL ISLAND: OUTBREAK is a story-centric game-based learning environment for middle grade microbiology, and CRYSTAL ISLAND: UNCHARTED DISCOVERY is a game-based learning environment for upper elementary science targeting map, model, and landform education. In both educational games, players are assigned a single high-level objective: solve a science mystery and develop an understanding of landforms and navigation, respectively. Players engage in periods of exploration and deliberate problem solving under an overarching narrative, while players achieve multiple sub-goals following non-deterministic paths in each environment. In these settings, goal recognition is formalized as predicting the next sub-goal that the player will complete as part of achieving the final goal, and so is cast as a multiclass classification problem in which a trained classifier predicts the most likely goal associated with the currently observed action sequence (Min, Mott, et al., 2016). The formulation of goal recognition used in this dissertation assumes that a given sequence of actions maps to a single goal, and no interleaving occurs between actions associated with different goals, since the goal recognition corpus does not lend itself to this type of analysis (Ha et al. 2011). Computational goal recognition models are devised in a machine-learned approach, taking as input action sequences and as output player goals achieved by the action sequences, independently for each of the two open-world educational games.

Second, it is crucial for GOALIE to evaluate multiple aspects of induced goal recognition models and identify the most reliable model that robustly operates in the face of highly noisy, idiosyncratic sequences of player actions in a given open-world digital game.
Reliable goal recognition models should not only correctly predict player goals on overall sequences of actions, but also make early predictions (i.e., making consistently correct predictions as early as possible), since run-time game adaptation is a central objective of goal recognition. To address this demand, GOALIE supports offline multidimensional evaluations of goal recognition models through five metrics including three conventional metrics (accuracy rate, convergence point, and convergence rate) (Blaylock & Allen, 2003) and two novel metrics (standardized convergence point and n-early convergence rate). Among these metrics, convergence point (Blaylock & Allen, 2003) measures how early goal recognition models can consistently make accurate predictions within a converged sequence, which is an action sequence whose last goal prediction is correct. Min and colleagues (Min, Baikadi, et al., 2016; Min, Ha, et al., 2014; Min, Mott, et al., 2016) pointed out potential problems with this classic convergence point metric: it can be misleading with respect to favoring models with lower convergence rates, since the metric ignores non-converged action sequences. GOALIE addresses this problem by introducing two novel metrics and uses the novel metrics over the corresponding conventional metrics to identify reliable goal recognizers for a given open-world digital game.

First, the standardized convergence point metric measures the convergence point regardless of the convergence of a sequence. In this manner, there is a sharp contrast between the classic convergence point and standardized convergence point in the sense that the latter penalizes non-converged action sequences to appropriately capture models’ early prediction capacity while the former simply ignores non-converged ones.
The other novel metric is \( n \)-early convergence rate. The \( n \)-early convergence rate extends the convergence rate (Blaylock & Allen, 2003) that measures the percentage of action sequences in which the last goal prediction is correct, by considering the last \((n+1)\) action predictions. In a special case of this metric when \( n \) equals 0, the definition of \( n \)-early convergence rate is the same as the conventional convergence rate. The dissertation measures models’ early prediction capacity based on these two novel metrics over the two corresponding classic metrics.

Selecting a model with high predictive performance and early prediction capacity is of significant importance, since inaccurate player goal recognitions can subsequently mislead the run-time game adaptation, thereby severely disrupting the player’s game experiences. For example, providing incorrect goal-informed game adaptations for a player who is about to achieve a goal (i.e., poor \( n \)-early convergence rate) may result in considerable confusion to the player and disrupt her plan. As another example, a challenge adaptation driven by inaccurate goal recognition might diminish players game experiences, because improper difficulty level in games has been found to cause players to become bored when challenge levels are too low or frustrated when challenge levels are too high (Missura & Gärtner, 2009). Therefore, devising reliable, efficient goal recognition models is a critical next-step in building a robust goal recognition framework.

With this aim, GOALIE supports devising various kinds of computational goal recognition models, and then evaluating the induced models in terms of the predictive accuracy and early prediction capacity. Among various machine-learning techniques for goal
modeling, sequential patterns characterized in player gameplay behaviors within open-world digital games have inspired us to investigate machine-learning techniques for sequence labeling. In parallel, recent years have witnessed an explosion of interest in deep learning, which has contributed to significant advances in a wide range of pattern recognition research such as computer vision, speech recognition, and natural language processing (LeCun et al., 2015; Schmidhuber, 2014). Additionally, more recently, deep learning has also demonstrated considerable promise as a core computational modeling approach for goal and plan recognition tasks (Bisson et al., 2015; Min, Mott, et al., 2016). Inspired by this work, the dissertation hypothesizes that deep learning approaches targeted to sequence labeling such as long short-term memory networks would provide the best solution for goal modeling in open-world digital games. Along this line, the dissertation investigates five primary hypotheses as follows:

• Hypothesis 1: The pipeline for goal recognition supported by GOALIE, including model training and model evaluation process, is scalable to multiple open-world digital games such as Crystal Island: Outbreak and Crystal Island: Uncharted Discovery.

• Hypothesis 2: LSTM-based goal recognition models outperform conditional random fields, n-gram encoded feedforward neural networks pre-trained with stacked denoising autoencoders, and Markov logic networks with respect to the predictive accuracy across the two open-world educational games, when evaluated using GOALIE.

• Hypothesis 3: LSTM-based goal recognition models outperform conditional random fields, n-gram encoded feedforward neural networks pre-trained with stacked denoising
autoencoders, and Markov logic networks with respect to the convergence rate and convergence point, across the two open-world educational games, when evaluated using GOALIE.

- **Hypothesis 4**: LSTM-based goal recognition models outperform conditional random fields, $n$-gram encoded feedforward neural networks pre-trained with stacked denoising autoencoders, and Markov logic networks with respect to the two novel metrics introduced in GOALIE, standardized convergence point and $n$-early convergence rate across the two open-world educational games, when evaluated using GOALIE.

- **Hypothesis 5**: LSTM-based goal recognition models that harness distributed action representations through a linear projection layer outperform the corresponding LSTM models that utilize the one-hot encoding-based discrete action representation with respect to the accuracy rate, convergence rate, $n$-early convergence rate, and standardized convergence point across the two open-world educational games, when evaluated using GOALIE.

### 1.2 Contributions

We introduce a generalized goal recognition framework that is scalable to multiple open-world digital games, which represents a significant advance toward providing goal-driven game adaptations at run-time. Moreover, since the LSTM-based goal recognition models automatically extract predictive features, such as distributed action representations that effectively represents discrete actions in a dense, continuous vector space, successful outcomes of the LSTM-based goal recognition approach will further enhance generalizability
of the GOALIE framework by reducing the labor-intensive requirement to manually engineer hand-crafted features targeted to each game. This dissertation builds on research that has made contributions to the fields of Goal Recognition, Plan, Activity, and Intent Recognition (PAIR), Player Modeling, and Game AI:

- Goal recognition models devised with \( n \)-gram encoded feedforward neural networks pre-trained with stacked denoising autoencoders (FFNNs) significantly outperform Markov logic networks (MLNs)-based goal recognition models with respect to prediction accuracy rates in 10-fold cross validation (Min, Ha, et al., 2014).

- FFNN-based goal recognition models significantly outperform MLN-based goal recognition models with respect to the convergence rate, a metric that measures the percentage of sequences that are eventually classified to the correct goal, in 10-fold cross validation (Min, Ha, et al., 2014).

- Goal recognition models devised with long short-term memory networks featuring distributed action representations (LSTMs) significantly outperform both the FFNN and MLN-based goal recognition models with respect to prediction accuracy rates in 10-fold cross validation (Min, Mott, et al., 2016).

- The LSTM-based goal recognition approach significantly outperforms both the FFNN and MLN-based goal recognition approaches with respect to the convergence rate in 10-fold cross validation (Min, Mott, et al., 2016).

- Previous work has reported that the classic convergence point metric can be misleading with respect to favoring models with a better early prediction capacity. This results stems
from the *inherent tension between convergence rate and convergence point*, in which accurate models (i.e., models achieving high accuracy rates and convergence rates) have to consider more noisy action sequences that eventually converged to correct goals but are not trivial to predict when calculating the convergence point, thereby inducing higher (i.e., worse) convergence points (Min, Baikadi, et al., 2016; Min, Ha, et al., 2014; Min, Mott, et al., 2016).

### 1.3 Organization

The dissertation is organized as follows. Chapter 2 discusses background on plan, activity, and intent recognition (PAIR) and deep learning as well as related work on user modeling in educational games. Chapter 3 presents a high-level illustration of open-world digital games enriched with a goal-directed game adaptation feature with a particular focus on the operational pipeline that includes goal recognition model training, model evaluation, model deployment, online goal recognition, online goal-directed game adaptation, and game revision. Chapter 3 further discusses GOALIE in terms of (1) model training with illustrations using three previous goal-modeling techniques (i.e., Bayesian inference models, Markov logic networks, and feedforward neural networks) and (2) model evaluation using the five metrics described in a preceding paragraph. Chapter 4 describes two goal recognition data corpora that the dissertation examines, along with the two corresponding testbed educational games: CRYSTAL ISLAND: OUTBREAK and CRYSTAL ISLAND: UNCHARTED DISCOVERY. Chapter 5 describes the two novel goal-modeling approaches in depth: long short-term memory networks and linear-chain conditional random fields, specifically in the
context of goal recognition in open-world digital games. Chapter 6 demonstrates empirical evaluation results for the investigated goal recognition models with respect to the five multidimensional metrics as well as precision, recall, F1, and prediction speed for high performing models. This evaluation is conducted in player-level 10-fold cross validation for each open-world digital game. Chapter 7 discusses the findings from the evaluation as well as the limitations of the current work. Finally, Chapter 8 summarizes the contributions of this work, revisits the hypotheses designed to support the dissertation statement, and discusses several possible directions for future research.
CHAPTER 2

RELATED WORK

The goal recognition framework presented in this dissertation is situated at the intersection of three areas of work: (1) goal recognition, a multi-class classification task for predicting players’ high-level intention based on an observed sequence of low-level game behaviors, (2) deep learning, a family of machine learning techniques that use artificial neural networks, and (3) intelligent game-based learning environments, which serve as the testbeds for the dissertation research. The remainder of this chapter is organized as follows. Section 2.1 describes plan, activity, and intent recognition. In Section 2.2, a brief history, a variety of algorithms, and recent achievements of deep learning are presented. Finally, Section 2.3 introduces user-modeling research centering on modeling users’ knowledge.

2.1 Plan, Activity, and Intent Recognition

In human-human interactions, humans can adeptly infer others’ cognitive and affective states based on observable cues such as bodily expressions and linguistic expressions in addition to contextual information, and this social intelligence plays a pivotal role in interpersonal communication and relationships (Baker et al., 2009; Sukthankar, Geib, Bui, Pynadath, & Goldman, 2014). With the aim to model humans’ cognitive skills and social intelligence from an AI perspective, a broad range of research has been undertaken to investigate plan, activity, and intent recognition (PAIR).

PAIR is a modeling task that predicts an agent’s high-level plans, objectives, and activities based on a sequence of low-level observations (Baker et al., 2009; Geib &
Goldman, 2009; Kautz & Allen, 1986; Sukthankar et al., 2014). Prior work on PAIR has largely focused on observation sequences in which agents’ actions are rationally motivated by well-defined objectives. For instance, smart space activity monitoring (Duong, Phung, Bui, & Venkatesh, 2009), network security (Geib & Goldman, 2009), natural language story understanding (Singla & Mooney, 2011), user interactions (Goodman & Litman, 1992), digital games (Bisson et al., 2015; Fagan & Cunningham, 2003; Gold, 2010; Kabanza et al., 2013; Synnaeve & Bessière, 2011; Weber & Mateas, 2009), and intelligent tutoring systems (Alvarez et al., 2015; Brusilovsky & Millán, 2007; Conati, Gertner, & Vanlehn, 2002; S. J. Lee et al., 2014) have served as testbed applications for plan, activity, and intent recognition, among others (Sukthankar et al., 2014).

It should be noted that “intent recognition” (or “intention recognition”) has been used interchangeably used with both “plan recognition” or “goal recognition.” In some work, “intent recognition” indicates “plan recognition” (Demiris, 2007; Geib & Goldman, 2009; Tomasello, Carpenter, Call, Behne, & Moll, 2005), while “intent recognition” denotes “goal recognition” in other work (Baikadi et al., 2014; Han & Pereira, 2013; Sadri, 2012). This dissertation adopts the latter definition, and interchangeably uses intent recognition and goal recognition. Plan recognition, goal recognition, and activity recognition are presented in the following sections.

2.1.1 Plan Recognition

Plan recognition (Baker et al., 2009; Kautz & Allen, 1986; Schmidt, Sridharan, & Goodson, 1978) is a player-modeling task for predicting an agent’s high-level plans and goals based on
a sequence of observed behaviors, in which the agent can be either human or machine (Fagan & Cunningham, 2003; Ramirez, Geffner, Ram, & Geffner, 2010; Sukthankar et al., 2014). Compared to planning in which a plan is sought to achieve the given goal, the objective of plan recognition is to infer future plans and goals utilizing already-observed actions, and thus plan recognition can be formulated as inverse planning (Baker et al., 2009; Ramirez et al., 2010; Sadri, 2012).

Three types of plan recognition can be distinguished: keyhole, intended, and adversarial plan recognition, based on the role of the agent whose goal is being observed (i.e., observee) (Avrahami-Zilberbrand & Kaminka, 2014; Carberry, 2001; Cohen, Perrault, & Allen, 1981; Geib & Goldman, 2009). First, keyhole plan recognition assumes that the observee is unaware or does not care that she is being observed, and executes actions without intentions of impacting the observer’s recognition. On the other hand, both the intended and adversarial plan recognition assumes that the observee knows that she is being observed, but they differ in the way of changing her behaviors, such as in a cooperative way (intended plan recognition) (e.g., communications) or an adversarial way (adversarial plan recognition) (e.g., warfare, real-time strategy games). In these last two types of plan recognition scenario, the observee purposefully takes actions to aid (intended plan recognition) or thwart (adversarial plan recognition) recognition of her plan, while the observer may simultaneously execute actions that influence the plans of the observee (Albrecht, Zukerman, & Nicholson, 1998; Bisson et al., 2015).
While classic plan recognition work used rule-based systems that require domain experts to design such rules and so does not offer a scalable solution to various tasks (Sukthankar et al., 2014), much modern work on plan recognition has investigated a plan library-based plan recognition method initially outlined by Kautz and Allen (1986). The plan library approach performs deductive reasoning over a set of first order logic formulae through circumscription. In this approach, a plan library is described in the form of a plan graph that consists of top-level actions as root nodes, in which each top-level action is expanded with unordered sets of child actions, called plan decomposition (Geib & Goldman, 2009). Thus, the task of plan recognition is cast as identifying a set of possible plan hypotheses consistent with a sequence of observed actions, which can be solved as a graph-covering problem on the plan graph. This plan library-based approach has provided a fundamental framework for much of plan recognition work to date (Geib & Goldman, 2009; Sukthankar et al., 2014).

There are two issues centered on this classic approach: (1) the runtime prediction complexity is exponential in the size of the plan library, and (2) it is unable to deal with ambiguity when multiple plan hypotheses are explained by observed actions (Blaylock, 2005). Delving into the second issue, the plan library-based plan recognition approach (1986) does not account for prior probabilities of goals (Charniak & Goldman, 1993; Geib & Goldman, 2009; Singla & Mooney, 2011; Sukthankar et al., 2014). In other words, goals that are consistent with a sequence of observed actions would provide the same degree of explanation for the actions. However, goals in real-world problems often have different prior
probabilities, where a goal is more likely to be achieved than other goals, even though all of them are explained by the current observation (Geib & Goldman, 2009).

Charniak and Goldman (1993) presented an abduction-based plan recognition approach in the context of the plan library-based approach. Abduction-based reasoning postulates that if a plan is achieved by a sequence of actions and an agent performs the action sequence, the agent is inferred to execute the plan (Charniak & Goldman, 1993; Geib & Goldman, 2009; Sukthankar et al., 2014). To address the problems of prior probabilities of goals in Kautz and Allen (1986), Charniak and Goldman proposed a probabilistic graphical model-based approach utilizing Bayesian networks. They cast the plan recognition as finding plans and goals that have the highest posterior probability in the context of probabilistic abductive reasoning, and empirically evaluated the proposed method on a natural language story understanding task.

Since the work of Charniak and Goldman (1993), probabilistic abductive reasoning has been investigated in the form of Bayesian belief networks (e.g., Albrecht, Zukerman, & Nicholson, 1998; Geib & Goldman, 2009; Pearl, 1988), hierarchical hidden Markov models (e.g., Bui, Venkatesh, & West, 2002; Bui, Phung, & Venkatesh, 2004), and also undirected graphical models such as conditional random fields (Hu & Yang, 2008; Liao, Fox, & Kautz, 2007a; Vail & Veloso, 2008). Markov logic networks (Domingos et al., 2009), which combine first-order logic and probabilistic graphical models in a single representation, have been also investigated for plan recognition (Sadilek & Kautz, 2012; Singla & Mooney, 2011).
In addition to these probabilistic graphical model-based approaches, other plan recognition techniques (sometimes as a hybrid approach with probabilistic models) based on plan library-based methods have been investigated, including parsing and case-based plan recognition. First, parsing is inspired by natural language parsing and understanding because of its similarities to action parsing and plan recognition (Geib & Steedman, 2007; Vilain, 1990). A context-free grammar of plans is generated based on a plan library, and the plan recognition is conducted by parsing observed actions utilizing the induced grammars (Vilain, 1990). In this approach, the runtime complexity is reduced to a polynomial time, which effectively addresses the intractability that underlies in the classic work by Kautz and Allen, but at a cost due to assumptions made in this work (Blaylock, 2005).

Pynadath and Wellman further explored parsing methods. They examined plan libraries in the context of probabilistic context-free grammars (D Pynadath & Wellman, 1996) and probabilistic state-dependent grammars (Pynadath & Wellman, 2000), which subsequently inform Bayesian plan recognition model structures. A statistical parsing approach, which incorporates parsing with probabilistic graphical modeling, has shown promise in addressing challenges stemming from partial observations (David Pynadath & Wellman, 1995) and interleaved and partially-ordered plans (Geib & Goldman, 2009).

Second, case-based plan recognition leverages machine learning techniques to induce plan libraries in a data-driven approach (Fagan & Cunningham, 2003). In most of plan library-based plan recognition work, a plan library must be handcrafted, which is not only labor-intensive but also requires a significant degree of runtime complexity in order to
process a considerable size of plan library and reason about an agent’s plans. To address authoring issues, case-based plan recognition devises the plan library with machine-learning techniques (Fagan & Cunningham, 2003; Synnaeve & Bessière, 2011; Weber & Mateas, 2009). This approach offers several benefits over the handcrafted plan-library approach: it enables a plan library to be automatically learned in a data-driven approach and the induced plan library can support some sets of idiosyncratic plans and goals that individual agents revealed relative to the manual approach. Some case-based plan recognition systems incrementally learn the plan library that is initialized with generic cases and is tailored as more personal plans are observed (Cheng & Thawonmas, 2004; Fagan & Cunningham, 2003).

Recently, Bisson and colleagues proposed recursive neural network-based decision models (Bisson et al., 2015). Recursive neural networks, a deep learning technique that was first proposed for natural language parsing (Socher, Manning, & Ng, 2010), have been investigated to automatically extract features that discriminate between correct and incorrect plan hypotheses and predict the plan hypothesis that best explains the observed action sequence. Evaluations of recursive neural network-based plan recognition suggest that it outperforms previous state-of-the-art plan recognition algorithms: a probabilistic plan-library based approach (Geib & Maraist, 2008) and an inverse planning approach (Ramirez et al., 2010). The evaluations were conducted with three benchmark datasets, including a real-time strategy game dataset.
In summary, the plan library-based approach requires an plan library and a decision model \textit{a priori} (Bisson et al., 2015). The plan library specifies the space of plan hypotheses using a hierarchy of goals and actions to achieve the goals, and the decision model describes how the observee makes a decision based on the plan library and finds the most likely plans based on prior observations. Plan libraries are crafted utilizing hierarchical task networks, partially-ordered multiset context-free grammars (Bisson et al., 2015; Geib & Goldman, 2009), or Markov logic formulae (Song et al., 2013), while decision models have been often devised utilizing machine learning techniques. While the plan library based approach has achieved great success in a wide range of plan recognition tasks, designing plan libraries and decision models pose significant challenges; both cases that the plan hypothesis space circumscribed by the plan library does not specify all possible actions and the decision model does not accurately recognize the agent’s plan will together degrade the performance of plan recognition. Thus, both of these should be carefully designed for a successful implementation of plan recognizers.

In contrast to the plan library based approach, a salient line of investigation has addressed plan recognition by dispensing with the need for a plan library (Baker et al., 2009; Ramirez & Geffner, 2011) by interpreting plan recognition as an inversion of action planning given a goal, called \textit{inverse planning} (Baker et al., 2009; Ramirez & Geffner, 2009, 2011; Ramirez et al., 2010). The inverse planning approach calculates posterior probabilities of all available goals, using planning algorithms such as Strips (Ramirez & Geffner, 2009),
Markov decision process (MDP) (Baker et al., 2009), and partially observable MDP (Ramírez & Geffner, 2011).

In Ramírez & Geffner’s work (2009), the goal probability is calculated based on the cost difference between two optimal plans to achieve the goal. More specifically, for each goal, (1) an optimal plan is generated achieving the goal being consistent with the observation sequence and (2) another optimal plan is generated achieving the same goal being inconsistent with the observed sequence. The cost difference between the two induced plans informs computation of the likelihood of each goal. While this method eliminates the labor-intensive plan library creation effort, it requires invoking a planning algorithm twice for each goal and for each observation update and thus results in a scalability problem especially in complex domains (Bisson et al., 2015; E-Martín, R-moreno, & Smith, 2015).

Bayesian inverse planning calculates the posterior probability of goals given observed actions and the environment (Baker et al., 2009). Following the Bayesian rule, this posterior probability is proportional to the likelihood of actions given the goal multiplied by the prior probability of goals, in which the likelihood is solved using probabilistic planning in an MDP. This work assumes that agents’ policies follow the principle of rationality that is agents whose goals are predicted choose action sequences that minimize the expected cost to achieve their goals (i.e., rational agents). According to this assumption, action costs are defined to be proportional to the negative length of the agents’ movement. Notably, instead of adopting a deterministic policy that was found by maximizing the defined state-action value function, the authors assume that agents have a probability distribution over actions
parameterized by $\beta$ (i.e., a noise in agents’ actions), thereby formulating the likelihood function as probabilistic planning that accounts for an *approximate principle of rationality*.

Most plan recognition tasks are influenced by several sources of uncertainty such as noisy sensors, sub-optimal plans, and actions with stochastic effects, but they assume that agents’ actions are directly driven by concrete goals held by the agents (Min, Mott, et al., 2016). In contrast to these environments, open-world digital games, which often do not explicitly present goals to players, are marked by highly idiosyncratic sequences of player actions (Ha et al., 2011; Min, Ha, et al., 2014). In cases where players have little or no prior experience with a game, players are likely to explore the game world (e.g., conversing with non-player characters, triggering game world events) in order to identify goals, rather than perform actions in order to achieve a specific gameplay objective. It is also possible that players unintentionally achieve goals through exploratory actions, abandon goals with little warning, or adopt new goals based upon recent or prior events. Thus, devising a reliable plan library or using a planning approach in these environments is infeasible. For example, unlike Baker et al.’s work (2009), which assumes costs are defined based on the negative length of the agents’ movement, players who have no prior experience with a game are likely to carry out exploratory, non-goal driven actions, rather than perform an optimal set of actions in order to achieve a goal. On the other hand, goal recognition effectively deals with this challenge in open-world digital games by requiring only a list of goals and a corpus containing action sequences that achieve the goals, and thus holds significant potential for performing high-level game adaptations.
2.1.2 Goal Recognition

Goal recognition is a cognitive player modeling task that involves automatically inferring agents’ intermediate goals based on observed primitive actions in the environment, agents’ perceptual access to the environment, the past, current context of the environment, and the future context of the environment that will bring about due to the agents’ past and expected behaviors (Baker et al., 2009; Sadri, 2012). Because goal recognition does not attempt to identify agents’ plans including the sequence of their future actions, goal recognition is a sub-task of plan recognition (Blaylock & Allen, 2003; Ha et al., 2011; Mott et al., 2006).

Blaylock and Allen formulated the task of goal recognition with a particular focus on fast predictions, early predictions, and portability (Blaylock & Allen, 2003). In their work, the authors presented a statistical, corpus-based goal recognition method using n-gram models. Evaluations conducted on a Unix-command data corpus demonstrate that the induced goal recognizer was fast (i.e., linear time scalability in proportion to the number of goals) and robust (e.g., fairly high predictive accuracy, a capability of handling unknown actions and plans, early predictions), and provided a high degree of scalability to other domains because the approach does not require a handcrafted plan library. Following Blaylock and Allen’s work, a data-driven goal recognition approach using variable-order Markov models combined with an exponential moving average has been investigated in the Unix environment (Armentano & Amandi, 2011). This approach was empirically evaluated on the Linux Plan Corpus (Blaylock, 2005) consisting of 19 goal schemas and 48 command schemas collected from 56 users’ interactions. Empirical results indicate that the
computational approach can achieve 89.6% convergence rate, 58.1% online accuracy rate, and 51.0% convergence point, which outperform hidden Markov models with 5 hidden states, the best competing model.

Goal recognition models have been successfully developed for a wide range of application domains (sometimes in the context of plan recognition), including a home ambient intelligence (Han & Pereira, 2013; Sadri, 2012), cyber security (Geib & Goldman, 2009), terrorism detection (Jarvis, Lunt, & Myers, 2005), digital games (Bisson et al., 2015; Cheng & Thawonmas, 2004; Fagan & Cunningham, 2003; Gold, 2010; Kabanza et al., 2013; Synnaeve & Bessière, 2011; Weber & Mateas, 2009), story understanding (Charniak & Goldman, 1993), interactive storytelling (Karlsson, Ciarlini, Feijô, & Furtado, 2006), human-computer interaction and intelligent interface agents (Armentano & Amandi, 2011; Blaylock & Allen, 2003; Hong, 2001), traffic monitoring (Pynadath & Wellman, 1995), and defense (Mao & Gratch, 2004).

Goal (and plan) recognition in digital games has the potential to support dynamic, multifaceted game adaptations in various directions such as promoting users’ interest, motivation and replayability, enhancing communication efficiency between agents, and facilitating student learning especially in educational games. Accurately identifying players’ goals can play a central role in player-adaptive games possibly together with other player modeling techniques (Min, Mott, et al., 2016) through game balancing (e.g., challenge and difficulty adjustment) (Jennings-Teats et al., 2010; Missura & Gärtner, 2009), individualized narratives (Mateas & Stern, 2005; Rowe, 2013; Vannini et al., 2011), believable non-player
characters (Karpov, Schrum, & Miikkulainen, 2012; Muñoz, Gutierrez, & Sanchis, 2012; Ortega, Shaker, Togelius, & Yannakakis, 2013), level generation (Shaker, Yannakakis, & Togelius, 2010), and adaptive strategy planning (Kabanza et al., 2013; Laviers & Sukthankar, 2014; Weber & Mateas, 2009). Furthermore, educational games enriched with goal recognition can provide personalized curricula and feedback according to students’ goals (Baikadi et al., 2014; Ha et al., 2011; Min et al., 2014; Mott et al., 2006). Goal recognition can be used in conjunction with post-hoc data analyses (e.g., playtesting analysis, telemetry game development) to provide game designers insights to improve game experiences. In this case, player goal recognition can be used to enhance game mechanics and aesthetics for the future game revision (Yannakakis et al., 2013; Zoeller, 2010).

Research on goal recognition has used a variety of machine learning techniques. Probabilistic n-gram models (Blaylock & Allen, 2003) and variable-order Markov models combined with an exponential moving average (Armentano & Amandi, 2011) were investigated for intelligent interface agents. Input-output hidden Markov model (IOHMM) (Bengio & Frasconi, 1995) examined to identify a player’s current goal in an action-adventure game (Gold, 2010). In this work, three goals that represent key objectives in the game are defined and utilized as the hidden states of the IOHMM model. An evaluation found that an online-trained IOHMM goal recognition model outperformed a hand-authored finite state machine, a commonly used representational technique in commercial games. Kabanza and colleagues investigated behavior recognition in real-time strategy games (RTSs) with the aim of creating adaptable computer-controlled opponents (Kabanza,
Bellefeuille, Bisson, Benaskeur, & Irandoust, 2010). Their work extended Geib and Goldman’s PHATT algorithm (Geib & Goldman, 2009) to perform intent recognition on opponents’ behaviors. Kabanza et al. (2013) presented a heuristic weighted model counting algorithm that enables recognition of upper and lower bounds of posterior probabilities of goals in real-time strategy games. Synnaeve and Bessière investigated a probabilistic goal recognition approach in RTSs, in which plans and build trees are directly learned from game replays in unsupervised learning, while probabilistic decision models handle partial observations from agents (Synnaeve & Bessière, 2011).

Closely related to the dissertation research, Mott, Lee, and Lester (2006) investigated goal recognition in an open-world educational game with exploratory actions featuring a middle-grade microbiology education. This work presented that successful goal recognition can inform a narrative planner’s delivery of personalized narrative experiences and provide tailored pedagogical support in order to promote players’ engagement and motivation. Mott and colleagues investigated probabilistic graphical goal recognition models, and found that unigram models outperform bigrams and Bayesian networks in an empirical evaluation.

In a related educational game domain, Ha and colleagues (2011) examined a Markov logic network (MLN) framework for goal recognition that combines probabilistic inference with first-order logical reasoning. Empirical evaluations in a narrative-centered game-based learning environment suggest that Markov logic network-based goal recognition models yield significant accuracy gains beyond the $n$-gram models for predicting player goals by achieving 48.4% of the accuracy rate. Baikadi and colleagues (2014) extended Ha et al.’s
MLN-based approach (2011) investigating discovery events that are domain-specific representations of user progress, and demonstrated that this approach outperforms the previous state-of-the-art work with respect to both the goal recognition accuracy and efficiency. These previous approaches are further described in detail in Sections 3.2.1.1 and 3.2.1.2.

Along with deep learning’s significant advance in computer vision, speech recognition, and natural language processing (LeCun et al., 2015), deep learning has demonstrated considerable success in goal and plan recognition (Bisson et al., 2015; Min, Mott, et al., 2016), perhaps because of the focus on extracting hierarchical abstractions from lower-level inputs to higher-level outputs (Baker et al., 2009; Min et al., 2014). Min and colleagues (2014) investigated stacked denoising autoencoder pre-trained feedforward neural networks for goal recognition based on the same benchmark educational game data corpus investigated in (Baikadi et al., 2014; Ha et al., 2011), which significantly outperformed the previous state-of-the-art MLN models. Compared to the MLN-based approaches (Baikadi et al., 2014; Ha et al., 2011), which used a combination of hand-authored logic formulae and machine-learned weights, this deep learning approach eliminated labor-intensive feature engineering efforts by utilizing multi-level feature abstraction techniques. In recent work by Min and colleagues (Min, Mott, et al., 2016), they examined goal recognition with long short-term memory networks (LSTMs) leveraging distributed action representations. The results show that the LSTM-based goal recognition approach achieved state-of-the-art
predictive accuracy by effectively modeling temporal information characterized in observation sequences of player behavior in an open-world digital game.

### 2.1.3 Activity Recognition

While activity recognition and plan recognition share the similar aim of analyzing low-level, goal-driven behaviors to identify users’ high-level objectives, there is a sharp contrast between these two tasks (Geib & Goldman, 2009). Activity recognition centers on recognizing a single, flat activity without predicting agent plans by examining complex, hierarchical structures of activities, which is central in plan recognition. Activity recognition also shares similarities with goal recognition; however, unlike goal recognition, activity recognition focuses on analyzing low-level, spatiotemporal sensor data (e.g., video frame, motion capture data, user interface trace logs), extracting key features from sensor signals that explain users’ activities, and recognizing target activities (e.g., theft detection, health alert for a patient, smart space activity monitoring) harnessing predictive models devised based on extracted features and machine-learning techniques.

Much of previous work on activity recognition is grounded in a variety of supervised learning techniques specifically targeted to temporal data, such as variants of dynamic Bayesian networks (Duong et al., 2009; Liao, Fox, & Kautz, 2007b), variants of conditional random fields (Chieu, Lee, & Kaelbling, 2006; Hu & Yang, 2008; Liao et al., 2007a; D. L. Vail & Veloso, 2008), and combinations of multiple machine-learning techniques such as support vector machines for posture classification followed by hidden Markov models for activity modeling (Gaglio, Re, & Morana, 2015). Additionally, unsupervised learning
techniques have been explored to address the computational challenges posed by activity recognition. Freedman and colleagues (2015) investigated temporal and object relations in addition to the agent’s postural data for unsupervised activity recognition, and Wyatt et al. (2005) presented an unsupervised learning method to automatically produce labeled segmentations of activity data.

2.2 Deep Learning

Deep learning is a family of machine learning techniques grounded in artificial neural networks (Bengio, 2009). Deep learning’s capacity for hierarchical representation learning through multi-level data abstraction has achieved significant advance for various pattern recognition tasks in computer vision, natural language processing, and speech recognition (Schmidhuber, 2014; LeCun, Bengio, & Hinton, 2015). In the following sections, we present an overview of deep learning, introduce two principal deep learning approaches, namely, deep feedforward neural networks and recurrent neural networks, and describe various computational tasks that have been successfully investigated using deep neural networks.

2.2.1 Overview of Deep Learning

A classic form of artificial neural networks (NNs) that lacked a training capability was introduced by McCulloch and Pitts (1943). NNs’ performance was later enhanced by employing backpropagation training (Werbos, 1974). Rumelhart and colleagues demonstrated NNs’ capacity to extract internal representations of data through hidden layers trained with backpropagation with momentum (Rumelhart, Hinton, & Williams, 1985). A
discussion regarding the history of artificial neural networks can be found in (Schmidhuber, 2014).

Interest in NNs declined in the late 1980s due to the absence of effective methods for dealing with pathological problems (i.e., vanishing gradients, exploding gradients) that result in model underfitting when training deep NNs. Researchers began making progress on NNs again through multi-dimensional efforts, which collectively have launched a rich line of investigation on deep learning (Schmidhuber, 2014). This achievement has been made by combinations of the following contributions: (1) improvements in hardware (e.g., fast CPUs, GPU acceleration, parallel computing), (2) an increasing amount of data including both labeled and unlabeled data, (3) the advent of effective NN models such as recurrent neural networks (Elman, 1990), long-short term memory models (Hochreiter & Schmidhuber, 1997), and convolutional neural networks (LeCun, Bottou, Bengio, & Haffner, 1998), (4) unsupervised pre-training techniques such as deep belief networks (Hinton, Osindero, & Teh, 2006) and stacked denoising autoencoders (Vincent, Larochelle, Bengio, & Manzagol, 2008), (5) optimization techniques such as 2nd-order Hessian-free optimization (Martens, 2010) and adaptive learning rates (Zeiler, 2012), and (6) effective regularization methods (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014; Vincent et al., 2008).

2.2.2 Deep Feedforward Neural Networks

One type of deep learning is grounded in feedforward neural networks (FFNNs), acyclic networks that feature multiple layers including an input layer, an output layer, and one or more hidden layers, one or more neurons per layer, and weighted connections that bridge the
The \((n)\)th layer with \((n+1)\)th layer. Hidden neurons and output neurons often have a non-linear activation function (e.g., \textit{sigmoid} or \textit{hyperbolic tangent} for hidden layers, \textit{softmax} for the output layer), while input neurons are directly encoded based on the training set of interest.

Training deep neural networks is often performed with backpropagation. The training error is propagated from the output layer to the input layer, and all weights in networks are updated using a gradient descent method, in which the gradient is computed by the derivative of the error with respect to weights. Detailed descriptions of NNs can be found in the literature (e.g., Le, 2014; Le, 2015). The back-propagation algorithm using a gradient descent method has been popularly utilized to train NNs since the mid-1980s (Schmidhuber, 2014); however, this algorithm has been shown to not be scalable to deep NNs with multiple layers, not only for its slow learning rate, but also because it often gets trapped in poor local optima (Erhan et al., 2010; Vincent et al., 2010) or due to proliferation of saddle points (Dauphin et al., 2014).

As an approach to addressing these challenges, Hinton et al. (2006) proposed a deep belief network based layer-wise unsupervised pre-training technique leveraging restricted Boltzmann machines, undirected probabilistic graphical models constrained to form a bi-partite graph with two layers: a visible layer and a hidden layer, in which weights are updated using contrastive divergence learning. Another class of greedy layer-by-layer unsupervised pre-training algorithms utilizing autoencoders (AEs) was introduced for initializing the network (Bengio, Lamblin, Popovici, & Larochelle, 2007; Vincent et al., 2010). The
following paragraphs further delineate AE-based pre-training methods, including stacked AEs and stacked denoising AEs.

An AE is interpreted as a nonlinear generalization of principal component analysis, where it encodes high-dimensional input data into a low-dimensional output data by applying a deterministic function (Hinton & Salakhutdinov, 2006). As a variant of AEs, stacked autoencoders (SAEs) perform greedy layer-wise representation learning, sequentially for all layers (Bengio et al., 2007). To pre-train a model, in the first layer, a SAE encodes an example input vector $x$ in the visible layer into $h(x)$ in the hidden layer, using the current weight parameters $W_1$ and an activation function $s$ (e.g., sigmoid) (Equation 2.1). Then, it decodes the encoded input $h(x)$ to $z(x)$ using $W_2$ and $s$ (Equation 2.2), and updates the current weight parameters $W_1$ and $W_2$ so as to minimize the reconstruction error between the original input $x$ and the decoded input $z(x)$, for example using stochastic gradient descent. In the equations, $b_1$ and $b_2$ are biases for $W_1$ and $W_2$, respectively.

$$h(x) = s(W_1x + b_1)$$  \hspace{1cm} (2.1)

$$z(x) = s(W_2h(x) + b_2)$$  \hspace{1cm} (2.2)

Once the first layer is trained, the hidden neuron vector $h(x)$ activated by Equation 2.1 on the input vector $x$ now serves as a visible input vector for the next hidden layer, and this step is iteratively applied to all hidden layers. Once the last hidden layer is trained, it is connected to a supervised layer that consists of output neurons, and all the weight parameters in the deep architecture are fine-tuned using supervised learning (Bengio et al., 2007).
Stacked denoising autoencoders (SDAEs) is a variant of SAE (Vincent et al., 2010). SDAEs address a commonly observed challenge posed by SAEs, such that $W_1$ often converges to a trivial solution (i.e., identity function), particularly when the number of variables in the hidden layer is equal to or greater than the number of variables in the visible layer (Vincent et al., 2010). Figure 2.1 shows a conceptual illustration of how the SDAE algorithm learns weights for a visible layer and a hidden layer. A training example $x$ in the visible layer is corrupted (for example, random neurons are set to 0) into a partially destroyed input $x'$, then $x'$ is deterministically mapped to $h(x')$ in the hidden layer using Equation 2.1, and weight parameters ($W_1$ and $W_2$) are updated to minimize the reconstruction error ($L$) between the uncorrupted input $x$ and the decoded value of the corrupted input $z(x')$, similar to the manner in which SAEs operate. In summary, SDAEs perform a stochastic mapping of $x$ in the process of the SAEs’ layer-wise pre-training, and this regularization method has been found to help $W_1$ eschew from simply learning the identity function (Vincent et al., 2010).

Figure 2.1: A conceptual illustration of layer-wise stacked denoising autoencoders; red crosses denote corruption (Vincent et al., 2010).
2.2.3 Recurrent Neural Networks

Another type of NNs is grounded in recurrent neural networks (RNNs) (e.g., Elman, 1990). As illustrated in Figure 2.2, RNNs feature cyclic networks, which are specifically designed for sequence labeling tasks handling variant length inputs and outputs such as language and speech data (Bengio, 2009). To effectively deal with sequential inputs in the dataset, RNNs utilize shared parameters; $U$, $V$ and $W$ matrices are shared across multiple time steps, as described in the unfolded structure of RNNs (Figure 2.2, right). Updates on hidden state vectors and output vectors are made based on Equations 2.3 to 2.6. The variables $x_t$, $s_t$, and $o_t$ denote the input vector, the hidden state vector, and the output vector at time $t$, respectively.

In Equation 2.3, the variable $a_t$ is calculated by adding a linear transformation of the current input vector ($x_t$) using $W$, a linear transformation of the previous hidden state vector ($s_{t-1}$) using $U$, and a bias vector $b_1$. The hidden state vector ($s_t$) at time $t$ is computed based on the induced vector, $a_t$, and the hyperbolic tangent function ($tanh$) (Equation 2.4). A linear transformation of the hidden state $s_t$ using $V$ and a bias vector $b_2$ are used to calculate the output vector at time $t$. 

Figure 2.2: Recurrent neural networks (Goodfellow, Bengio, & Courville, 2016). Left. input vectors ($x$), hidden state vectors ($s$), and output vectors ($o$). $W$, $U$, and $V$ are weight matrices for the input-to-hidden, hidden-to-hidden, and hidden-to-output connections, respectively. Right. $x_1$, ..., $x_t$ are input vectors, $o_1$, ..., $o_t$ are output vectors, and $s_1$, ..., $s_t$ are hidden state vectors from time 1 to $t$, respectively.
output vector $o_t$ at time $t$ (Equation 2.5), and the output vector is referenced to compute $p_t$ using another activation function (e.g., softmax) (Equation 2.6 and Equation 2.7). $p_t$ is interpreted as a vector that consists posterior probabilities for all available classes, where the unit with the highest value in $p_t$ becomes the class label for the input vector, $x_t$. In Equation 2.7, $o_{ti}$ denotes the $i^{th}$ component in $o_t$ that consists of $n$ components.

$$a_t = Wx_t + Us_{t-1} + b_1$$  \hspace{1cm} (2.3)

$$s_t = tanh(a_t)$$  \hspace{1cm} (2.4)

$$o_t = Vs_t + b_2$$  \hspace{1cm} (2.5)

$$p_t = softmax(o_t)$$  \hspace{1cm} (2.6)

$$softmax(o_{ti}) = \frac{e^{o_{ti}}}{\sum_{j=1}^{n} e^{o_{tj}}}$$  \hspace{1cm} (2.7)

RNNs have addressed various forms of sequence labeling tasks (Karpathy, 2015; LeCun et al., 2015; Schmidhuber, 2014). Figure 2.3 describes several types of modeling using RNNs (Figure 2.3B – 2.3E) with a comparison to a FFNN (Figure 2.3A). Figure 2.3A illustrates a network structure for classification tasks with a fixed-size input and output (e.g., image classification that takes a hand-written digit image and predicts the actual digit). Figure 2.3B describes a task that generates a sequence output with a fixed input (e.g. image captioning that takes as input an image and outputs a sentence of words). Figure 2.3C depicts a system that predicts a fixed output using sequential inputs (e.g. sentiment analysis that takes as input a sequence of words in a sentence and outputs whether a given sentence has a positive or negative sentiment). Figure 2.3D illustrates asynchronous sequence-to-sequence labeling tasks that produce a delayed sequence of outputs using a sequence of inputs (e.g. a
machine translation system that takes as input an English sentence and translates it to a corresponding Korean sentence). A variant of this model includes Sutskever et al.’s work (Sutskever, Vinyals, & Le, 2014), in which $o_1$ is fed into the model as input at time $t+1$ to predict $o_2$ (this procedure is repeated until the last $o$ prediction). Finally, Figure 2.3E describes synchronous sequence-to-sequence labeling tasks that produce a synchronized sequence of outputs using a sequence of inputs (e.g. a frame-level video classification that takes as input each frame of the video and predicts if the frame has a human image). Figure 2.2 is in the same context as Figure 2.3E when unfolded.

Figure 2.3: Different classification tasks using recurrent neural networks (Karpathy, 2015).
Multiple variations have been derived from the seminal form of RNNs. Recursive neural networks, which relax the *from-first-to-last* sequential-chain constraint in standard RNNs by generalizing to a tree structure, have been proposed to address challenges in natural language processing work, including sentiment analysis (Socher, Perelygin, & Wu, 2013) and parsing natural scenes and natural languages (Socher & Lin, 2011). Long short-term memory networks (LSTMs) that feature three gating units have been successfully investigated on various sequence labeling challenges (Hochreiter & Schmidhuber, 1997).

An LSTM (Hochreiter & Schmidhuber, 1997) is a variant of RNNs. LSTMs maintain a longer term memory compared to traditional RNNs by using a memory block that features one or more self-connected memory cells along with three gating units: input gate, forget gate and output gate per memory cell (Figure 2.4) (Gers et al., 2000; Graves, 2012). Traditional RNNs often suffer from the vanishing gradient problem, in which gradients for front layers in deep networks decrease close to zero when trained using a backpropagation method, and exploding gradient problem, both of which prevent RNNs from storing long-term dependencies from previous time steps in the sequential data (Bengio, Simard, & Frasconi, 1994; Hochreiter, Bengio, Frasconi, & Schmidhuber, 2001). In LSTMs, the input and output gate modulate the incoming and outgoing signals on the memory cell, and the forget gate controls the previous state of the memory cell whether to remember or forget; this structure allows it to preserve gradient information over long periods of time, and thus effectively address the vanishing gradient problem (Bengio et al., 1994; Graves, 2012; Hochreiter et al., 2001; Hochreiter & Schmidhuber, 1997).
In an implementation of LSTMs without peephole connections (Graves, 2012), the input gate \( (i_t) \), forget gate \( (f_t) \), and candidate value of the memory cell \( (c'_t) \) at time \( t \) are computed by Equation (2.8) – (2.10), respectively, in which \( W \) and \( U \) are weight matrices for transforming the input \( (x_t) \) at time \( t \) and the cell output \( (h_{t-1}) \) at time \( t-1 \), \( b \) is the bias vector of each unit, and \( \sigma \) and \( \tanh \) are the logistic sigmoid and hyperbolic tangent function, respectively:

\[
\begin{align*}
  i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (2.8) \\
  f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (2.9) \\
  c'_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (2.10)
\end{align*}
\]

Once the three gate vectors are computed, the current memory cell’s state is updated to a new state \( (c_t) \), by modulating the current memory candidate value \( (c'_t) \) via the input gate \( (i_t) \) and the previous memory cell state \( (c_{t-1}) \) via the forget gate \( (f_t) \). Through this process, a memory cell decides whether to keep or forget the previous memory state and regulates the

---

Figure 2.4: A conceptual illustration of an LSTM memory block that features three gating units and a memory cell (Gers et al., 2000; Graves, 2012).
candidate of the current memory state via the input gate. This step is described in Equation (2.11):

\[ c_t = i_t \tilde{c}_t + f_t c_{t-1} \]  

(2.11)

In Equation (2.12), the output gate \((o_t)\), similarly calculated as in Equation (2.8) – (2.9), is utilized to compute the cell activation \((h_t)\) of the LSTM block, based on the computed memory state \((c_t)\) (Equation 2.13):

\[ o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \]  

(2.12)

\[ h_t = o_t \tanh(c_t) \]  

(2.13)

Compared to this variant, another variant of LSTMs has been presented: peephole LSTMs (Graves, 2012). In this architecture, the input and forget gates additionally take as input the previous memory cell’s state, while the output gate utilizes the current memory cell’s state using peephole connections.

Once the cell output vectors \((h)\) are induced per time step, the next step is using these vectors to predict the label(s) of the current training example. Implementations of the final output layer vary according to applications as illustrated in Figure 2.3. For instance, if the current task of interest is frame tagging such as Figure 2.3E, cell outputs \((h)\) at each time step will be utilized to predict the output sequences (multiple labels) in a synchronous fashion. On the other hand, if the task is goal recognition or sentiment analysis (Figure 2.3C), all memory cell outputs induced from the input sequence can be potentially utilized to predict the class label. In this case, a prediction can be accomplished by directly using the final cell output vector \((h=h_t)\) (Figure 2.5A) or an averaged vector \((h=\text{mean}(h_1, \ldots, h_t))\) on all cell output
vectors ($h_1$ to $h_t$) (i.e., mean pooling) (Figure 2.5B), among others. Then, the $h$ vector is used to predict the class label of the input sequence $x$, ($x_1, ..., x_t$), in an output layer featuring an activation function (e.g., softmax). The unit that is maximally activated (i.e., with the highest posterior probability) in the output layer becomes the most likely class label for $x$.

Figure 2.5: Two examples of LSTM to solve a classification task illustrated in Figure 2.3C. Both consist of a single, forward LSTM layer. (A) The final memory cell output at time $t$, $h_t$, is utilized to conduct classification in a softmax layer. (B) A pooling (e.g., mean, max) is performed over a sequence of memory cell outputs ($h_1$ to $h_t$) to compute a single vector, $h$, which is used for classification in a softmax layer (Carrier & Cho, 2015).

2.2.4 Deep Learning in Vision, Language and Speech

Various deep learning techniques have achieved state-of-the-art results in diverse computational tasks. Many sequence labeling tasks investing speech and language data have
been successfully achieved by recurrent neural networks (RNNs), while many computer vision tasks have been most frequently achieved by convolutional neural networks (CNNs) (Schmidhuber, 2014) in the context of deep learning.

RNNs, a variant of RNNs (e.g., gated RNNs, recursive neural networks), or a combination of RNNs with other machine learning techniques (e.g., hidden Markov models) achieved state-of-the-art performance in benchmark tasks including sentiment analysis (Socher et al., 2013), speech recognition (Graves, Mohamed, & Hinton, 2013), optical character recognition (Breuel, Ul-Hasan, Al-Azawi, & Shafait, 2013), language identification (Gonzalez-Dominguez & Lopez-Moreno, 2014), speech recognition (Geiger et al., 2014; Sak, Senior, & Beaufays, 2014), text-to-speech synthesis (Fan, Qian, Xie, & Soong, 2014), and machine translation (Sutskever et al., 2014).

CNNs along with their variants accomplished the-state-of-the-art performance in a Chinese handwriting recognition benchmark (Cireşan & Schmidhuber, 2013), image classification (Szegedy et al., 2015; Md Zeiler & Fergus, 2014), multi-digit number recognition (Goodfellow, Bulatov, Ibarz, Arnoud, & Shet, 2013), scene labeling (Farabet, Couprie, Najman, & LeCun, 2013), object detection (Szegedy, 2013), and video classification (Karpathy et al., 2014). A discussion of the broad range of computational tasks addressed by deep learning can be found in Schmidhuber (2014).

On the other hand, the distributed representation technique has been investigated for effective utilization of the input vectors compared to the one-hot representation method utilizing a sparse vector. Distributed representation has been broadly explored in a wide
range of natural language processing (NLP) tasks (Mikolov, Chen, Corrado, & Dean, 2013; Mikolov, Sutskever, Chen, Corrado, & Dean, 2013; Mikolov, Yih, & Zweig, 2013) with the aim of representing lexical units’ syntactic and semantic properties on a low-dimensional, continuous vector space, identify inter-relationships across the lexical units (e.g., words, phrases), and improve predictive performance of an NLP task of interest. Initial work on distributed representations was technically grounded in language modeling (Bengio, Ducharme, Vincent, & Janvin, 2003), an unsupervised machine learning task that finds the next word using a sequence of previous words. Words’ distributed representations induced by RNNs are found to effectively capture linguistic regularities between words, such as “queens” – “queen” = “kings” – “king”, which represents syntactic information and “queen” – “woman” = “king” – “man”, which represents semantic information (Mikolov, Yih, et al., 2013).

2.3 User Modeling in Intelligent Game-Based Learning Environments

Recent years have witnessed growing interest in intelligent game-based learning environments (a class of educational games), and their potential to foster student learning and create personalized and engaging learning experiences. Intelligent game-based learning environments are situated at the intersection of (1) digital games that increase students’ motivation through rich settings, engaging characters, and compelling plots in virtual environments, and (2) intelligent tutoring systems that foster students’ learning through tailored scaffolding and context-sensitive feedback. These environments simultaneously merge adaptive pedagogical functionalities delivered through intelligent tutoring systems
with the motivating and engaging learning experiences provided by digital games (Jackson & McNamara, 2013; Johnson, 2010; Lester et al., 2013). A key benefit of game-based learning environments is their ability to embed problem-solving challenges within interactive virtual environments, which can enhance students’ motivation (Garris, Ahlers, & Driskell, 2002) and facilitate learning through customized narratives, feedback, and problem-solving support (Rowe et al., 2011; Shute, Ventura, Zapata-rivera, & Bauer, 2009). Moreover, student interaction data from game-based learning activities can be leveraged to evaluate students’ development of competencies and progression towards learning goals (Min et al., 2015; Shute et al., 2009).

Recent work in game-based learning focuses on a broad spectrum of subject matters ranging from high school mathematics (Kebritchi, Hirumi, & Bai, 2010) and middle school computer science (Min et al., 2015) to anti-bullying (Vannini et al., 2011), language and culture learning (Johnson, 2010), science inquiry (Hickey, Ingram-Goble, & Jameson, 2009; Nelson, Kim, Foshee, & Slack, 2014), and biosafety training (Alvarez et al., 2015). Narrative-centered learning environments have investigated narrative adaptation for individual students in the context of intelligent game-based learning, and have been found to deliver experiences in which learning and engagement are synergistic (Alvarez et al., 2015; Johnson, 2010; Nelson et al., 2014; Rowe et al., 2011; Vannini et al., 2011). Intelligent game-based learning environments not only have demonstrated promise from the pedagogy aspect (Min et al., 2015, 2013), but also have been investigated to enable personalized game experiences from the game AI perspective (Min et al., 2014).
Intelligent game-based learning environments support many forms of student modeling to provide user-adaptive game and learning experiences. Student modeling has been mainly conducted in the two principal dimensions: cognitive modeling and affect modeling. Cognitive modeling centers on predicting (1) students’ plans, goals, and behaviors (Baikadi et al., 2014; Conati et al., 2002; Ha et al., 2011; Lee et al., 2014; Min, Ha, et al., 2014; Min, Mott, et al., 2016; Mott et al., 2006) and (2) performance and knowledge (Alvarez et al., 2015; Corbett & Anderson, 1994; Lee & Brunskill, 2012; Min et al., 2015, 2013; Pardos & Heffernan, 2010; Sahebi, Huang, & Brusilovsky, 2014; Shute et al., 2009; Yudelson, Koedinger, & Gordon, 2013), while affect modeling emphasizes on predicting students’ on-going emotional (Calvo & D’Mello, 2010; Picard, 2003) and engagement (Fredricks, Blumenfeld, & Paris, 2004) states during learning processes.

Goal recognition used in conjunction with knowledge modeling holds significant potential for fine-grained game and curricular adaptation in educational digital games. Dynamically predicting what the student is achieving (Min, Ha, et al., 2014; Min, Mott, et al., 2016) and accurately identifying student knowledge (Min et al., 2015, 2013) could enable intelligent game-based learning environments to promote students’ engagement and facilitate learning through customized narratives, feedback, problem-solving support, task and content generation in real-time. For instance, suppose that a student is learning the concept of binary numbers to represent base-ten numbers in a game-based learning environment (Min et al., 2015). While knowledge modeling can infer whether the students have competency about the weights associated with each bit in binary numbers, knowledge modeling itself is not capable
of identifying if the student is actively participating in the learning activities or engaged in the learning objective. Off-task behavior modeling, a cognitive model, enables to identify game behaviors that are not conducive to learning and then eventually support a fine-grained game adaptation synergistically with the student’s knowledge model. Below, we describe two knowledge-modeling approaches that have been investigated to date.

**Stealth Assessment.** Stealth assessment leverages student interaction data to provide real-time behind-the-scenes measurement of students’ learning processes and outcomes (Shute et al., 2009). Stealth assessments allow educators and researchers to draw inferences about student competencies in an invisible and non-disruptive manner. Specifically, students’ learning is measured by analyzing sequences of observable behavioral cues that revealed from learning activities without posing explicit formative assessments. Recently, Min et al. proposed DeepStealth, a deep-learning-based stealth assessment framework for student knowledge modeling in game-based learning environments (Min et al., 2015).

**Knowledge Tracing.** As another effective approach, Bayesian knowledge tracing (BKT) has been widely explored for assessing latent knowledge and skills in the context of cognitive modeling (Corbett & Anderson, 1994). BKT predicts students’ latent knowledge utilizing four parameters in hidden Markov models (HMMs). The four parameters consist of an initial probability of knowing the skill a priori, the probability of transitioning knowledge of a skill from not-known to known, the probability for a *slip* (making a mistake when applying a known skill), and the probability for *guess* (getting it correct without mastering the skill), which can be fit using optimization techniques such as expectation maximization and
conjugate gradient search (Yudelson et al., 2013). Individualized BKT models that consider learner-specific aspects such as initial probability of mastery (Pardos & Heffernan, 2010), speed of learning (Yudelson et al., 2013), and other parameters (J. I. Lee & Brunskill, 2012) have demonstrated improved predictive performance compared to the classic BKT approach. Recently, Piech et al. investigated knowledge tracing using the deep learning formalism, called Deep Knowledge Tracing (Piech et al., 2015). This work demonstrated that recurrent neural network-based knowledge modeling significantly outperforms the classic HMM-based BKT method with respect to predictive performance on a range of knowledge tracing datasets.
CHAPTER 3
GOAL RECOGNITION-BASED OPEN-WORLD DIGITAL GAMES

This chapter introduces how open-world digital games can be enriched with player goal recognition. Implementing a goal recognition-based open-world digital game requires goal recognition model training, model evaluation, model deployment, runtime goal recognition, runtime goal-directed game adaptation, and game revision. This dissertation centers on the first two steps, model training and evaluation, as formalized in the Generalized Observable Action Learning for Intent Evaluation (GOALIE) framework. This chapter is organized as follows: we first provide a high-level description of goal recognition-enabled digital games with respect to required components in the pipeline, and then detail the GOALIE framework that accounts for goal recognition model training and evaluation. The goal recognition model training step is described with three previously investigated goal modeling techniques: Bayesian belief networks (Mott et al., 2006), Markov logic networks (Baikadi et al., 2014; Ha et al., 2011), and n-gram encoded feedforward neural networks (Min, Ha, et al., 2014) for a concrete illustration of the process. Finally, the model evaluation step grounded in the five metrics including three conventional and two novel metrics is presented.

3.1 An Overview of Goal Recognition-Enabled Digital Games

Figure 3.1 presents a conceptual illustration of open-world digital games that utilize a goal recognition feature to provide player-adaptive game experiences. Once the player begins playing the game, the game receives a stream of user interaction data, sends the interaction data to the runtime Goal Recognition Engine, and interactively controls the game progress
through the Goal-Directed Game Adaptation Engine. The information sent to the Goal Recognition Engine is not bound to game trace logs, but can include any type of interaction data observed through multimodal interfaces such as players’ bodily expressions and physiological signals.

Multimodal data streams have been demonstrated to be tightly linked to affective and cognitive processes (Pantic, Pentland, Nijholt, & Huang, 2007; Yannakakis et al., 2013). Martinez et al. have investigated physiological sensor data to devise players’ affect models utilizing skin conductance and blood volume pulse captured from interactions with digital games (Martinez et al., 2013). Along this line, multimodal physiological signals have been shown to be useful channels for differentiating affective states in learning (Calvo & D’Mello, 2010). On the other hand, physiological sensor data has been investigated for users’ activity

Figure 3.1: Open-world digital games enriched with the goal recognition feature.
recognition using conditional random fields (Chieu et al., 2006) and fuzzy Bayesian networks (Yang & Cho, 2008).

Moreover, a range of research has demonstrated that bodily expressions (e.g., facial expressions, gestures, postures, eye tracking) offer a rich source of evidence for one’s affect (Ekman & Friesen, 1977; Scherer, Bänziger, & Roesch, 2010) during learning (Grafsgaard, Wiggins, Boyer, Wiebe, & Lester, 2014; Vail, Grafsgaard, Wiggins, Lester, & Boyer, 2014), user attention and interest understanding (Arroyo et al., 2009; Mota & Picard, 2003), and plan and activity recognition in both the unsupervised learning (Freedman et al., 2015) and supervised learning (Gaglio et al., 2015). Eye tracking data streams have been also examined in the context of intelligent user interfaces that support user-adaptive interactions through a user’s high-level mental state understanding (Conati, Merten, Amershi, & Muldner, 2007).

For goal-directed game adaptations, the Goal Recognition Engine preprocesses a multi-stream user interaction data. The low-level interaction logs are encoded into a format that the goal recognition model embedded in Goal Recognition Engine take as input. Then, the Goal Recognition Engine infers a player’s high-level intention based on the goal recognition model that was identified by GOALIE (Section 3.2), and passes a list of goals along with inferred posterior probabilities to Goal-Directed Game Adaptation Engine.

Although elucidating how the Goal-Directed Game Adaptation Engine operates is out of the scope of the dissertation, we briefly discuss some expected behaviors of the game adaptation engine and prospective benefits to game developers. First, this engine takes as input information about the goal delivered from the Goal Recognition Engine, such as the
predicted goal and models’ predictive confidence (e.g., the posterior probability of the recognized goal) per player action. The Goal-Directed Game Adaptation Engine can further utilize any observable information such as the game progress (e.g., actions, locations, triggered events, narrative progress, previously achieved goals) and the current environment (e.g., object, non-player character locations) as well as the static player profile.

Then, the Goal-Directed Game Adaptation Engine must determine the type of adaptation (e.g., event generation, content generation) and timing of the adaptation (e.g., if the goal recognition confidence is low, the engine may decide to bypass the current prediction), both of which can be operated by a decision model embedding a set of rules or a machine-learned model. Player goal-driven adaptations can be realized in a variety of forms based on the focus of the game. For example, intelligent game-based learning environments can leverage goal recognition models to predict students’ progress towards learning goals, enabling delivery of tailored hints and feedback to enhance learning. Alternatively, goal recognition models can be utilized to identify user objectives that conflict with a game’s intended narrative (e.g., eliminating an important character), thereby triggering an intervention to re-direct the user’s behavior in the gameworld (Ha, Rowe, Mott, & Lester, 2014).

In addition to online game adaptations, goal recognition can serve as a post-hoc game analysis and refinement tool. Playtesting analysis and game telemetry involves offline analyses on the trace log data collected during gameplay sessions to identify game components that positively appealed to players or caused problems for players (Ha et al.,
Developers’ telemetry efforts can inform game refinements (e.g., game mechanics, game design changes) and directions of the future game design, to improve playing experiences. By analyzing players’ high-level intentions in conjunction with raw game trace logs, goal recognition can further facilitate the playtesting analysis and game refinement process by game developers. In the following sections, we describe the GOALIE framework that performs offline goal recognition model training and evaluation, which constitutes a pivotal step to enable goal-directed game adaptations and game telemetry-driven game refinement.

3.2 Generalized Observable Action Learning for Intent Evaluation (GOALIE)

GOALIE (the bottom box in Figure 3.1) is a generalized goal recognition framework that accounts for model creation and multidimensional evaluation. The objectives of this framework are to provide a scalable goal modeling method to various open-world digital games and identify reliable goal recognition models that robustly operate in the face of highly noisy sequences of low-level player actions characterized in open-world games. Goal recognition models should not only correctly predict player goals on overall sequences of actions, but also make early and speedy predictions since run-time game adaptation is a central objective of goal recognition. To address the demands for accuracy and early prediction, GOALIE supports multidimensional evaluations of goal recognition models through five metrics including three conventional metrics (accuracy rate, convergence point, and convergence rate) (Blaylock & Allen, 2003) and two novel metrics (standardized convergence point and $n$-early convergence rate) (Min, Baikadi, et al., 2016). The following
sections detail the two principal steps, model training and model evaluation, provided by GOALIE to identify reliable goal recognition models.

### 3.2.1 GOALIE: Goal Recognition Model Training

Goal recognizers make predictions on goals using observable game behaviors. The training process to induce a goal recognizer can be performed either offline or in an incremental manner as a new set of user data becomes available. In supervised learning, training computational models requires a data corpus that consists of observations (e.g., actions) and the corresponding labels (e.g., goals) per observation. The data corpus used to train classifiers can be obtained from players’ previous interactions with the game, the class labels for the data can be obtained in various ways such as investigating in-game achievements, players’ self-reports, and retrospective labeling by either players or experts.

To provide a generalized goal modeling solution scalable to various machine learning techniques as well as open-world digital games, GOALIE suggests four action properties to constitute a player action: *action type, location, narrative states, and previously achieved goals* (Min, Ha, et al., 2014), which we intentionally designed to be general and applicable to a broad array of digital games. This action schema has been investigated in a wide range of previous work that explored a range of machine learning techniques (Ha et al., 2011; Min, Ha, et al., 2014; Min, Mott, et al., 2016).

**Action Type.** The type of current action taken by the player, such as “move” to a particular location, and “talk” to a non-player character. The action type can be extended by combining
its argument, such as talk to “Robert”, and in this case “TalkRobert” and “TalkJames” can be treated as two different actions.

**Location.** The location in the virtual environment, where a current player action is taken.

**Narrative State.** An indication of the player’s progress in solving the narrative scenario, when a current player action is taken.

**Previously Achieved Goals.** An indication of the previous goals achieved by the player, when a current player action is taken.

It should be noted that, although the four action properties that form an action are likely to be specified in various types of open-world digital games, this list is neither the minimum nor maximum set of properties to represent an action; for example, goal recognition in multi-player games would need to incorporate additional information about multiple players’ observed actions and goals, as players can interact with each other; in the case of the Go game, the definition of the narrative state may not be obvious.

If a trainable data corpus is ready, various machine learning techniques can devise computational goal recognizers. Some machine learning techniques require domain expert knowledge for handcrafting structures or logics demanded by predictive models (e.g., Bayesian belief networks, Markov logic networks), while structure learning techniques utilizing search algorithms have enabled to automatically learn model structures using the dataset (e.g., simulated annealing for Bayesian networks (Bouckaert, 1995)). Some other models do not need expert-crafted models by constraining the model structure, as in Naïve Bayes models, hidden Markov models, and linear-chain conditional random fields. Even with
the constrained representational capability, these models exhibit benefit over manually crafted models by reducing labor-intensive feature engineering efforts and decreasing both the learning and inference complexity.

Below we present three previously examined goal-modeling approaches, applications of which have been evaluated with open-world digital games. The chapter illustrates the work from a domain-independent point of view, focusing on the action schema featured by the four action properties and goal modeling based on the action schema; the first approach is based on a probabilistic graphical model using \( n \)-gram models and Bayesian networks (Mott et al., 2006), second is based on Markov logic networks (Baikadi, 2014; Ha et al., 2011), and third is based on \( n \)-gram encoded feedforward neural networks pre-trained with stacked denoising autoencoders (Min, Ha, et al., 2014). All these approaches have demonstrated significant success for goal recognition in the open-world digital game domain.

### 3.2.1.1 Probabilistic Graphical Model-Based Goal Recognition

Mott et al. examined goal recognition in an open-world serious game (an edition of CRYSTAL ISLAND) featuring interactive science narratives for education, training, and entertainment (Mott et al., 2006). A key objective of this work is both accurately and quickly recognizing users’ goals, so that narrative planners can generate contextualized story plots and character actions customized to individual players’ goals. To achieve this objective, the authors present an inductive reasoning approach to predicting users’ goals by learning Bayesian goal recognition models. This approach was empirically evaluated on a data corpus based on
players’ interactions with an educational game, and the authors demonstrated that $n$-gram models and Bayesian inference models offered significant predictive power.

In this work, goal recognition is defined as follows: “Given a sequence of $n$ observed user actions, $n$ associated narrative states, and $n$ occurred player locations, goal recognition is to identify the most likely goal $G^*$ from a set of candidate goals that accounts for the action sequence in the given context” (Mott et al., 2006). More formally, if a sequence of $n$ observation actions: $O_1, O_2, \ldots, O_n$ is given, wherein $O_t$ represents the observed action at time $t$ that has a tuple of action, narrative state, and location at time $t$, the most likely goal $G^*$ is formulated as described in Equation 3.1:

$$G^* = \text{argmax}_G P(G \mid O_1, \ldots, O_n)$$  \hspace{1cm} (3.1)$$

Applying the Bayesian rule and chain rule to Equation 3.1, a unigram model is represented as in Equation 3.2, and a bigram model is formulated as in Equation 3.3:

$$G^* = \text{argmax}_G P(G) \prod_{i=1}^{n} P(O_i \mid G)$$ \hspace{1cm} (3.2)$$

$$G^* = \text{argmax}_G P(G) \prod_{i=1}^{n} P(O_i \mid O_{i-1}, G)$$ \hspace{1cm} (3.3)$$

Additionally, Mott et al. (2006) hand-authored a Bayesian network based goal recognition model for narrative-centered game environments considering relationships among the sequence of narrative states, the sequence of user actions, the sequence of user locations, the user’s previous goal, and the user’s current goal as described in Figure 3.2, in which subscript $i$ in a node indicates the $i^{th}$ observation.
The goal recognition formula for the proposed Bayesian network is defined as in Equation 3.4, where $A$, $L$, and $N$ denote action, location, and narrative state, respectively, $X_{0:n}$ indicates a sequence of $X$ from time 0 to time $n$, and $G'$ is a previously achieved goal.

$$G^* = \text{argmax}_G P(G|G', A_{0:n}, L_{0:n}, N_{0:n})$$

(3.4)

The authors evaluated predictive performance of these three probabilistic approaches on an open-world digital game data corpus. Empirical results in 10-fold cross validation indicate that the unigram (54.8%) outperformed the two more sophisticated models: bigram (51.5%) and Bayesian network (53.7%) in terms of the accuracy rates, while every model accomplished significantly higher performance compared to the majority selection based baseline (around 5% accuracy rate out of 20 candidate goals). These probabilistic approaches also exhibited potential in making early predictions, by which classifiers’ predictions are quickly converged to the correct goal after only a few numbers of observations. This work
has provided a framework for goal recognition that has informed more recent work on goal recognition (Baikadi, 2014; Ha et al., 2011; Min, Ha, et al., 2014; Min, Mott, et al., 2016).

3.2.1.2 Markov Logic Network-Based Goal Recognition

Ha et al. (2011) examined goal recognition in an education-focused open-world digital game with the aim of enabling player-adaptive digital games in concert with players’ dynamically changing goals. This work proposed a Markov logic networks (MLNs) based goal recognition approach. As briefly discussed in Introduction, MLNs (Richardson & Domingos, 2006) combine probabilistic inference with first-order logical reasoning. The authors evaluated the MLN-based goal recognition approach against unigram and bigram baselines, the approach investigated in Mott et al.’s work (2006). These $n$-gram comparison models were created in MLNs, using logic formulae specifically designed for each $n$-gram. The results demonstrated that MLN-based goal recognition yields significant accuracy gains beyond the two competitive probabilistic approaches (i.e., unigram and bigram models) for predicting player goals in an open-world digital game.

MLN-based goal recognition has focused on the characteristics of undirected probabilistic graphical models that can effectively capture cyclical relationships between actions and goals characterized in open-world digital games. An MLN is devised using a set of hand-authored first-order logic formulate and weights trainable using a data corpus. The authors noted that the MLN-based approach has a representational benefit over hidden Markov models (HMMs) (Rabiner, 1989); whereas HMMs require an unwieldy number of observation symbols ($x$ actions x $y$ locations x $z$ narrative states) to represent the investigated
goal recognition domain, MLNs grounded in first-order logics enables them to only use a compact size of symbols (x actions + y locations + z narrative states) (Ha et al., 2011).

In this work, four predicates are considered per action property. The goal predicate serves as a hidden predicate: \(\text{goal}(t, g)\), while the other three predicates serve as observed predicates: \(\text{action}(t, a)\), \(\text{loc}(t, l)\), and \(\text{state}(t, s)\). Since the player’s goal is inherently hidden from the goal recognizer’s perspective, it should be computationally inferred, while all the other predicates are observable by the goal recognizer.

Using the four predicates, 13 logic formulae were authored, in which twelve formulae are soft constraints and only one formula is a hard constraint. Equations (3.5) – (3.8) are four example logic formulae that Ha et al. (2011) investigated. Among these, Equations (3.5) – (3.7) are soft-constraint logic formulae that are not necessarily true from the observations. Equation (3.5) specifies a relation between actions and goals. In this formula, given an action \(a\) at time \(t\), the player’s goal at the time \(t\) will be \(g\). However, a player action is not necessarily mapped to a single goal; for example, a \textit{move} action can be observed in achieving multiple different goals while the goal that the player is attempting to achieve gets clarified by a series of following actions. Since this characteristic allows many-to-many mappings between actions and goals, this formula is better represented as a soft constraint characterized by a weight onto the logic formula. In addition to the twelve soft constraints, the MLN model considered one hard constraint logic formula described in (3.8). As opposed to soft constraints, this hard constraint asserts that there should exist exactly one goal per action at any given time (Ha et al., 2011).
∀t, a, g : action(t, a) ⇒ goal(t, g) \quad (3.5)

∀t, a, s, g : action(t, a) ∧ state(t, s) ⇒ goal(t, g) \quad (3.6)

∀t, a_1, a_2, g : action(t, a_1) ∧ action(t - 1, a_2) ⇒ goal(t, g) \quad (3.7)

∀t, a : action(t, a) ⇒ \exists g : goal(t, g) = 1 \quad (3.8)

The authors utilized Markov: The Beast, an off-the-shelf tool for MLNs (Riedel, 2005) for training the weights for the soft constraints. The tool uses the Cutting Plane Inference algorithm that performs maximum a posteriori inference for Markov logic networks. Figure 3.3 describes a part of the undirected graphical model induced by the logic formulae used in the MLN, wherein connected nodes indicate that the two associated predicates appear together in one or more logic formulae.

Figure 3.3: Graphical representation of the Markov logic network (Ha et al., 2011).

An empirical evaluation demonstrated that the MLN-based goal recognition approach holds potential in recognizing players’ goals in a non-linear game environment with multiple solution paths. The proposed MLN model along with the two n-gram models and majority baseline were evaluated with a 10-fold cross validation on the testbed data set. The MLN model scored 48.4% in the accuracy rate, significantly outperforming the non-trivial n-gram
baselines (unigram: 39.6%, bigram: 33.0%) as well as the majority selection baseline (26.6%). The unigram model outperforming the bigram model was echoed in this work on a different data corpus, which can be partially explained by that data sparsity issue deteriorates the bigram model’s performance. A one-way repeated-measures ANOVA and a Tukey post-hoc test demonstrated that the difference in the all pairs of the investigated models were statistically significant ($p < .01$) (Ha et al., 2011).

Extending Ha et al.’s work, Baikadi et al. examined a MLN-based goal recognition approach enriched with discovery events, domain-specific representations of user progress in digital games (Baikadi et al., 2014). Discovery events capture a set of major plot milestones required to achieve the mission in the game. Baikadi and colleagues designed six discovery events and utilized them in addition to a subset of the baseline logic formulae (e.g., Equation 3.5 – 3.8) to induce a domain-specific implementation of MLNs.

The discover-event based approach extended the previous work in two folds. First, these discovery-event formulae defined a special case of predicates such as $\text{action}(t, \text{“Test”})$ in addition to maintaining general cases of predicates described in symbolic representations (e.g., $\text{action}(t,a)$). This domain-specific information associated with key milestones in the testbed application holds capacity to improve the model’s predictive performance. Second, this work extended the scope of previous actions from a single action taken at time $t-1$ to all previous actions ($t_1 < t$) before the goal at time $t$, which enabled the logic to capture a broader range of events observed in the game trace data.
The empirical evaluation of the discovery-event based approach demonstrated that logical encodings of domain-specific discovery events improved goal recognition model performance with respect to both the predictive accuracy and efficiency. The discovery-event models were able to make accurate predictions earlier in the observation sequences relative to other baseline approaches including the baseline MLN models, when measured using the classic convergence-based metrics (Blaylock & Allen, 2003).

3.2.1.3 *N*-gram encoded Feedforward Neural Networks

Min and colleagues examined *n*-gram encoded feedforward neural networks for goal recognition in an open-world digital game (Min, Ha, et al., 2014), where the models were initialized through a greedy layer-by-layer pre-training technique based on stacked denoising autoencoders (Vincent et al., 2010) (see Section 2.2.2). In this section, the model description centers on the architecture of *n*-gram encoded feedforward neural networks.

This work specifically took two salient features of open-world digital games into consideration for goal modeling. First, goals are dependent; some goals are completed in rapid succession to other goals, and a pair of goals is more likely to occur subsequently than other pairs. This is likely attributed to the geographical proximity in open-world games, where some goals are more closely located than others. Second, there exist *sequential patterns* as well as *cyclical causality* between player actions and goals; players can learn about goals through a *sequence of interactions* (e.g., past action sequences, a history of achieved goals) with the virtual environment. These correlations among goals and actions inspired the authors to investigate *n*-gram encoded neural networks (especially when *n* > 1),
in which \((n-1)\) previous actions and \((n-1)\) previously achieved goals are considered along with the current action in order to recognize the goal associated with the current action.

Figure 3.4: \(N\)-gram encoded feedforward neural network (Min, Ha, et al., 2014).

An \(n\)-gram encoded feedforward neural network (Figure 3.4) is a fully connected, dense feedforward neural network. In this figure, \(Type\), \(Loc\), \(N.S.\), and \(P.G.\) denote the action type, location, narrative state, and previously achieve goal, respectively. The model takes as input the latest \(n\) actions (red dashed box in Figure 3.4) from the action \(A_{t-(n-1)}\) at time \(t-(n-1)\) to the current action \(A_t\) at time \(t\), along with \((n-1)\) previously achieved goals (blue dashed box in Figure 3.4), in which \(A_t^{P.G.-1}\) and \(A_t^{P.G.-*(n-1)}\) denote the last achieved goal and \((n-1)\)^th-to-the-last achieved goal, respectively.
For an illustration purpose, Figure 3.4 describes each action property at a time as an input unit; however, for an application of this method, each action property should be represented with a multidimensional vector represented in an one-hot vector (also called 1-of-\(N\) vector) or a distributed representation-based vector (Hinton, 1986; Mikolov, Yih, et al., 2013). If the one-hot encoding approach is leveraged for an action type, a bit vector whose length is the number of all possible action types is created, where only the associated action type bit is on (i.e., 1) while all other bits are off (i.e., 0). For example, if a player demonstrated a Move action type where three action types including Move, Talk, and Pick Up are available, the action type is represented with the vector of (1, 0, 0) (i.e., three input neurons), while (0, 1, 0) is created when she performed Talk.

It should be noted that the previously achieved goals do not mean the last achieved goal at time \(t\), \(t-1\), \(t-2\), and so on, because this framing of previously achieved goals would result in a list full of the same goal (it is very likely the lastly achieved goal at \(t\), \(t-1\), and \(t-2\) are all same); rather, \((n-1)\) previously achieved goals mean the last achieved goal, the second-to-the-last achieved goal, and so on. So, these goals indicate \((n-1)\) distinct goals achieved so far in the game. In cases that there are fewer than \((n-1)\) goals achieved at some time, the input units for the corresponding missing goals can be fed with 0s.

One should specify a set of hyperparameters prior to training the \(n\)-gram encoded feedforward neural networks. These include hyperparameters related to (A) optimization such as initial learning rate, learning rate schedule (e.g., adaptive learning rate), momentum, mini-batch size, and the number of epochs, (B) model structure such as the number of
hidden units, the number of hidden layers, and initialized weights, and (C) **training criterion** such as regularization terms (e.g., weight decay), activation functions, and loss functions, among others (Bengio, 2012). Furthermore, *n*-gram encoded feedforward neural networks pre-trained with stacked denoising autoencoders take two model-specific hyperparameters into consideration: *n* in the *n*-gram setting (i.e., how many actions to consider as input; thus model structure-based hyperparameter) and the corruption level attributed to stacked denoising autoencoders (i.e., how much portion of the input neurons are corrupted; thus, training criterion-based hyperparameter) (Vincent et al., 2010).

Selecting an effective network topology (i.e., hyperparameters) for feedforward neural networks is often empirically determined by selecting a model with the minimum validation error among multiple variants of models in a grid or random search setting (Bengio, 2012; Bergstra & Bengio, 2012), or determined by model-based optimization methods, such as Bayesian optimization (Snoek et al., 2015) and sequential model-based optimization (Hutter, Hoos, & Leyton-Brown, 2011).

Previous work showed that this neural network-based approach significantly outperforms both the Markov logic network and Bayesian network-based goal recognition approaches in a goal recognition task, even with reduced featuring engineering efforts (Min, Ha, et al., 2014). The following section describes multidimensional evaluation metrics supported in GOALIE, in order to robustly measure goal recognition models’ performance from various angles and identify reliable ones among candidate models.
3.2.2 GOALIE: Goal Recognition Model Evaluation Metrics

A standard framework and set of metrics for goal recognition has not been established (Sukthankar et al., 2014). A commonly used approach measures models’ predictive performance, such as their accuracy rate, precision, recall, and F1 (Baikadi et al., 2014; Ha et al., 2011). As a supplementary set of metrics, Blaylock and Allen (2003) proposed convergence-based metrics that capture models’ early prediction capacity, which has been investigated in a wide range of goal recognition work (Baikadi et al., 2014; Mott et al., 2006; Yin, Chai, & Yang, 2004). Early prediction is of significant importance for goal recognition because goal recognizers that lack an early prediction capacity cannot effectively support goal-driven adaptation of gameplay.

The metric of convergence point (Blaylock & Allen, 2003) measures how early goal recognition models can consistently make accurate predictions within a converged sequence, i.e., an action sequence in which the last goal prediction is correct. More formally, convergence point is calculated by \[ \frac{\sum_{i=1}^{m} (k_i/n_i)}{m}, \] in which \( m \) is the number of converged action sequences, and \( n_i \) and \( k_i \) are the total number of actions and the number of actions after which the goal recognizer consistently makes accurate predictions in the \( i^{th} \) converged action sequence, respectively (Min, Mott, et al., 2016). Note that convergence point ignores all action sequences that do not converge to a correct goal.

In this section, we discuss the convergence point metric, which provides misleading results with respect to early prediction in some cases (Min, Baikadi, et al., 2016). For
example, suppose that we have two goal recognition models: GR₁ and GR₂, and there are two action sequences (AS₁ and AS₂) to predict the following:

- **AS₁**: Action₁₁, Action₁₂, and Action₁₃, whose goal is G_A.
- **AS₂**: Action₂₁, Action₂₂, and Action₂₃, whose goal is G_B.

In this situation, assume that both GR₁ and GR₂ correctly predict the goal (G_A) associated with the three actions in AS₁, while GR₁ makes incorrect predictions on all three actions’ of AS₂, but GR₂ makes correct predictions for the goal of the last two actions (Action₂₂ and Action₂₃). The convergence point of GR₁ is 0.33 since it consistently makes correct predictions after observing the first action and AS₁ is the only converged sequence. However, the convergence point of GR₂ is 0.5 which is computed as (1/3+2/3)/2, and thus GR₁ is identified as the model with better early prediction capacity based on convergence point (lower is better for this metric) even though GR₂ is more reliable than GR₁ since it accurately predicts more actions (and earlier) in both sequences. We introduce **standardized convergence point** to overcome this issue with the conventional converge point metric.

### 3.2.2.1 Standardized Convergence Point

The standardized convergence point metric measures the convergence point regardless of whether an action sequence converged to a correct goal or not. To compute this metric, a non-converged action sequence has a convergence point of (the total number of actions + p) divided by (the total number of actions) in which p is greater than 0, thereby yielding a convergence point higher than 1. In this manner, non-converged action sequences are penalized in terms of early prediction.
Definition 1. Standardized convergence point is calculated by \( \frac{\sum_{i=1}^{m} (k_i/n_i)}{m} \), in which \( m \) is the total number of action sequences, and \( n_i \) is the total number of actions in the \( i^{th} \) action sequence. \( k_i \) is contingent on whether the \( i^{th} \) action sequence converged or not; if converged, \( k_i \) is the number of actions after which the goal recognizer consistently makes accurate predictions; otherwise, \( k_i \) is \( n_i + p \), where \( p (p > 0) \) is a penalty parameter. Lower is better for this metric.

In definition 1, \( p \) is a settable parameter. \( p \) can be either a static value or a dynamic value (e.g., a multiple of the length of each action sequence). Returning to the example presented in the preceding section (we set \( p \) to 1 in this dissertation), standardized convergence point of GR\(_1\) is 0.83, computed by \((1/3+4/3)/2\), while standardized convergence point of GR\(_2\) remains the same as 0.5, computed by \((1/3+2/3)/2\). Thus, using this definition, the standardized convergence point of GR\(_2\) is lower (i.e., better) than GR\(_1\). Models’ early prediction is better captured with the standardized convergence point metric than the conventional convergence point, and GOALIE measures models’ early prediction capacity with this metric.

3.2.2.2 N-Early Convergence Rate

Convergence rate (Blaylock & Allen, 2003) measures the percentage of action sequences in which the last goal prediction is correct. While the goal recognition system generally makes correct predictions on individual actions (i.e., high accuracy rate), if it makes an incorrect prediction on the last action (i.e., low convergence rate), the model lacks reliability since incorrect game adaptation at the end of an action sequence may result in confusion to players
who are about to achieve the planned goal. We extend this metric to consider the last \((n+1)\) action predictions through a new metric, \(n\)-early convergence rate.

**Definition 2.** \(N\)-early convergence rate is calculated by \(\sum_{i=1}^{m} k_i / m\), in which \(m\) is the total number of action sequences. \(k_i\) is 1 if the last \((n+1)\) goal predictions are all correct in the \(i^{th}\) action sequence; otherwise, \(k_i\) is 0 for the action sequence. In a special case of this metric when \(n\) equals 0, the definition of \(n\)-early convergence rate is the same as the conventional convergence rate. Higher is better for this metric.

\(N\)-early convergence rate views goal recognizers’ early prediction from a different perspective compared to standardized convergence point. The \(n\)-early convergence rate takes a static approach (i.e., a fixed window size of \(n+1\)) backward from the end of an action sequence, while standardized convergence point takes a dynamic approach forward from the beginning of an action sequence. These two novel measurements are designed to complement the corresponding conventional convergence metrics when evaluating goal recognition models’ early prediction capacity.

It should be noted that neither of these novel convergence metrics nor the conventional convergence metrics could be dynamically computed during gameplay since players’ goals are hidden from the goal recognizer. Nonetheless, offline measurements of these metrics offer considerable insight into selecting the most reliable goal recognition model to best support runtime game adaptation.
CHAPTER 4

GOAL RECOGNITION DATA CORPORA

This chapter presents the two testbed games and data corpora induced from player interactions with the two games. Below, we describe the two educational games. Section 4.1 presents CRYSTAL ISLAND: OUTBREAK, which focuses on middle-grade microbiology learning, and Section 4.2 describes CRYSTAL ISLAND: UNCHARTED DISCOVERY, which focuses on upper elementary science learning. Both the digital games are discussed with respect to backstory, gameplay, and user studies.

Generalizability of GOALIE is demonstrated through the investigation of two different open-world games. As discussed, goal recognition models are devised in a data-driven approach utilizing machine learning techniques, where each data corpus should be first encoded into a format that machine learning algorithms can interpret and take as input or output. To address this, in the both datasets, the input features are encoded based on an observed action (or observed action sequences depending on the model formalism) that consists of four action properties: action type, locations, narrative state, and goals previously achieved within the environment (see Section 3.2.1), and the output features represent the goal that the player is currently achieving using a sequence of actions. This encoding method is identically applied to two different games.

In the both open-world educational games, players’ action sequences do not necessarily represent optimal paths for achieving goals in both games since the players did not have prior experience with the games. Rather, action sequences are often sub-optimal and
noisy; players explore the virtual environment in order to familiarize themselves with the
gameworld and often do not utilize the most efficient problem-solving strategies available.
Players make gradual but possibly circuitous progress toward each objective, eventually
culminating in the final action that achieves a goal (Min, Mott, et al., 2016).

4.1 **CRYSTAL ISLAND: OUTBREAK**

**CRYSTAL ISLAND: OUTBREAK** is a rich, virtual 3D digital game implemented using the Source
game engine by Valve\(^1\), where players learn microbiology concepts, which is grounded in
North Carolina standard course of study for eight-grade microbiology, through interactive
science narratives. **CRYSTAL ISLAND: OUTBREAK** has been the subject of extensive empirical
investigation, and has been found to provide substantial learning and motivational benefits
(Rowe et al., 2011), while also offering significant challenge with fewer than half of players
solving the mystery in less than an hour.

4.1.1 **Backstory and Game Play**

**CRYSTAL ISLAND: OUTBREAK** (Figure 4.1) features a science mystery where players attempt
to discover the identity and source of an infectious disease that is plaguing a research team
stationed on a remote island. Players play the role of a visiting investigator to the island, who
is drawn into a mission to save the research team from the outbreak. Players explore the
research camp from a first-person viewpoint and manipulate virtual objects, converse with
virtual characters, and use lab equipment and other resources to solve the mystery. Player
microbiology scaffolding is delivered in various formats throughout the environment, such as

\(^1\) [http://www.valvesoftware.com/](http://www.valvesoftware.com/)
conversation with non-player characters, viewing posters and books to learn about infectious agents, and performing hypothesis testing using lab equipment. Players record their findings, hypotheses, and a final diagnosis in the diagnosis worksheet, and solve the mystery by submitting the correct worksheet to the camp nurse.

Figure 4.1: CRYSTAL ISLAND: OUTBREAK educational game.

Players advance through CRYSTAL ISLAND: OUTBREAK’s non-linear narratives by completing a partially ordered sequence of goals. In this work, seven goals are considered: speaking with the camp nurse about the spreading illness, speaking with the camp’s virus expert, speaking with the camp’s bacteria expert, speaking with a sick patient, speaking with the camp’s cook about recently eaten food, running laboratory tests on contaminated food, and submitting a complete diagnosis to the camp nurse.

Players interact with CRYSTAL ISLAND: OUTBREAK using a diverse set of actions occurring in the seven major locations of the research camp (Figure 4.2): a large outdoor region, an infirmary, a living quarters, a waterfall, the lead scientist’s quarters, a dining hall, and a laboratory. Players can perform actions that include: moving around the camp, picking
up and dropping objects, using the laboratory’s testing equipment, conversing with virtual characters, reading microbiology-themed books and posters, completing a diagnosis worksheet, labeling microscope slides, and taking notes.

Figure 4.2: Map of the Crystal Island: Outbreak research camp.

4.1.2 Study Description

All player actions are logged by the Crystal Island: Outbreak learning environment and stored for later analysis. The data to be used for creating the goal recognition models was collected from a study involving 153 eighth grade student players, aged 12–15 (M=13.3, SD=0.48) in Wake County public middle schools. 16 players were removed from the analysis due to incomplete data or prior experience with Crystal Island: Outbreak, and thus 137 players (males: 77, female: 60) will be used to evaluate the goal recognition framework (Baikadi, 2014; Ha et al., 2011; Min, Ha, et al., 2014; Min, Mott, et al., 2016).
Prior to starting the game, the players were informed the backstory and overall descriptions of the game, such as the list of non-player characters and the island map (Figure 4.2) along with game mechanics required to control the player character. The players conducted a pre-survey measuring their pre-content knowledge before the intervention. The game play lasted for 60 minutes at maximum, and immediately after interactions with the game, the players conducted a post-survey. More details about the study instrument can be found in the Rowe et al.’s literature (2011).

4.1.3 Corpus

Similar to previous work on goal recognition (Blaylock & Allen, 2003), we define goal recognition as the task of predicting the most likely goal for a given sequence of observed player actions in the environment. In Crystal Island: Outbreak, goals are not directly presented to players, and thus should be inferred based on observable evidence that players revealed during interactions with the game. The dissertation assumes that a given sequence of actions maps to a single goal, and no interleaving occurs between actions associated with different goals, since our existing dataset does not lend itself to this type of analysis. Under these conditions, goal recognition is cast as a multiclass classification problem in which a learned classifier predicts the most likely goal associated with the currently observed sequence of actions after the previously observed goal. The Crystal Island: Outbreak corpus includes 137 students’ gameplay data that consists of 77,182 player actions (i.e., the total number of possible goal recognitions) and 893 achieved goals, with an average of 86.4 player actions per goal (Ha et al., 2011).
**Goal Recognition Input.** A key step that must precede the model-training step is to encode data in the format that machine-learning algorithms can take as input (i.e., features) and output (i.e., labels). According to the GOALIE’s action encoding method described in Section 3.2.1, a player action is encoded with four properties: action type, location, narrative state, and previously achieved goals.

- **Action Type:** CRYSTAL ISLAND: OUTBREAK includes 19 distinct types of player actions.
- **Location:** CRYSTAL ISLAND: OUTBREAK includes 39 fine-grained and non-overlapping sublocations that decompose the seven major camp locations.
- **Narrative State:** The four milestone events are discussing the illness with the nurse, testing the contaminated object, submitting a diagnosis to the nurse, and submitting a correct diagnosis to the nurse.
- **Previously Achieved Goals:** The eight previously achieved goals are available, including ‘None’ in case the player has not yet achieved any goals.

**Goal Recognition Output.** Goal recognition in CRYSTAL ISLAND: OUTBREAK defines seven goals (i.e., seven class labels), where each goal represents a key high-level objective collectively required to achieve the final mystery situated in the game. Descriptive statistics of the distribution of the seven goals are shown in Table 4.1. The majority class-based accuracy rate is 26.6%: “Running laboratory test on contaminated food”.

Once a goal is achieved, any future occurrence of the goal-achieving action is considered as a plain action rather than a goal, because players already know how to achieve the goal and might have taken the action as a tool to achieve a new goal. Along this line, it
should be noted that the distribution in Table 4.1 does not mean a specific goal is more often achieved than the others; rather, it indicates that the specific goal required more actions to get achieved, because a training data sample is generated per player action.

Table 4.1: **Outbreak**: Distributions of goals (Ha et al., 2011)

<table>
<thead>
<tr>
<th>Goals</th>
<th>Distributions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running laboratory test on contaminated food</td>
<td>26.6%</td>
</tr>
<tr>
<td>Submitting a diagnosis</td>
<td>17.1%</td>
</tr>
<tr>
<td>Speaking with the camp’s cook</td>
<td>15.2%</td>
</tr>
<tr>
<td>Speaking with the camp’s bacteria expert</td>
<td>12.5%</td>
</tr>
<tr>
<td>Speaking with the camp’s virus expert</td>
<td>11.2%</td>
</tr>
<tr>
<td>Speaking with a sick patient</td>
<td>11.0%</td>
</tr>
<tr>
<td>Speaking with the camp nurse</td>
<td>6.4%</td>
</tr>
</tbody>
</table>

4.2 **Crystal Island: Uncharted Discovery**

To evaluate generalizability of GOALIE, another digital game called **Crystal Island: Uncharted Discovery** (Figure 4.3) is analyzed. **Crystal Island: Uncharted Discovery** is an educational game for upper elementary science, which is designed to engage students in problem solving and science (Lester et al., 2014). Its curricular focuses center on landforms with an emphasis on the map interpretation and navigation skills, which include the identification of common landforms, introduction of modeling objects (e.g., maps are a portable model representing land forms and structures), and interpreting maps for navigation (Lester et al., 2014).
Approximately eight hundred students participated in the study spanned over four weeks. The study was conducted in multiple sessions of a teacher-led deployment of CRYSTAL ISLAND: UNCHARTED DISCOVERY and classroom studies that provide supplemental lessons to facilitate students’ learning. Throughout this study, the participants achieved significantly improved content test scores from pre (M=11.84, SD=4.05) to post (M=13.63, SD=3.65), and also significant gains in problem-solving skills ($t(713)=3.72$, $p<.01$) (Lester et al., 2014).

4.2.1 Backstory and Game Play

CRYSTAL ISLAND: UNCHARTED DISCOVERY is set on a fictional island in the Oceania region of the Pacific Ocean, where the explorers including the student player were shipwrecked by a devastating storm and landed on the island. After the player embarks on exploring the newly discovered island, she meets non-player characters such as the mayor of the village, initiates
a conversation with them, and solves quests asked by the village’s teacher. Conversation
dialogues take place through an interactive way in which non-player characters’ utterances
are delivered on the screen and players select their utterances from a menu.

The student plays the role of a navigator from a third-person viewpoint, using a
virtual tablet (Figure 4.4), and solving quests situated on the island. A set of quests is
designed to enhance students’ landform understanding, map interpretation, navigation ability,
and problem-solving skills (Lester et al., 2014). The virtual tablet, a multi-functional device
containing several built-in apps, is introduced to provide problem-solving scaffolding and
create engaging learning experiences. The virtual tablet includes the IslandPedia app
(Figure 4.4, Left) and the Problem-Solving app (Figure 4.4, Right). IslandPedia presents key
concepts such as landforms, map navigation and scale, and problem-solving process, through
a visual presentation and voiceover narration. The problem-solving app dynamically guides
students’ problem-solving activities. The app is grounded in the Polya problem-solving
framework featuring four basic principles: understanding the problem, devising a plan,
carrying out the plan, and looking back (Polya, 2014). The other apps available in CRYSTAL
ISLAND: UNCHARTED DISCOVERY is described in depth in Lester et al.’s work (Lester et al.,
2014).
4.2.2 Study Description

Over a four-week period, approximately eight hundred participants were involved in the study across the eight participating elementary schools. The proportion of male to female students was 49:51, and the ethnicity was 62% Caucasian, 14% African American, 8% Asian and other. The schools are at diverse locations: 40% of urban, 20% of suburban, and 40% of rural settings. The entire study was conducted in twelve 50-minute class periods over the four-week period, which consists of 50% of game play activities (teacher-driven implementation of CRYSTAL ISLAND: UNCHARTED DISCOVERY) and another 50% of classroom activities (supplemental lessons developed by teachers to enhance the learning experience). Students were allowed to play the game inside and outside of class, and the optimal CRYSTAL ISLAND: UNCHARTED DISCOVERY experience could take place over approximately 800 min (Lester et al., 2014). Students’ game trace was logged in a remote database.
4.2.3 Corpus

Baikadi formulated a goal recognition task in *CRYSTAL ISLAND: UNCHARTED DISCOVERY* into predicting goals in each quest (Baikadi, 2014). Following Baikadi’s initial work, the dissertation considers the data corpus generated from the first two-week sessions. During the period, players encountered four quests (Table 4.2), while the tutorial interaction data was removed from the analysis, because it contains a pre-defined set of interactions. The two quests are about landform identification tasks, such as plateaus, dams and waterfalls while the other two quests involve map interpretation and orientation tasks using a map and heading on the island.

Table 4.2: *UNCHARTED DISCOVERY*: Quest descriptions (Baikadi, 2014)

<table>
<thead>
<tr>
<th>Quest</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Landform Identification</td>
<td>The player is given three landforms drawn on a blackboard, and is asked to place a sign next to each one.</td>
</tr>
<tr>
<td>Landform Photography</td>
<td>The player is given a list of three landforms, and is asked to take a photo of each one.</td>
</tr>
<tr>
<td>Map Navigation</td>
<td>The player is given three map cells, and asked to pick up the flag at each location.</td>
</tr>
<tr>
<td>Orienteering</td>
<td>The player is given directions and scales to three locations and is asked to take a photo of the animal located there.</td>
</tr>
</tbody>
</table>

Once a quest has begun, the quest presents three goals that are not inherently constrained in any order but can be dynamically achieved based on students’ plans and goals.
due to the digital game’s open-world characteristic. Further, because players also can initiate multiple quests at the same time, the total number of possible goals is 12.

The Crystal Island: Uncharted Discovery dataset used in the evaluation of our goal recognition work involves 828 fifth grade students from eight public elementary schools. The dataset consists of 811,647 player actions and 7,652 achieved goals, with an average of 106.1 player actions per goal.

Goal Recognition Input. As in Crystal Island: Outbreak, raw game trace data are encoded into a machine-trainable format, following the procedure described in GOALIE.

- **Action Type:** Action types include 126 distinct types of low-level actions (the low-level action types are the ones that are actually utilized in the model) under 19 high-level action categories (the action categories are just for an illustration of the low-level actions, and are not directly used in the model). Examples of the high-level action categories include “open virtual tablet” and “change tablet app” associated with the virtual tablet, and “see dialogue” through which the player views a menu-based dialogue to interact with the non-player characters in the virtual environment.

- **Location:** Location includes 84 non-overlapping sub-locations within the gameworld. Unlike Crystal Island: Outbreak, the location is designed based on physical grid cells that decompose the entire island without leveraging semantic geographical information.

- **Narrative State:** Narrative states contain 16 possible values based on the interactive storyline’s plot structure. The four quests are considered as milestone plot events following Baikadi’s work (2014), which are landform identification, landform photography, map
navigation, and orienteering. Since multiple quests can be simultaneously running, the total number of possible narrative states is 16, including ‘None’.

• **Previously Achieved Goals:** There are 13 possible goals that could be previously achieved, including ‘None’ in case the player has not yet achieved any goals.

**Goal Recognition Output.** As in the *CRYSTAL ISLAND: OUTBREAK* goal recognition formulation, each observation is associated with the next goal completed, and interleaved goals (i.e., a sequence of actions during a period are taken with an aim to achieve multiple goals) or concurrent goals (i.e., a goal is the sub-goal of another higher-level goal) are not considered. Two types of actions are discarded from the corpus: 1) actions taken after the last goal completed were removed, because actions after the final goal are not likely to account for a specific goal, and 2) goal-achieving actions are discarded, because it would be trivial to predict goals from the goal-achieving actions (i.e., goal-achieving actions are goals). Lastly, as in the *OUTBREAK* dataset, once a goal is achieved, any future occurrence of the goal (i.e., goal-achieving action) is considered as a plain action rather than a goal.

Table 4.3 describes the twelve goals extracted from the 4 quests. The most frequent goal (“Picking up the dark blue flag”) and the least frequent goal (“Placing sign at the volcano”) appeared 13.5% (i.e., majority class baseline) and 3.1% out of the entire set of players’ achieved goals, respectively.
Table 4.3: UNCHARTED DISCOVERY: Distribution of goals

<table>
<thead>
<tr>
<th>Goals</th>
<th>Distributions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Photograph Animal at 175E 25N</td>
<td>9.66%</td>
</tr>
<tr>
<td><strong>Pick up Dark Blue Flag</strong></td>
<td><strong>13.53%</strong></td>
</tr>
<tr>
<td>Photograph Lake</td>
<td>8.13%</td>
</tr>
<tr>
<td>Pick up Burgundy Flag</td>
<td>5.81%</td>
</tr>
<tr>
<td>Photograph Animal at 75E</td>
<td>11.25%</td>
</tr>
<tr>
<td>Photograph Animal at 200W 75S</td>
<td>11.66%</td>
</tr>
<tr>
<td>Place Sign at Waterfall</td>
<td>3.67%</td>
</tr>
<tr>
<td>Pick up Green Flag</td>
<td>5.58%</td>
</tr>
<tr>
<td>Place Sign at Plateau</td>
<td>10.74%</td>
</tr>
<tr>
<td>Photograph Delta</td>
<td>9.22%</td>
</tr>
<tr>
<td>Place Sign at Volcano</td>
<td>3.14%</td>
</tr>
<tr>
<td>Photograph Tributary</td>
<td>7.61%</td>
</tr>
</tbody>
</table>
CHAPTER 5
SEQUENCE MODEL-BASED GOAL RECOGNITION

To implement the core goal recognition functionality that predicts players’ dynamically changing goals and informs run-time adaptation of gameplay in open-world digital games, game developers should explore multiple goal modeling techniques and find one that is most effective for gameplay. Reliable goal recognition models must not only predict goals accurately in the face of highly noisy sequences of low-level player actions, but also make fast and consistently correct recognitions as early as possible, only relying on an early portion of an action sequence.

Due to the exploratory nature of player behavior in open-world digital games, goal recognition models should robustly handle cyclical relationships between player goals and actions (Ha et al., 2011; Min, Mott, et al., 2016). Players’ previously achieved goals may inform their subsequent actions, and their current actions may influence their upcoming goals. Consequently, extracting patterns from sequences of player actions and goals is likely to provide strong evidence to predict the next high-level objective that a player will achieve.

These characteristics of open-world digital games have inspired the investigation of a set of machine learning techniques for goal recognition. Ha and colleagues (2011) investigated Markov logic networks (MLNs) grounded in a statistical relational learning framework (Richardson & Domingos, 2006). MLNs are well suited for machine learning tasks in domains with complex associations between modeled entities such as actions and goals in goal recognition. Min et al. (2014) examined n-gram encoded feedforward neural
networks, where previous actions and previously achieved goals are encoded in the feature set and are jointly utilized to predict the player’s current goal.

In contrast to these two methods that formulate the sequential and cyclical relationships between actions and goals using a fixed length of inputs and outputs, sequence labeling techniques can take variable length inputs and outputs without constraining them to a fixed size. This flexible modeling capacity featured in sequence labeling techniques is well suited for capturing sequential, complex patterns across players’ previous behavior, achieved goals, and the current goal, and has motivated the investigation of goal recognition using sequential supervised learning methods such as long short-term memory networks (LSTMs) (Hochreiter & Schmidhuber, 1997) and linear-chain conditional random fields (CRFs) (Sutton & McCallum, 2012).

The remainder of this chapter is organized as follows: Section 5.1 describes two vector-based representation methods for player actions, *one-hot encoding* and *distributed representation*, both of which are applicable to various machine learning techniques. Sections 5.2 and 5.3 delineate details about LSTM and CRF-based goal recognition models in the context of the two testbed open-world digital games, respectively.

### 5.1 Data Representation Methods: One-Hot Encoding and Distributed Representations

Generating appropriate representations for training data is an important pre-processing step for various computational modeling tasks. The decision about which representation technique to use depends on the characteristics of the data; some sources of data consist of categorical variables such as nominal (e.g., genders, days) or ordinal (e.g., academic grades) variables;
some other data have continuous variables such as numeric variables (e.g., time, pixel values in an image) and discrete variables (the number of household members). While general probabilistic graphical models handle the categorical variables by inducing probability distribution tables per node, some machine learning techniques such as artificial neural networks utilize continuous data representations due to the capability of interpreting magnitude in each variable.

An interesting line of research about the data representation has been conducted in natural language processing (NLP). Much of classical NLP work has utilized a simple representation method, called one-hot encoding, which converts a discrete identifier of a nominal lexical unit (hereafter, words are used as an example) to a discrete, multidimensional vector (Turian, Ratinov, & Bengio, 2010). One-hot encoding creates a bit vector whose length is the size of the vocabulary of words, where only the associated word bit is on (i.e., 1) while all other bits are off (i.e., 0). Vectors induced by one-hot encoding are inherently sparse (i.e., zeros for all features but one in a vector) and high-dimensional (i.e., the size of a vector equals the size of the vocabulary), thereby easily causing the curse of dimensionality problem, as the size of vocabulary and length of sequence increase.

More recently, researchers in the NLP domain have examined distributed representations (Hinton, 1986) of words, called word embeddings (Bengio et al., 2003; Collobert & Weston, 2008; Mikolov, Chen, et al., 2013; Turian et al., 2010). The word embeddings are machine-learned often by embedding a projection layer (i.e., a linear layer that maps a discrete word index onto a continuous, multidimensional vector space using a
shared word embedding matrix) within a neural network architecture such as feedforward neural networks (Bengio et al., 2003), convolutional neural networks (Collobert & Weston, 2008), simple recurrent neural networks (Mikolov, Yih, et al., 2013), recursive neural networks (Socher et al., 2013), and long short-term memory networks (Min & Mott, 2015). Neural networks that utilize distributed lexical unit embeddings have demonstrated significant success in a wide range of NLP tasks such as language modeling (Bengio et al., 2003; Mikolov, Yih, et al., 2013), sentiment analysis (Maas et al., 2011), paraphrase detection (Socher, Huang, & Pennington, 2011), language parsing (Socher et al., 2010), text normalization (Min & Mott, 2015), and multiple tasks (e.g., part-of-speech tagging, chunking, named entity recognition) in a unified architecture (Collobert & Weston, 2008).

In addition to the improved performance, the distributed representation learning method has representational and computational benefits over the one-hot encoding technique, since it effectively represents data in a reduced feature space with dense, continuous values compared to one-hot encoding that induces sparse representations in a high-dimensional vector space (Leeman-Munk, 2015). Moreover, distributed representations of words have demonstrated remarkable success in both semantic (e.g., queen – king = female – male) and syntactic word analysis tasks (e.g., go – went = sing – sang) (Mikolov, Yih, et al., 2013), by effectively capturing linguistic regularities across words. In contrast to the distributed representation method, one-hot representations do not capture these relationships, since every pair of two different words has the same distance (i.e., $\sqrt{2}$ Euclidean distance).
Notably, Le and Mikolov (2014) have shown that distributed representations that were randomly initialized can eventually capture salient patterns characterized in inputs (e.g., semantic and syntactic information of lexical units) as an indirect result of the supervised learning task. Additionally, Collobert and colleagues (2011) have demonstrated that discrete features of words defined in a dictionary can be represented in a continuous, dense vector space through a training process of deep neural networks and these low-level distributed representations along with higher level features can synergistically improve the performance of sequence labeling tasks such as part-of-speech tagging, named entity recognition and semantic role labeling. Significant successes exhibited in a wide range of computational NLP tasks by leveraging distributed representation learning techniques have inspired our investigation of distributed representations for the goal recognition task.

In the next two following sections, we formulate player goal recognition as a sequence-labeling task and present two sequence labeling techniques, long short-term memory networks and linear-chain conditional random fields, for devising sequential goal recognition models. Especially, long short-term memory network-based goal recognition models incorporate a projection layer to train distributed representations of action properties, inspired by deep learning’s representation learning capability and its success in natural language processing.

5.2 Long Short-Term Memory Network-Based Goal Recognition

Long short-term memory networks (LSTMs) are a variant of recurrent neural networks (RNNs) that are specifically designed for sequence labeling of temporal data. While training
standard RNNs has faced challenges regarding the vanishing gradients when training unfolded, deep neural networks (Bengio et al., 1994; Pascanu, Mikolov, & Bengio, 2013), the three gating units (input gate, output gate, and forget gate) featured in LSTMs has enabled hidden units to adaptively maintain long-term memory by allowing gradient information to flow over many time steps (Chung, Gulcehre, Cho, & Bengio, 2015). LSTMs achieved high predictive performance in various sequence labeling tasks (Schmidhuber, 2014), often outperforming standard RNNs by leveraging a longer-term memory, preserving short-term lag capabilities, and effectively addressing the vanishing gradient problem (Graves, 2012).

Specifically, LSTMs are a type of *gated* recurrent neural networks. Since the initial version of LSTMs that feature an input gate and an output gate was proposed (Hochreiter & Schmidhuber, 1997), several variants of LSTMs have been introduced, including LSTMs with a forget gate (Gers et al., 2000) (see Section 2.2.3) and LSTMs with peephole connections (Gers, Schraudolph, & Schmidhuber, 2002). In LSTMs augmented by peephole connections, the memory state at the previous and current time step is utilized to calculate the input, forget, and output gates at the current time step, directly compared to the previous LSTM implementation. More recently, other gated recurrent neural networks were proposed, such as gated recurrent units featuring a reset gate and an update gate (Cho et al., 2014) and gated feedback recurrent neural networks using stacked RNNs featuring gated-feedback connections between different layers (Chung et al., 2015). In this dissertation, an LSTM implementation without peephole connections (Graves, 2012) is investigated. More details about this LSTM can be found in Section 2.2.3.
5.2.1 LSTM-Based Goal Recognition Model Architecture

Figure 5.1 depicts a conceptual illustration of LSTM-based goal recognition models. The model architecture is described in the bottom-up fashion (i.e., input layer to softmax layer). Each instance of training data consists of a sequence of actions and its goal label. For example, if three actions, $x_1$, $x_2$, and $x_3$, are taken to achieve the goal ($g$), three data examples are generated: (1) $[x_1]$ for $g$, (2) $[x_1, x_2]$ for $g$, and (3) $[x_1, x_2, x_3]$ for $g$. We set the maximum length of an action sequence to be under $k+1$ for training complexity, so if the length of an action sequence is greater $k$, then an action sequence consists of the last $k$ actions (in the dissertation, we use $k$ of 10) for the both corpora. The following paragraphs describe how an action (input) in an action sequence is processed to $x_t$ (Figure 5.1B) prior to being fed into the LSTM model (Figure 5.1A). Figure 5.1A follows the standard LSTM operation, and details of this step can be found in Section 2.2.3 or Grave’s work (Graves, 2012).

To represent an action input for the OUTBREAK and UNCHARTED DISCOVERY data corpora, we utilize a 10-dimensional discrete vector ($N=7$) and 15-dimensional discrete vector ($N=12$), respectively (Figure 5.1B). In the both cases, the first three dimensions of the vector are allocated to represent the action type, action location, and current narrative state with integer-based indices, while the following 7 (OUTBREAK) and 12 (UNCHARTED DISCOVERY) dimensions represent a sequence of previously achieved goals also with integer-based indices. Note that since (1) the goals are often achieved after many time steps (the average number of player actions per goal for OUTBREAK is 86.4, and UNCHARTED DISCOVERY is 106.1), and (2) action sequences longer than a threshold ($k=10$), as noted, are
In other words, if the model only considers the last achieved goal (i.e., $N=1$) per time step, the action sequence based on the last $k$ actions is...
likely to have the same last achieved goals across all the \( k \) actions within an action sequence, and therefore the input would not effectively capture a history of previously achieved goals.

As noted in Section 5.1, the distributed representation-based encoding has a representational benefit as well as a computational efficiency over the one-hot encoding method. In this goal recognition approach, we employ a projection layer (Figure 5.1B) that converts a discrete value of each action property: action type (\( \text{Type} \)), location (\( \text{Loc.} \)), narrative state (\( \text{N.S.} \)), previously achieved goals (\( \text{Goal 1 to Goal N} \)) to a continuous, \( d \)-dimensional vector. To operationalize this for the \textbf{OUTBREAK} dataset, a shared, comprehensive action-property embedding matrix (\( \text{EM} \)), the size of which is 74 (the total number of possible values of action properties computed as 19 action types + 39 locations + 8 narrative states + 8 previously achieved goals including \textit{None}) by \( d \) (embedding size), is created (Figure 5.1B). For \textbf{UNCHARTED DISCOVERY} dataset, the same process is applied, but a different size of \( \text{EM} \) is generated: 239 (126 action types + 84 locations + 16 narrative state + 13 previously achieved goals including \textit{None}) by \( d \) (embedding size).

For example, in \textbf{CRYSTAL ISLAND: OUTBREAK}, suppose that a player \textit{talks} (action type whose discrete value is 7) with the camp nurse at the \textit{infirmary} (action location whose discrete value is 30), while having just completed the narrative milestone of \textit{Submit a Diagnosis to the Camp Nurse} (narrative state whose discrete value is 60) and having achieved \textit{Speak with the Camp Cook about Recently Eaten Food} (previously achieved goal whose discrete value is 71). To simplify the illustration, we set \( N \) to 1, so that the model takes as input only the last achieved goal for previously achieved goals. The discrete representation
of the action in the input layer is represented with \( (7, 30, 60, 71) \). After finding the corresponding \( d \)-dimensional distributed representations for each value in the \( EM \), each of the four distributed representations (i.e., \( EM(7) \), \( EM(30) \), \( EM(60) \), and \( EM(71) \)) is projected on the projection layer, after which the four distributed representations are concatenated into a single vector, \( (EM(7), EM(30), EM(60), EM(71)) \). The length of the vector after the concatenation step is \( (N+3) \times d \) for the both data corpora.

The \( EM \) is learned in two different ways: (1) the matrix is randomly initialized following a uniform distribution (max: 0.05, min: -0.05) and is fine-tuned during supervised machine learning, and (2) the matrix is randomly initialized following the same uniform distribution and is pre-trained following a language modeling approach (Bengio et al., 2003) (i.e., predicting the next action based on a sequence of previous actions), and is then fine-tuned during supervised learning. Previous goal recognition work based on the OUTBREAK dataset (Min, Mott, et al., 2016) shows that these two different approaches do not constitute a significant difference, while the first approach without pre-training outperforms the other approach by the 0.09% point with respect to the predictive accuracy; thus, the dissertation investigates the “random initialization followed by supervised fine-tuning” approach to train the \( EM \).

At recognition time, a sequence of actions is sequentially fed into the LSTM model per time step in the recurrent neural network formalism. The memory cell state and output at the previous time step are used to compute the cell state and output at the current time step. The final memory cell output vector \( (h_t \) in Figure 5.1A) is used to predict the most likely goal
for the sequence of actions in a softmax layer as in Figure 2.5A, which is interpreted as a calculation of posterior probabilities of goals (Min, Mott, et al., 2016).

5.2.2 LSTM-Based Goal Recognition Model Configurations

As discussed in Section 3.2.1.2, model hyperparameters should be determined prior to training the model. For hyperparameter optimization of our LSTM-based goal recognition models, we adopt a grid search and empirically determine an optimal configuration of the networks through cross validation. For devising LSTM-based goal recognition models, we explore three hyperparameters: the size of the action property embedding among \{10, 20\}, the number of hidden units among \{100, 200\}, the dropout rate (Srivastava et al., 2014) among \{0.5, 0.75\}, all of which have significant potential to influence LSTMs’ predictive performance (Min, Mott, et al., 2016).

Other than these, we use a softmax layer for classifying given sequences of actions, adopt a mini-batch gradient descent with the mini-batch size of 128, and utilize categorical cross entropy for the loss function and the Adam stochastic optimizer (Kingma & Ba, 2015) for the both data corpora. For training efficiency, action sequences greater than ten are pruned to keep only the last ten actions. Finally, the training process stops early if the validation score has not improved within the last seven epochs. In this work, 10% of the training data is used to determine early stopping, while 90% is utilized for supervised training. The maximum number of epochs is set to 100.
For devising LSTM-based goal recognition models, we use Keras (Chollet, 2015), a python-based modular neural networks library, which utilizes a backend of Theano (Bastien et al., 2012).

5.3 Linear-Chain Conditional Random Field-Based Goal Recognition

Conditional random fields (CRFs) are discriminative models for structured prediction (Lafferty, McCallum, & Pereira, 2001). CRFs harness the graphical modeling capacity for multivariate data classifications, especially modeling interdependencies in predictive features (e.g., pixels in an image, words in a sentence) along with the class labels associated with the features. CRFs are regarded as a sequential extension of logistic regression models or a discriminative analog of hidden Markov models (Sutton & McCallum, 2012). While they have proven useful in various domains, recent research has achieved significant advances by incorporating CRFs with deep learning techniques to address computational challenges situated in natural language processing (e.g., text normalization (Chrupała, 2014)) and computer vision (e.g., semantic image segmentation (Chen, Papandreou, Kokkinos, Murphy, & Yuille, 2014)).

As the goals exhibit a flat sequential structure, linear-chain CRFs (Lafferty et al., 2001; Sutton & McCallum, 2012) are examined in the dissertation. In an implementation of linear-chain CRFs (Figure 5.2), the conditional probability of a sequence of classes (y) given a sequence of feature vectors (x) is defined in Equation 5.1 (Sutton & McCallum, 2012). In Equation 5.1, \( x_t \) is the feature vector at time \( t \), and \( y_t \) is the class label for the feature vector at time \( t \). \( T \) is the length of the sequence (i.e., the length of \( x \) and \( y \)), \( K \) is the number of feature
functions, while $f_k$ is the $k$th feature function associated with $y_t$, $y_{t-1}$, and $x_t$, and $\theta_k$ is the parameter for $f_k$. Lastly, $Z(x)$ is a normalization function per instance, $x$ (Equation 5.2).

\[
p(y|x) = \frac{1}{z(x)} \prod_{t=1}^{T} \exp \{ \sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, x_t) \} \tag{5.1}
\]

\[
Z(x) = \sum_{y} \prod_{t=1}^{T} \exp \{ \sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, x_t) \} \tag{5.2}
\]

Figure 5.2: A factor graph representation of linear-chain CRFs (Sutton & McCallum, 2012). $y$ is a label vector, each component of which (e.g., $y_1$) represents the label of the feature vector (e.g., $x_1$) that consists of features (e.g., $x_1$, $x_2$). $n$ is the number of features. The shaded boxes are factor nodes.

Figure 5.2 illustrates a factor graph of linear-chain CRFs, and Figure 5.3 depicts a CRF-based goal recognition model in an unfolded structure over time that ranges from 1 to $T$. The features are engineered based on the action properties at a time step, wherein each action property is encoded in one-hot representations, and so each action property node (shaded node in Figure 5.3A) consists of multiple features rather than one feature. A feature vector is created by concatenating all the action property-based features. An instantiation of the feature vector is the current time basis (e.g., $Action_T$ to $x_T$).
As in LSTMs, we set the maximum length of an action sequence to ten for training complexity; if the length of an action sequence is greater than ten, the actions taken prior to the last ten actions are discarded, constructing an action sequence only with the last ten actions.

For identifying the best performing CRF-based goal recognition models, the dissertation examines two hyperparameters. First, we investigate the model optimization technique between (1) OneSlackSSVM: a structural support vector machine (SVM) using the
1-slack formulation and cutting plane method (Joachims, Finley, & Yu, 2009) using a convex optimization technique (Andersen, Dahl, Liu, & Vandenberghe, 2011) and (2) SubgradientSSVM: structured SVM solver using subgradient descent (Ratliff, Bagnell, & Zinkevich, 2006; Shalev-Shwartz, Singer, Srebro, & Cotter, 2011). Second, we examine the regularization parameter for the both optimization techniques among \{0.1, 0.5, 1\}. The maximum number of iterations over a dataset to find constraints and perform updates is set to 41 for OneSlackSSVM and 11 for SubgradientSSVM considering the training efficiency. PyStruct (Müller, 2014), an off-the-shelf CRF modeling library, is used for devising CRF-based goal recognition models.
CHAPTER 6
EVALUATION

In the GOALIE framework, competitive goal recognition models are induced based on the four machine-learning techniques, Markov logic networks (MLNs), n-gram encoded feedforward neural networks pre-trained with stacked denoising autoencoders (FFNNs), linear-chain conditional random fields (CRFs), and long short-term memory networks (LSTMs), and evaluated through the five multidimensional evaluation metrics including the three conventional metrics: predictive accuracy (accuracy, precision, recall, and F1 rates) convergence rate, and convergence point and the two novel metrics, standardized convergence point and n-early convergence rate. In the dissertation, we only report 1-early convergence rate for n-early convergence rate, by which we evaluate whether goal recognizers correctly predict the goal for the last two actions in every action sequence.

While LSTMs, CRFs, and FFNNs have identified the best performing models through a machine-driven approach (i.e., grid-search of hyperparameters in cross validation), MLNs have utilized human expert-crafted logic formulae in terms of discovery events, domain-specific representations of user progress specifically targeted to each digital game (Baikadi et al., 2014). To measure generalizability of the model learning and evaluation processes supported in GOALIE, two gameplay corpora from two different open-world educational games are examined. For each data corpus, we use the same player-level data split in 10-fold cross validation across the machine-learning techniques, for fair comparisons. The following sections present evaluation results conducted on the two goal recognition data corpora.
6.1 Results for Crystal Island: Outbreak

As a brief summary of the Outbreak dataset, the game interaction data includes 77,182 player actions and 893 achieved goals, with an average of 86.4 player actions per goal. The distribution of the seven goals ranges from 6.4% to 26.6% (i.e., majority class baseline). Table 6.1 describes averaged accuracy rate results based on 10-fold cross validation across different goal modeling techniques.

Distributed LSTMs utilize a projection layer to learn distributed action representations, whereas discrete LSTMs harness one-hot encoding to represent each action property without employing a projection layer. The size of input units to distributed LSTMs is 100 and 200 for the embedding size \(d\) of 10 and 20, respectively, which are calculated by “10 action properties (an action type, a location, a narrative state, and seven previously achieved goals)” x \(d\). On the other hand, discrete LSTMs has the input feature size of 122, which is computed by 56 units for previously achieved goals (eight possible values, including None, for each of the seven previously achieved goals) + 19 possible values for the action type + 39 possible locations + 8 possible narrative states.

The hyperparameter search space for LSTMs and CRFs were presented in Sections 5.2 and 5.3, respectively. Here, we describe hyperparameters investigated for FFNNs and MLNs. For FFNNs, we explore two hyperparameters: \(n\) in \(n\)-gram encoding among \(\{1, 2, 3, 4, 5\}\) and the number of hidden layers among \(\{2, 3\}\). For other hyperparameters, we utilize the corruption level of 0.5 for stacked denoising autoencoders (fraction of corrupted input neurons) (Vincent et al., 2010), 100 hidden units per layer, and
one-hot encoding for the input representation. Also, we adopt stochastic gradient descent for both unsupervised pre-training and supervised fine-tuning, with learning rates of 0.1 for pre-training and 1 for fine-tuning, which are decided based on a preliminary work (Min, Ha, et al., 2014). Since FFNNs have a fixed network structure rather than a recurrent structure, the size of input neurons rapidly grows as we have a higher \( n \) in \( n \)-gram encoding. For example, unigram-encoded FFNNs take as input 62 input neurons, while 5-gram encoded FFNNs utilize 342 input neurons. Formally, the input size of \( n \)-gram encoded FFNNs is as follows:
\[
(n \text{ actions}) \times (19 \text{ action types} + 39 \text{ locations} + 4 \text{ bit representations of narrative states}) + (n-1 \text{ previously achieved goals}) \times (8 \text{ possible goals including } None).
\]

MLNs were trained based on human expert-crafted logic formulae. In this dissertation, we evaluate two types of MLNs, baseline MLNs and discovery event-based MLNs. The discovery event-based MLNs extended the initial set of logic formulae designed to train baseline MLNs (Ha et al., 2011), considering domain-specific representations of user progress specifically targeted to the game of interest (Baikadi et al., 2014).

As noted, GOALIE performs multidimensional goal recognition model evaluation, using the five metrics. A higher number is better for the accuracy rate, convergence rate, and \( n \)-early convergence rate, while a lower score is better for the convergence point and standardized convergence point, since these two latter metrics indicate how early a goal recognizer can inform the system about the player’s current goal.

Table 6.1 describes the accuracy rates of the five goal modeling techniques across individual hyperparameter settings. For the multidimensional model performance evaluation,
Table 6.1: Accuracy rates of goal recognition models for the OUTBREAK dataset. Distributed LSTM models per “the size of embeddings – the number of hidden units – the dropout rate”, discrete LSTM models per “the number of hidden units – the dropout rate”, CRF models per “the optimization algorithm – the regularization parameter”, FFNN models per “n-gram encoding – the number of hidden layers”, and MLN models per “class of logic formulae” are reported. The highest accuracy rate per approach is marked bold.

<table>
<thead>
<tr>
<th>Distributed LSTMs</th>
<th>Accuracy Rate</th>
<th>Distributed LSTMs</th>
<th>Accuracy Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 – 100 – 0.5</td>
<td>64.65%</td>
<td>20 – 100 – 0.5</td>
<td>65.13%</td>
</tr>
<tr>
<td>10 – 100 – 0.75</td>
<td>65.36%</td>
<td>20 – 100 – 0.75</td>
<td>66.35%</td>
</tr>
<tr>
<td>10 – 200 – 0.5</td>
<td>63.40%</td>
<td>20 – 200 – 0.5</td>
<td>63.52%</td>
</tr>
<tr>
<td>10 – 200 – 0.75</td>
<td>63.68%</td>
<td>20 – 200 – 0.75</td>
<td>62.21%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Discrete LSTMs</th>
<th>Accuracy Rate</th>
<th>Discrete LSTMs</th>
<th>Accuracy Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 – 0.5</td>
<td>64.69%</td>
<td>200 – 0.5</td>
<td>64.75%</td>
</tr>
<tr>
<td><strong>100 – 0.75</strong></td>
<td><strong>65.18%</strong></td>
<td>200 – 0.75</td>
<td>64.63%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CRFs</th>
<th>Accuracy Rate</th>
<th>CRFs</th>
<th>Accuracy Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>OneSlackSSVM – 0.1</td>
<td>60.95%</td>
<td>SubgradientSSVM – 0.1</td>
<td>63.65%</td>
</tr>
<tr>
<td>OneSlackSSVM – 0.5</td>
<td>61.15%</td>
<td><strong>SubgradientSSVM – 0.5</strong></td>
<td><strong>64.10%</strong></td>
</tr>
<tr>
<td>OneSlackSSVM – 1.0</td>
<td>58.28%</td>
<td>SubgradientSSVM – 1.0</td>
<td>61.64%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FFNNs</th>
<th>Accuracy Rate</th>
<th>FFNNs</th>
<th>Accuracy Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 – 2</td>
<td>49.16%</td>
<td>1 – 3</td>
<td>47.93%</td>
</tr>
<tr>
<td>2 – 2</td>
<td>50.94%</td>
<td>2 – 3</td>
<td>52.21%</td>
</tr>
<tr>
<td>3 – 2</td>
<td>57.55%</td>
<td>3 – 3</td>
<td>56.57%</td>
</tr>
<tr>
<td>4 – 2</td>
<td>59.79%</td>
<td>4 – 3</td>
<td>61.16%</td>
</tr>
<tr>
<td><strong>5 – 2</strong></td>
<td><strong>62.43%</strong></td>
<td>5 – 3</td>
<td>59.51%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MLNs</th>
<th>Accuracy Rate</th>
<th>MLNs</th>
<th>Accuracy Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>48.4%</td>
<td>Discovery event</td>
<td><strong>55.21%</strong></td>
</tr>
</tbody>
</table>
we compare results from the highest performing model per machine learning technique, which are (20 – 100 – 0.75) for the distributed LSTMs, (100 – 0.75) for the discrete LSTMs, (SubgradientSSVM – 0.5) for the CRFs, (5 – 2) for FFNNs, and the discovery event-based MLNs. Table 6.2 shows multidimensional evaluation results for CRYSTAL ISLAND: OUTBREAK.

Table 6.2: Accuracy rates (ACC), convergence points (CP), standardized convergence points (SCP), convergence rates (CR), and 1-early convergence rates (1-CR) of the highest performing goal recognition models for the OUTBREAK dataset. The best score per metric across goal modeling techniques is marked bold.

<table>
<thead>
<tr>
<th>Model</th>
<th>ACC</th>
<th>CP</th>
<th>SCP</th>
<th>CR</th>
<th>1-CR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed LSTMs</td>
<td>66.35%</td>
<td>33.34%</td>
<td>53.19%</td>
<td>71.32%</td>
<td>68.81%</td>
</tr>
<tr>
<td>Discrete LSTMs</td>
<td>65.18%</td>
<td>39.02%</td>
<td>58.31%</td>
<td>70.02%</td>
<td>67.26%</td>
</tr>
<tr>
<td>CRFs</td>
<td>64.10%</td>
<td><strong>15.06%</strong></td>
<td><strong>47.49%</strong></td>
<td>63.29%</td>
<td>62.38%</td>
</tr>
<tr>
<td>FFNNs</td>
<td>62.43%</td>
<td>41.30%</td>
<td>62.66%</td>
<td>70.06%</td>
<td>64.93%</td>
</tr>
<tr>
<td>MLNs</td>
<td>55.21%</td>
<td>30.80%</td>
<td>67.66%</td>
<td>49.09%</td>
<td>46.71%</td>
</tr>
</tbody>
</table>

For pairwise comparisons of the best performing models with respect to the accuracy rate, we run the Friedman test, a non-parametric equivalent of repeated measures ANOVA, along with a post-hoc analysis with Wilcoxon signed-rank tests, on the 10-fold cross validation results, since accuracy rates of folds do not necessarily follow normal distributions (Demšar 2006). Based on the Friedman test, there is a statistically significant difference in accuracy rates depending on the models, $\chi^2 (4) = 15.04$, $p=.005$. The Wilcoxon signed-rank post-hoc tests (Table 6.3) indicate there are statistically significant improvements in accuracy rates for distributed LSTM models over FFNNs and MLNs. Distributed LSTMs and CRFs do
not constitute a statistically significant difference, although the calculated $p$ value ($p=.14$) is close to the significance level (we use an alpha level of .05). All competitive baseline models significantly outperform MLNs.

Table 6.3: Wilcoxon signed rank statistical test results with respect to the accuracy rate for the OUTBREAK dataset. The values are formatted in “Z score ($p$ value)”. The machine learning techniques are sorted in the decreasing order with respect to the accuracy rate. Statistically significant results are marked bold, based on the significance level of 0.05.

<table>
<thead>
<tr>
<th></th>
<th>Distributed LSTMs</th>
<th>Discrete LSTMs</th>
<th>CRFs</th>
<th>FFNNs</th>
<th>MLNs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed LSTMs</td>
<td>-1.784 ($p=.074$)</td>
<td>-1.478 ($p=.139$)</td>
<td>-2.293 ($p=.022$)</td>
<td>-2.701 ($p=.007$)</td>
<td></td>
</tr>
<tr>
<td>Discrete LSTMs</td>
<td></td>
<td>-.764 ($p=.445$)</td>
<td>-1.886 ($p=.059$)</td>
<td>-2.803 ($p=.005$)</td>
<td></td>
</tr>
<tr>
<td>CRFs</td>
<td></td>
<td></td>
<td>-.663 ($p=.508$)</td>
<td>-1.988 ($p=.047$)</td>
<td></td>
</tr>
<tr>
<td>FFNNs</td>
<td></td>
<td></td>
<td></td>
<td>-2.191 ($p=.028$)</td>
<td></td>
</tr>
</tbody>
</table>

Additionally, we present precision, recall, and F1 results for a detailed illustration of the predictive performance per class label. The F1 score is defined as the harmonic mean between precision and recall. According to Blaylock and Allen’s chapter in Sukthankar et al. (2014), precision and recall for goal recognition are defined as follows:

- Precision: the number of correct predictions divided by the total number of predictions made ($\frac{\#\text{correct}}{\#\text{predictions}}$)
• Recall: the number of correct predictions divided by the total number of actions observed (\(\frac{\text{#correct}}{\text{#observations}}\))

These definitions of precision and recall might be more useful in online goal recognition systems because the number of predictions and observations can be different in real-time applications. For example, if the goal recognizer’s prediction confidence (e.g., the predicted goal’s posterior probability) is not sufficiently high, the goal recognition engine may give up informing the game adaptation engine of the current predicted goal, because game adaptations informed by incorrectly inferred goals can seriously impair the player’s gaming experience as discussed in Chapter 1. Instead, the goal recognition engine may postpone delivering the prediction result to the game adaptation engine until having a confident goal prediction utilizing more following observed actions.

However, our offline goal recognition models do not currently support selective goal recognitions, but enforce one prediction per time step (i.e., \(#\text{predictions} = \#\text{observations}\)). Thus, based on Blaylock and Allen’s definition on precision and recall, the multiclass classification accuracy, precision and recall rates are all equivalent to the F1 measure. For this reason, rather than adopting the definitions, we use precision and recall that have been broadly examined in general multiclass classification tasks. The definitions of precision and recall are as follows:

• Precision: Given instances that have predictions of a label, how many instances were correctly predicted?
• Recall: For all instances that should have a label, how many of these were correctly predicted?

Following these definitions, we provide detailed prediction results per goal label. Table 6.4 illustrates the precision, recall, and F1 scores for the two best-performing goal-modeling approaches, (1) the distributed LSTMs that achieve the highest accuracy and convergence rate, 1-early convergence rate and (2) CRFs that achieve the lowest convergence point and standardized convergence point. Test set evaluation results across the 10 folds are uniformly averaged to report a single value per metric.

Table 6.4: Precision (P), recall (R), and F1 scores of the distributed LSTMs and CRFs for the OUTBREAK dataset. G1–G7 are *speaking with a sick patient, running laboratory test on contaminated food, speaking with the camp nurse, submitting a diagnosis, speaking with the camp’s cook, speaking with the camp’s virus expert,* and *speaking with the camp’s bacteria expert,* respectively. The best score per measure between two goal recognition approaches is marked bold.

<table>
<thead>
<tr>
<th>Distributed LSTMs</th>
<th>G1</th>
<th>G2</th>
<th>G3</th>
<th>G4</th>
<th>G5</th>
<th>G6</th>
<th>G7</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>66.42%</td>
<td>74.97%</td>
<td><strong>75.83%</strong></td>
<td>83.65%</td>
<td><strong>67.88%</strong></td>
<td><strong>33.37%</strong></td>
<td>42.22%</td>
</tr>
<tr>
<td>R</td>
<td><strong>67.49%</strong></td>
<td><strong>73.95%</strong></td>
<td>86.68%</td>
<td><strong>80.74%</strong></td>
<td>77.00%</td>
<td>27.93%</td>
<td><strong>36.13%</strong></td>
</tr>
<tr>
<td>F1</td>
<td><strong>66.45%</strong></td>
<td>73.39%</td>
<td>79.81%</td>
<td><strong>81.14%</strong></td>
<td><strong>71.34%</strong></td>
<td>28.57%</td>
<td><strong>37.57%</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CRFs</th>
<th>G1</th>
<th>G2</th>
<th>G3</th>
<th>G4</th>
<th>G5</th>
<th>G6</th>
<th>G7</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td><strong>82.30%</strong></td>
<td><strong>86.22%</strong></td>
<td>75.19%</td>
<td><strong>84.91%</strong></td>
<td>65.18%</td>
<td>28.26%</td>
<td><strong>56.47%</strong></td>
</tr>
<tr>
<td>R</td>
<td>48.48%</td>
<td>67.87%</td>
<td><strong>100.00%</strong></td>
<td>79.25%</td>
<td>59.27%</td>
<td><strong>59.72%</strong></td>
<td>29.54%</td>
</tr>
<tr>
<td>F1</td>
<td>55.23%</td>
<td><strong>74.50%</strong></td>
<td><strong>85.09%</strong></td>
<td>80.94%</td>
<td>59.15%</td>
<td><strong>35.43%</strong></td>
<td>35.94%</td>
</tr>
</tbody>
</table>
Finally, we measure the average time to predict a goal per action, as the runtime efficiency is a key factor for goal recognition as pointed out in Blaylock and Allen (Blaylock & Allen, 2003). We empirically measure the speed of the distributed LSTMs and CRFs on 7,519 data samples. On average, it takes 0.016 milliseconds per goal prediction (0.122 seconds to predict 7,519 goals) and 0.055 milliseconds per goal prediction (0.416 seconds to predict 7,519 goals) for the LSTMs and CRFs, respectively. Both approaches yield reasonably fast prediction speeds, while LSTMs can recognize player goals 3.4 times faster than CRFs. For this evaluation, we use an Ubuntu machine with Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz, 65897MB RAM, and a GM200 (GeForce GTX TITAN X) graphics card.

6.2 Results for Crystal Island: Uncharted Discovery

The identical model training and evaluation process is applied to the Uncharted Discovery dataset, through which we estimate scalability of GOALIE to multiple digital games and evaluate its capacity as a generalized goal recognition framework.

As a brief summary of the Uncharted Discovery dataset, the dataset consists of 811,647 player actions and 7,652 achieved goals, with an average of 106.1 player actions per goal. The most frequent goal and the least frequent goal appeared 13.5% (i.e., majority class baseline) and 3.1% out of the entire set of players’ achieved goals, respectively.

As discussed, the size of input units for distributed LSTMs is contingent on the number of action properties and the embedding size. In Uncharted Discovery, distributed LSTMs take input features as many as “15 action properties (an action type, a location, a
narrative state, and 12 previously achieved goals)” x “d” (embedding size), in which d of 10 or 20 is examined in the dissertation. On the other hand, the input feature dimension of the one-hot encoding method is 382, which is computed by 156 units for previously achieved goals (13 possible values, including None, for each of the 12 previously achieved goals) + 126 possible values for the action type + 84 possible locations + 16 possible narrative states. In this dataset, the representational benefit of the distributed representation (e.g., 150 or 300) over the one-hot encoding (382) is highlighted.

The same hyperparameter space for the goal modeling approaches is investigated for Uncharted Discovery, except for FFNNs. From the previous analysis based on Outbreak, we found that higher n (4 for the three hidden layer models and 5 for the two hidden layer models) in the n-gram encoding achieves higher predictive accuracies relative to smaller ns. This result may indicate that FFNN models take advantage of more previous actions and previously achieved goals. A challenge posed by this approach, however, is on the high dimensionality of the input vector, as n increases. For Uncharted Discovery, the number of features that n-gram encoded FFNNs take as input is (n actions) x (126 action types + 84 locations + 16 narrative states) + (n-1 previously achieved goals) x (13 possible goals including None). Due to the computational complexity to train FFNNs with a higher n, the dissertation fixes n to 5 and investigates two hyperparameters: the number of hidden units per hidden layer among {100, 150, 200} and the number of hidden layers among {2, 3}. Table 6.5 illustrates the machine learning technique-level accuracy rates for the Uncharted Discovery dataset.
Table 6.5: Accuracy rates of goal recognition models for the UNCHARTED DISCOVERY dataset. Distributed LSTM models per “the size of embeddings – the number of hidden units – the dropout rate”, discrete LSTM models per “the number of hidden units – the dropout rate”, CRF models per “the optimization algorithm – the regularization parameter”, FFNN models per “the number of hidden units – the number of hidden layers”, and MLN models per “the class of logic formulae” are reported. The highest accuracy rate per approach is marked bold.

<table>
<thead>
<tr>
<th>Distributed LSTMs</th>
<th>Accuracy Rate</th>
<th>Distributed LSTMs</th>
<th>Accuracy Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 – 100 – 0.5</td>
<td>35.41%</td>
<td>20 – 100 – 0.5</td>
<td>34.28%</td>
</tr>
<tr>
<td><strong>10 – 100 – 0.75</strong></td>
<td><strong>35.81%</strong></td>
<td>20 – 100 – 0.75</td>
<td>34.94%</td>
</tr>
<tr>
<td>10 – 200 – 0.5</td>
<td>34.20%</td>
<td>20 – 200 – 0.5</td>
<td>35.29%</td>
</tr>
<tr>
<td>10 – 200 – 0.75</td>
<td>35.78%</td>
<td>20 – 200 – 0.75</td>
<td>35.32%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Discrete LSTMs</th>
<th>Accuracy Rate</th>
<th>Discrete LSTMs</th>
<th>Accuracy Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 – 0.5</td>
<td>34.52%</td>
<td>200 – 0.5</td>
<td>34.23%</td>
</tr>
<tr>
<td>100 – 0.75</td>
<td>34.64%</td>
<td><strong>200 – 0.75</strong></td>
<td><strong>34.91%</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CRFs</th>
<th>Accuracy Rate</th>
<th>CRFs</th>
<th>Accuracy Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>OneSlackSSVM – 0.1</td>
<td>28.52%</td>
<td>SubgradientSSVM – 0.1</td>
<td>34.48%</td>
</tr>
<tr>
<td>OneSlackSSVM – 0.5</td>
<td>27.41%</td>
<td>SubgradientSSVM – 0.5</td>
<td>33.70%</td>
</tr>
<tr>
<td>OneSlackSSVM – 1.0</td>
<td>26.97%</td>
<td>SubgradientSSVM – 1.0</td>
<td>34.01%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FFNNs</th>
<th>Accuracy Rate</th>
<th>FFNNs</th>
<th>Accuracy Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>100 – 2</td>
<td>31.82%</td>
<td>100 – 3</td>
<td>30.98%</td>
</tr>
<tr>
<td>150 – 2</td>
<td>31.56%</td>
<td>150 – 3</td>
<td>30.35%</td>
</tr>
<tr>
<td><strong>200 – 2</strong></td>
<td><strong>32.26%</strong></td>
<td>200 – 3</td>
<td>30.55%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MLNs</th>
<th>Accuracy Rate</th>
<th>MLNs</th>
<th>Accuracy Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline</td>
<td>22.60%</td>
<td>Discovery event</td>
<td>24.40%</td>
</tr>
</tbody>
</table>
For the multidimensional model performance evaluation, we compare results from the highest performing model per machine learning technique, which are (10 – 100 – 0.75) for the distributed LSTMs, (200 – 0.75) for the discrete LSTMs, (SubgradientSSVM – 0.1) for the CRFs, (200 – 2) for FFNNs, and the discovery event-based MLNs. Table 6.6 shows multidimensional evaluation results on models leveraging the configurations that achieve the highest accuracy rate per technique.

Table 6.6: Accuracy rates (ACC), convergence points (CP), standardized convergence points (SCP), convergence rates (CR), and 1-early convergence rates (1-CR) of the highest performing goal recognition models for the UNCHARTED DISCOVERY dataset. The best score per metric across goal modeling techniques is marked bold.

<table>
<thead>
<tr>
<th>Model</th>
<th>ACC</th>
<th>CP</th>
<th>SCP</th>
<th>CR</th>
<th>1-CR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed LSTMs</td>
<td>35.82%</td>
<td>53.80%</td>
<td>74.87%</td>
<td>57.84%</td>
<td>50.61%</td>
</tr>
<tr>
<td>Discrete LSTMs</td>
<td>34.91%</td>
<td>54.10%</td>
<td>76.00%</td>
<td>55.69%</td>
<td>50.13%</td>
</tr>
<tr>
<td>CRFs</td>
<td>34.48%</td>
<td><strong>27.42%</strong></td>
<td><strong>65.32%</strong></td>
<td>50.13%</td>
<td>46.37%</td>
</tr>
<tr>
<td>FFNNs</td>
<td>32.26%</td>
<td>47.38%</td>
<td>75.73%</td>
<td>49.94%</td>
<td>44.56%</td>
</tr>
<tr>
<td>MLNs</td>
<td>24.40%</td>
<td>75.54%</td>
<td>91.71%</td>
<td>43.64%</td>
<td>39.05%</td>
</tr>
</tbody>
</table>

Following the same procedure, we run the Friedman test followed by Wilcoxon signed-rank post-hoc tests on the accuracy rates obtained from the 10-fold cross validation. Based on the Friedman test, there is a statistically significant difference in accuracy rates depending on the models, $\chi^2 (4) = 30.16$, $p < .001$. The Wilcoxon signed-rank post-hoc tests (Table 6.7) indicate there are statistically significant improvements in accuracy rates for distributed LSTM models over all competitive baseline approaches, except for the discrete LSTM models ($p=0.203$). Models that achieve a higher accuracy rate significantly
outperform models with a lower accuracy rate, except for discrete LSTMs vs. CRFs that do not constitute a statistically significant difference ($p=0.721$).

Table 6.7: Wilcoxon signed rank statistical test results with respect to the accuracy rate for the Outbreak dataset. The values are formatted in “Z score ($p$ value)”. The machine learning techniques are sorted in the decreasing order with respect to the accuracy rate. Statistically significant results are marked bold, based on the significance level of 0.05.

<table>
<thead>
<tr>
<th></th>
<th>Distributed LSTMs</th>
<th>Discrete LSTMs</th>
<th>CRFs</th>
<th>FFNNs</th>
<th>MLNs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Distributed LSTMs</td>
<td>-1.274 ($p=.203$)</td>
<td>-2.090 ($p=.037$)</td>
<td>-2.803 ($p=.005$)</td>
<td>-2.803 ($p=.005$)</td>
<td></td>
</tr>
<tr>
<td>Discrete LSTMs</td>
<td>-0.357 ($p=.721$)</td>
<td>-2.803 ($p=.005$)</td>
<td>-2.803 ($p=.005$)</td>
<td>-2.803 ($p=.005$)</td>
<td></td>
</tr>
<tr>
<td>CRFs</td>
<td>-2.191 ($p=.028$)</td>
<td>-2.803 ($p=.005$)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FFNNs</td>
<td></td>
<td>-2.803 ($p=.005$)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

We also evaluate detailed prediction results based on precision, recall, and F1 for the distributed LSTM-based goal recognition models and CRFs; the former is identified as the most accurate goal modeling technique along with the highest convergence rates; the latter achieves the lowest standardized convergence points. As conducted in the analysis for Outbreak, test set evaluation results across the 10 folds are uniformly averaged to report a single value per metric. These results are described in Table 6.8.

For the Uncharted Discovery dataset, we also empirically measure the prediction speed of the distributed LSTMs and CRFs. Evaluation results show that it takes 0.020 milli-
Table 6.8: Precision (P), recall (R), and F1 scores of the distributed LSTMs and CRFs for the OUTBREAK dataset. G1–G12 are *Photograph Animal at 175E 25N*, *Photograph Animal at 200W 75S*, *Place Sign at Plateau*, *Place Sign at Waterfall*, *Place Sign at Volcano*, *Photograph Delta*, *Photograph Tributary*, *Photograph Lake*, *Pick up Dark Blue Flag*, *Pick up Green Flag*, *Pick up Burgundy Flag*, and *Photograph Animal at 75E*, respectively. The best score per measure between two goal recognition approaches is marked bold.

<table>
<thead>
<tr>
<th>Distributed LSTMs</th>
<th>G1</th>
<th>G2</th>
<th>G3</th>
<th>G4</th>
<th>G5</th>
<th>G6</th>
<th>G7</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td>34.25%</td>
<td>38.66%</td>
<td>43.99%</td>
<td>37.93%</td>
<td>34.51%</td>
<td>38.83%</td>
<td>60.69%</td>
</tr>
<tr>
<td>R</td>
<td>31.24%</td>
<td><strong>32.05%</strong></td>
<td><strong>38.22%</strong></td>
<td>35.64%</td>
<td><strong>38.42%</strong></td>
<td><strong>39.72%</strong></td>
<td>49.10%</td>
</tr>
<tr>
<td>F1</td>
<td><strong>32.04%</strong></td>
<td><strong>34.64%</strong></td>
<td><strong>39.96%</strong></td>
<td>36.57%</td>
<td><strong>36.08%</strong></td>
<td><strong>38.59%</strong></td>
<td>53.48%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CRFs</th>
<th>G1</th>
<th>G2</th>
<th>G3</th>
<th>G4</th>
<th>G5</th>
<th>G6</th>
<th>G7</th>
</tr>
</thead>
<tbody>
<tr>
<td>P</td>
<td><strong>37.00%</strong></td>
<td><strong>54.22%</strong></td>
<td><strong>75.05%</strong></td>
<td><strong>50.92%</strong></td>
<td><strong>38.40%</strong></td>
<td><strong>47.93%</strong></td>
<td><strong>75.96%</strong></td>
</tr>
<tr>
<td>R</td>
<td><strong>34.27%</strong></td>
<td>20.80%</td>
<td>29.42%</td>
<td><strong>38.44%</strong></td>
<td>21.96%</td>
<td>26.84%</td>
<td><strong>51.92%</strong></td>
</tr>
<tr>
<td>F1</td>
<td>31.56%</td>
<td>28.27%</td>
<td>39.93%</td>
<td><strong>42.57%</strong></td>
<td>25.38%</td>
<td>32.52%</td>
<td><strong>60.26%</strong></td>
</tr>
</tbody>
</table>

These results for seconds (1.66 seconds to predict 84,730 goals) for the LSTMs and 0.154 milliseconds (13.03 seconds to predict 84,730 goals) for the CRFs on average to predict a goal. These results for
Uncharted Discovery echo the results for Outbreak; LSTMs can recognize player goals quicker than CRFs (7.85 times faster).
CHAPTER 7
DISCUSSION

Explorations of the GOALIE framework in the two open-world digital games, CRYSTAL ISLAND: OUTBREAK and CRYSTAL ISLAND: UNCHARTED DISCOVERY, suggest that it is scalable. Recall that the GOALIE pipeline consists of low-level game trace data preprocessing (i.e., the four action property-based action representation), model training (i.e., LSTMs, CRFs, FFNNs, MLNs), and model evaluation (i.e., predictive accuracy such as accuracy rate, precision, recall, and F1, convergence point, standardized convergence point, convergence rate, and n-early convergence rate). GOALIE is scalable to multiple open-world digital games without extra domain-specific modifications.

7.1 Discussion on Empirical Results for CRYSTAL ISLAND: OUTBREAK

The results for OUTBREAK show that the distributed LSTM-based goal recognition models achieve a 20.2% marginal improvement over the best performing MLN model with a prediction rate accuracy of 66.3% (Table 6.2). Additionally, the distributed LSTMs significantly outperform FFNNs, a competitive baseline based on deep learning (DL) with marginal improvements of 6.3%. The distributed LSTMs also outperform CRFs, another competitive sequence labeling-based method, and the discrete LSTMs that do not use distributed action representations with marginal improvements of 3.5% and 1.8%, respectively. Possible explanations for the distributed LSTM-based model’s strong performance compared to other machine learning techniques are that, first, DL’s objective, “to discover high-level representations of raw low-level data”, is inherently related to the
latent intent of the goal recognition task, “to recognize higher-level patterns that result in goals using low-level action sequences”, second, the distributed LSTM-based goal recognition models automatically extract predictive features, specifically leveraging a distributed representation learning technique that represents discrete actions in a continuous vector space as well as representation learning provided by LSTMs, and third, the recurrent neural network formalism has effectively captured the sequential patterns pervasive in the players’ gameplay behaviors while preserving long-term dependencies through the gating units featured in LSTMs.

Additionally, the distributed LSTM-based goal recognition is found to make more accurate predictions not only on the last action (convergence rate of 71.3% in Table 6.2) but also on the last two actions (1-early convergence rate of 68.8%) than competitive baselines. On the other hand, LSTMs are found to have higher convergence points (distributed: 33.3%, discrete: 39.0%), which indicates that the models’ predictions slowly converge to the correct goal after observing longer sequences of actions within converged sequences, relative to CRFs (15.1%) and MLNs (30.8%). This result can be partially explained by the inherent tension between a high convergence rate and low convergence point (Min, Ha, et al., 2014; Min, Mott, et al., 2016). Accurate goal recognizers make correct goal predictions, even on noisy sequences of actions, which increases the number of converged action sequences, and thus the convergence rate. But goal recognition that occurs early in an observation sequence, especially with noisy data, is more often wrong, thereby yielding higher values for their convergence point. Since the convergence point metric can be misleading with respect to
favoring models with a better early prediction capacity, we have introduced a novel metric, standardized convergence point, that penalize non-converged action sequences in terms of early prediction, and use this metric over the conventional convergence point (Min, Baikadi, et al., 2016).

CRF-based goal recognition models possess the best early prediction capacity based on the convergence point (15.1%). CRFs also achieve the lowest standardized convergence point (47.5%) that outperforms the competitive models with sizable differences (the second best models with respect to the standardized convergence point are the distributed LSTMs with 53.2%). Interestingly, MLNs are the second best approach in terms of the convergence point metric (30.8%), but they do not exhibit a low early prediction score according to the standardized convergence point (the worst among the candidate approaches with 67.7% of the standardized convergence point). Considering that MLNs are the least reliable models that have the lowest accuracy rates as well as the lowest convergence rate-based scores, it can be problematic to conclude that MLNs have a high early prediction capacity. The standardized convergence point metric appropriately captures this potential threat, and suggests rectified results on models’ early prediction capacity.

The analysis based on precision, recall, and F1 between the distributed LSTMs and CRFs yield competitive results. LSTMs outperform CRFs in the F1 score for four goal labels. Among these, F1 scores for two (G1 and G5) of the four goals constitute more than the ten percent point difference. On the other hand, CRFs achieve higher F1 scores for three goal labels. Among them, two goals (G3 and G4) elicit more than the five percent point difference
compared to the LSTMs with respect to the F1 score. This clear distinction about models’ strengths on specific goal predictions informs directions for future work to obtain more accurate goal recognizers.

In the dissertation, we treat all three metrics, the accuracy rate, standardized convergence point, and \( n \)-early convergence rate, with an equal weight. Predictive accuracy represents the model’s overall reliability; standardized convergence point measures dynamic early prediction capacity; and the \( n \)-early convergence rate not only measures early prediction capacity, but also measures models’ reliability on the later action predictions that are often more important. Based on this criterion, we conclude that LSTM is the most reliable goal modeling technique when choosing a single model for the goal recognition engine. This model identification process is further discussed in Section 7.3.

### 7.2 Discussion on Empirical Results for CRYSTAL ISLAND: UNCHARTED DISCOVERY

Similar to the discussion above, we discuss findings from the GOALIE evaluation on the UNCHARTED DISCOVERY corpus.

For the predictive accuracy from the cross validation, the outperformance by the distributed LSTM-based goal recognition models is echoed in this dataset. The distributed LSTMs outperform all the non-LSTM-based competitive approaches in terms of the predictive accuracy (Table 6.6), with statistical significance (Table 6.7). Specifically, they outperform the best performing MLNs with 46.8\% marginal improvement, the best performing FFNNs with 11.0\% marginal improvement, the best CRFs with 3.9 \% marginal improvement, and the best discrete LSTMs with 2.6\% marginal improvement. The models’
predictive accuracy shows the same pattern as in the OUTBREAK analysis: distributed LSTMs > discrete LSTMs > CRFs > FFNNs > MLNs.

Notably, the distributed LSTM approach attains the highest accuracy with a smaller input vector size, relative to the discrete LSTM and CRF approaches. We discussed two representation techniques, distributed representation and one-hot representation, in Section 5.1. The distributed representation uses a dense feature vector where each feature can have a continuous value, and one-hot representation uses a sparse feature vector where only a feature is one, while all other features are zero. The best performing distributed LSTMs based on the distributed representation technique only use 150 dimensions to represent an action in a dense input feature space, whereas both the discrete LSTMs and CRFs grounded in one-hot encoding require 382 dimensions. Distributed LSTMs does not only outperform the other two computational approaches with respect to the accuracy rate, but also have a representational benefit by efficiently encoding an input.

Furthermore, results for standardized convergence point and 1-early convergence rate are echoed: the distributed LSTMs achieve the highest convergence rate (57.8%) and 1-early convergence rate (50.6%), and the CRFs achieve the lowest convergence point (27.4%) as well as the standardized convergence point (65.3%). More specifically, CRFs outperform distributed LSTMs with 12.8% marginal improvement with respect to the standardized convergence point, while LSTMs outperform CRFs with 9.1% marginal improvement for the 1-early convergence rate. FFNNs (47.4%) achieve a lower convergence point than the distributed LSTMs (53.8%) partially due to FFNNs’ low convergence rate, but the
standardized convergence point corrects the inherent tension, thereby inducing a lower standardized convergence point for the distributed LSTMs (74.9%) than FFNNs (75.7%).

Using the same criterion of choosing the best model (i.e., an equal weight assigned to the accuracy rate, standardized convergence point, and $n$-early convergence rate), we conclude that the distributed LSTMs are the best goal modeling technique for the UNCHARTED DISCOVERY dataset when evaluating with GOALIE, followed by CRFs.

We further analyze the two outperforming approaches, the distributed LSTMs and CRFs, using precision, recall and F1. This measurement is done also based on the cross validation results, in which precision, recall, and F1 are calculated for every goal per fold and the fold-level scores are averaged per metric.

We already have discussed that the differences between the distributed LSTM-based goal recognition models and the CRF-based goal recognition in terms of the accuracy rate are statistically significant, and so it is not surprising to see that LSTMs mostly outperform CRFs with respect to the F1 score for goals (LSTMs achieve higher F1 scores for nine goals out of 12 in total). However, we observe a clear pattern in this analysis as well as the previous analysis. CRFs especially predict G4, G7 and G12 better than the distributed LSTMs, achieving 6%, 6.8%, and 12.2% points of improvement, respectively. A larger implication suggested by these results with a particular focus on how to leverage distinctive performance by LSTMs and CRFs is discussed in Section 7.3.
7.3 Limitations

An important contribution of this work is identifying limitations and directions for improvement of GOALIE. To this end, we discuss four limitations of the current work in the following paragraphs.

First, to properly use GOALIE, we must specify a set of goals (i.e., goal schema) that plays the important roles to complete the digital game a priori. In the dissertation, we prescribed seven goals, which are key milestone events required to achieve the high-level objective, “solve the mystery”, for the OUTBREAK game, and also defined 12 central, quest-based goals for the UNCHARTED DISCOVERY game. Beyond these two investigated games, the GOALIE framework holds potential to be scalable to other types of games. For example, in real-time strategy games, goals can be designed based on the player’s high-level intention such as expand, explore, and attack, or based on build trees (i.e., tech trees), which are possible paths of research a player can take within the game (Bakkes, Spronck, & Herik, 2009; Kabanza et al., 2013; Synnaeve & Bessière, 2011; Weber & Mateas, 2009). Significant challenges, however, lie in digital games wherein a design of goals is not straightforward by nature and there are an enormous number of possible goals. For example, goals in the Go game characterized by a considerable number of strategies available in the game (Silver et al., 2016) are not easily identifiable, and this nature poses significant challenges to GOALIE, which has to perform a multi-class goal classification based on a pre-specified goal schema.

Second, GOALIE has shown promise for identifying a reliable goal recognition models through multidimensional model evaluation. Given a goal schema and action trace
logs that achieve specified goals a priori, GOALIE offers an evaluation capability to determine which machine-learned goal recognition models are more robust than other approaches with respect to the predictive performance and early prediction capacity (i.e., soundness). However, beyond this scope, GOALIE does not provide a rigorous method to evaluate the quality (e.g., validity, comprehensiveness) of the designed goal schema and thus does not guarantee the completeness of goal recognition.

To design goals, the current work has investigated a knowledge-lean approach (Ha et al., 2014). This approach uses a minimal set of domain knowledge for defining the goal schema, such as milestone events (OUTBREAK) and quests (UNCHARTED DISCOVERY) achievable by player actions, and thus provides a cost-effective solution compared to the approach that requires manual annotations of goals on the trace data. In spite of this efficiency, some types of games can leverage prospective benefits offered by sophisticatedly designed (i.e., customized) goals. As a promising application, customized goals hold potential for educational games, which contextualize learning and problem-solving within expansive virtual environments (Min et al., 2013). In educational games, it is of significant importance to identify students’ goals and behaviors that are not conducive to learning as early as possible and provide tailored gameworld events and pedagogical supports to discourage those behaviors and redirect students to achieve the game in a more deliberative manner (Min et al., 2015).

In case of educational games, goal recognition can detect sub-optimal learning behaviors by adding two goals, *gaming the system* (Baker, Corbett, Koedinger, & Wagner,
2004) and performing off-task. Gaming the system is a type of behaviors in which students attempt to solve problems by exploiting properties of the system. For instance, in CRYSTAL ISLAND: OUTBREAK, a student may keep submitting diagnosis worksheet (i.e., final solution) in hope of stumbling upon the correct solution, without gathering sources enough to solve the mystery (Ha et al., 2014). On the other hand, the off-task is a type of behaviors in which students keep exploring the virtual world in a way not directly related to the learning objective, such as “finding glitches in the software” and “climbing on top of the laboratory.” Designing these types of goals often requires students’ self-report, experts’ judgment, or system development that dynamically detect the scenarios and log the behaviors, so it is more labor-intensive than the knowledge-lean approach; nonetheless, these additional considerations on goals would significantly advance the utility of goal recognition in various domains.

Third, although GOALIE provides a scalable solution to support model training and model evaluation, it currently lacks a robust solution for choosing the final model (i.e., model identification) that will eventually operate as the core goal recognition model at runtime. We have presented a heuristic for choosing a model based on evaluation results in Section 7.1; based on the heuristic, the model’s accuracy rate, the standardized convergence point, and n-early convergence rates have the same degree of importance. Since this heuristic has not been fully examined, we discuss the model identification procedure as a limitation of the current work.
As seen in the evaluations of Outbreak and Uncharted Discovery, it can be common to see conflicting evaluation results. Conflicts can be resolved in a domain-specific approach considering the characteristics of the game being investigated. A simple solution is prioritizing goal recognizers based on the importance of measures such as the accuracy rate and convergence-based metrics. In some other cases, a specific goal prediction could be more important the others, and then we can use the F1 rates for the goal as an important criterion when comparing performances.

It is also possible to adopt an ensemble approach that utilizes multiple models to predict player goals. An example of this hybrid approach is using models’ confidence rates (e.g., posterior probability) weighted by the F1 scores per goal, by which the goal recognition engine can selectively harness each model’s distinctive strength. As indicated in Table 6.4 and Table 6.8, the two competitive models exhibit differentiated predictive performance depending on the goals; for the Outbreak data corpus, LSTMs outperform CRFs for two goals, while CRFs outperform LSTMs for other two goals with respect to the F1 score, both with sizable differences. The ensemble approach would enable the goal recognition engine to leverage individual strengths of models in a sophisticated fashion.

Lastly, we want to point out some limitations in the current implementation of the two novel metrics, standardized convergence point and \(n\)-early convergence rate. First, standardized convergence point addresses challenges in the classic convergence point by penalizing non-converged sequences. A natural question on this idea is how to appropriately penalize non-converged sequences. Suppose two non-converged action sequences, one
whose length is 10 and the other whose length is 100. Based on the presented metric with the penalty parameter of 1, the standardized convergence points are 1.1 and 1.01, as computed by \((10+1)/10\) and \((100+1)/100\), respectively (remember that if the goal recognizer makes a correct prediction only on the last action, then the standardized convergence point is 1). So, this metric virtually assumes that the goal recognizer would make a correct prediction on the action that follows the goal-achieving action. Of course, this virtual action does not exist in the real world, since the goal-achieving action is the last action in a sequence. This penalization poses a question about whether it is rational to have a shorter sequence (0.1 computed by 1.1 - 1) disadvantaged more than a longer sequence (0.01 computed by 1.01 - 1). To address this challenge, we introduced a settable penalty parameter \((p)\) in the standardized convergence point definition in Section 3.2.2.1. An implementation of standard convergence point can be set \(p\) to a multiple \((m)\) of the number of actions \((n)\) in the action sequence, so \(p\) equals \(m \times n\), where \(n\) dynamically changes depending on the length of the action sequence. In this case, the standardized convergence point becomes \(m + 1\) computed by \((n + m \times n) / n\), regardless of the lengths of action sequences. \(p\) should be determined per application; however, since we do not present a heuristic to choose \(p\) for a game, we leave this as a direction for future work.

Second, \(n\)-early convergence rate accounts for the importance of later actions, while also providing means of measuring models’ early prediction capacity. An open question for this metric is to find a proper \(n\). In the dissertation, we proposed a fixed number for \(n\), but did not suggest what could be an appropriate number for \(n\) in GOALIE. One possible option is
using dynamic numbers for $n$ based on the length of the action sequence; for example, $n$ can be dynamically set to the half-length of the action sequence whose $n$-early convergence rate is being calculated. We leave these as open questions suggesting future directions for improving GOALIE.
CHAPTER 8

CONCLUSIONS AND FUTURE WORK

Human intelligence plays a pivotal role in interpersonal behaviors, communication, and relationships. Social interactions are pervasive in various forms such as driving and verbal communication in daily lives, and, to successfully engage in them, humans reason about others’ cognitive and affective states and take proper actions according to the inferred states as well as the context and situation. A broad range of research has been undertaken to emulate social intelligence using AI for plan, activity, and intent recognition.

Specifically targeting intent recognition in digital games, goal recognition has been the subject of growing attention. Automated player goal recognition is core functionality for intelligent open-world digital games that are designed to dynamically adapt gameplay experiences. As noted above, player goal recognition in open-world digital games poses significant challenges due to the inherent complexity in players’ exploratory gameplay behaviors, and thus devising sophisticated goal recognition models that effectively deal with highly noisy action sequences is key to the success of goal recognition in open-world digital games.

In this dissertation, we have presented GOALIE, a generalized goal recognition framework, which supports a pipeline of devising reliable goal recognition models, including data preprocessing, model training, model evaluation, and model identification. The dissertation has demonstrated the generalizability of GOALIE through its scalability to two
different open-world educational games, **CRYSTAL ISLAND: OUTBREAK** and **CRYSTAL ISLAND: UNCHARTED DISCOVERY**, without requiring any domain-specific modification.

Multidimensional evaluations of goal recognition models administered in GOALIE show that long short-term memory network-based goal recognition models harnessing distributed feature representations (distributed LSTMs) achieve the highest accuracy rate as well as the highest $n$-early convergence rate, consistently across the two examined digital games. Based on the highest predictive performance, GOALIE identifies the distributed LSTMs as the best goal modeling technique for the two testbed games. Linear-chain conditional random fields (CRFs), a machine learning technique designed for structured predictions, exhibit the lowest standardized convergence point for the two data corpora, thereby demonstrating a high early prediction capacity across the two datasets, and thus are identified as the second best goal modeling technique.

In addition to distributed LSTMs’ outperformance with respect to the predictive performance, they have a representational benefit over discrete LSTMs and CRFs, both of which adopt the one-hot feature representations in order to encode player actions, since the distributed LSTMs learn and utilize dense, continuous representations of actions in a relatively smaller dimension. Especially for the **CRYSTAL ISLAND: UNCHARTED DISCOVERY** dataset, the best performing distributed LSTM models only require 150 dimensions to represent an action, while both CRFs and discrete LSTMs require 382 dimensions to represent the same action. The prediction speed results, which can be partially attributed to this representational strength, indicate that distributed LSTMs can recognize goals 7.85 times
faster than CRFs. Also, for the Crystal Island: Outbreak dataset, the distributed LSTMs utilizing 100 dimensions (10 – 100 – 0.75) outperform the best performing discrete LSTMs (100 – 0.75) that uses 122 dimensions (65.36% vs. 65.18%). In the following section, we revisit the hypotheses made in this dissertation using the empirical evaluation results.

8.1 Hypotheses Revisited

This dissertation investigated the thesis that the GOALIE framework can offer a principled, generalized methodology to devise and evaluate goal recognition models with an aim to provide an insight to predictive models that are more accurate and converge faster for various open-world digital games. Five hypotheses were investigated to support this thesis statement.

Hypothesis 1: The pipeline for goal recognition supported by GOALIE, including low-level game trace data preprocessing, model training, and model evaluation process, is scalable to multiple open-world digital games such as Crystal Island: Outbreak and Crystal Island: Uncharted Discovery.

- Empirical results show that (1) the action representation method based on the four action properties are scalable to the two open-world digital games, (2) goal recognition models are trainable based on the preprocessed data using multiple machine learning techniques, and (3) goal recognition model performance can be evaluated using the multidimensional metrics furnished in GOALIE for the two game corpora. So, we have confirmed that the GOALIE framework provides a scalable solution to the both games.
**Hypothesis 2:** LSTM-based goal recognition models outperform linear-chain conditional random fields (CRFs), \(n\)-gram encoded feedforward neural networks pre-trained with stacked denoising autoencoders (FFNNs), and Markov logic networks (MLNs) with respect to the predictive accuracy across the two open-world educational games, when evaluated using GOALIE.

- Evaluation results on 10-fold cross validation show that (1) the distributed LSTMs achieved a statistically significant improvement over FFNNs and MLNs for the OUTBREAK dataset, and also (2) the distributed LSTMs significantly outperformed CRFs, FFNNs and MLNs for the UNCHARTED DISCOVERY dataset. The distributed LSTMs also outperform CRFs for the OUTBREAK dataset, but the difference between these two approaches does not constitute a statistical significance (\(p=0.14\)).

**Hypothesis 3:** LSTM-based goal recognition models outperform CRFs, FFNNs, and MLNs with respect to the convergence rate and convergence point, across the two open-world educational games, when evaluated using GOALIE.

- Empirical evaluations using 10-fold cross validation show that the distributed LSTMs converged more often than the competitive baselines with sizable improvement. For the OUTBREAK dataset, the convergence rate of the distributed LSTMs is 71.3%, outperforming CRFs (63.3%), FFNNs (70.1%) and MLNs (49.1%). For the UNCHARTED DISCOVERY dataset, the distributed LSTMs (57.8%) also outperform CRFs (50.1%), FFNNs (49.9%) and MLNs (43.6%) with respect to the convergence rate.
Empirical evaluations on 10-fold cross validation show that the CRFs converged sooner than the competitive baselines for converged sequences with substantial differences. The distributed LSTMs achieved the third best convergence points, while the CRFs achieved the best convergence points, consistently across the two datasets.

**Hypothesis 4:** LSTM-based goal recognition models outperform CRFs, FFNNs, and MLNs with respect to the standardized convergence point and $n$-early convergence rate across the two open-world educational games, when evaluated using GOALIE.

- In the dissertation, we illustrate an application of the $n$-early convergence rate using $n$ of 1. Based on the 1-early convergence rates, distributed LSTMs achieved the highest scores consistently on the both datasets.

- For the standard convergence point employing the penalty parameter of 1, CRFs achieved the lowest score in the both datasets, while the distributed LSTMs, FFNNs, and MLNs sequentially achieved the next best score consistently for the both datasets.

**Hypothesis 5:** LSTM-based goal recognition models that harness distributed action representations through a linear projection layer (i.e., distributed LSTMs) outperform the corresponding LSTM models that utilize the one-hot encoding-based discrete action representations (i.e., discrete LSTMs) with respect to the accuracy rate, convergence rate, $n$-early convergence rate, and standardized convergence point across the two open-world educational games, when evaluated using GOALIE.

- Interestingly, the same performance pattern has been found from the two game analyses; the distributed LSTMs outperformed the discrete LSTMs across all the five
metrics. The difference in the predictive accuracy did not constitute statistical significances ($p=0.07$ and $p=0.20$) for the both games. However, based on the outperformance on all the metrics, we conclude that the distributed LSTMs strongly outperformed the discrete LSTMs. It should be noted that the distributed LSTMs could achieve a better accuracy rate using lower dimensional input feature vectors.

8.2 Future Work

There are several promising directions for future work. First, it will be important to investigate various forms of recurrent neural networks along with different optimization and regularization techniques in order to identify models achieving a higher predictive accuracy. The gating mechanism supported in LSTMs has effectively dealt with the vanishing gradient problems, but also demands a number of matrices to store parameters for the input, forget, and output gates along with the candidate memory state and bias vectors. The number of parameters is a key factor to devise a goal recognition model, because it will significantly affect the system memory as well as the computation time for optimizing the model. Recently, Cho et al. (2014) proposed gated recurrent units (GRUs) that feature two gating units: a reset gate and an update gate. Compared to the LSTMs investigated in the dissertation, GRUs do not have a gate equivalent to the output gate, but employ a candidate activation (it is similar to the candidate memory state in LSTMs, but it is internally modulated by the reset gate) along with the two mentioned gating units to update the activation of GRUs. According to empirical evaluations on a set of sequence labeling tasks including music and raw speech signal data, GRUs showed comparable performance to
LSTMs, while significantly outperforming classic recurrent neural networks. In this evaluation, GRUs and LSTMs used a similar number of entire parameters, by which GRUs could utilize more hidden units than LSTMs due to the reduced number of gates. Although it is uncertain how GRUs would perform compared to LSTMs for goal recognition tasks, it could be a strong candidate technique for goal recognition in the future.

In parallel, model optimization and regularization can further enhance the model performance. Model optimization aims to minimize a cost function while training a model, and regularization adds some noise to the cost function with an aim to improve the generalization error, effectively avoiding overfitting. In the neural network formalism, model optimization consists of finding a set of optimal hyperparameters (e.g., the number of hidden units) as well as a set of optimal model parameters (e.g., weights) based on the predetermined hyperparameters. For hyperparameter optimization, a recent work by Bergstra and Bengio (2012) suggested that randomly chosen hyperparameters are more efficient than a grid search in the same computational budget. Based on a Gaussian process analysis of the function from hyperparameters to validation set performance, most hyperparameters do not impact the model’s predictive performance but only a few set of hyperparameters really matters; thus, a random search that explores a larger configuration space can more efficiently find a set of hyperparameters than a grid search that often consumes computational resources on less impactful hyperparameters. This result directly informs a method to determine the hyperparameters for neural networks, such as optimization-focused hyperparameters (e.g., learning rates, momentum, optimization techniques), model structure-specific
hyperparameters (e.g., the number of hidden units, weight initialization), and training criterion-related hyperparameters (e.g., activation function, loss function, and regularization parameters such as dropout rates, and corruption levels). Regarding optimization, a broad range of optimization techniques has been actively introduced, including RMSprop, Adagrad, Adadelta, Adamax, and Nadam among others (Chollet, 2015), in addition to the Adam stochastic optimization method that the current work utilized. Choosing an appropriate optimization technique as well as hyperparameters could significantly improve the generalization error in goal recognition.

Another promising line of work lies in incorporating distributed representations trained in LSTMs within other computational models. In this dissertation, we have demonstrated that the distributed LSTMs outperform the discrete LSTMs that explore the same hyperparameter space. In the meantime, linear-chain CRFs have shown great potential in the early prediction capacity evidenced by the lowest standardized convergence point, while leveraging the one-hot encoding method in representing features. Considering these two findings, CRFs that harness deep learning’s feature extraction capacities have significant promise to advance the current goal recognition model performance. A variety of recent previous work has investigated CRFs along with deep learning for computational tasks. Chrupala (2014) proposed an edit operation-based text normalization approach, in which features utilized in CRF models are character-level neural text embeddings induced from recurrent neural networks. This approach substantially lowered word error rates on an English tweet normalization dataset, producing a state-of-the-art result. Another line of work
was conducted in computer vision, specifically targeting semantic image segmentation (Chen et al., 2014). In this work, the authors connected deep convolutional neural networks’ final layer, which is not sufficiently localized for accurate object segmentation, with CRFs. This combined model achieved a level of accuracy beyond previous approaches. Inspired by these results, the method of intertwining the distributed action representation trained in LSTMs with CRFs could significantly advance the current quality of CRF-based goal recognition models.

Regarding the Crystal Island goal recognition data corpora, it will be important to validate how the current goal schema aligns with players’ actual goals. It is possible that players were attempting to achieve goals that are not currently designed in the games. A sophisticated design of goals would not only keep players engaged in the game through tailored game adaptations, but also deliver detailed information to game designers for the future game analysis, refinement and development process. On the other hand, it will be important to measure reliability between the player’s actual goal and the goal that the data corpus labeled. It is possible that players accidentally achieve a goal while keeping another goal in mind. Players might inadvertently achieve a goal through exploratory actions. Players also might have multiple goals in mind, where the player actions can be concurrent or interleaved to achieve the goals. To label players’ actual goals, a prospective way (e.g., self-report, think aloud) or a retrospective way (e.g., labeling goals on a recorded gameplay video by either players or experts) are available; since the former approach might disrupt the players’ gameplay while the latter approach does not guarantee to obtain accurate goals, a
sophisticated manner of getting gold standard labels should be devised. We did not note these considerations as limitations of this work because GOALIE can induce and evaluate goal recognition models using the new data, once it becomes ready. However, we agree that a deliberate process of designing and labeling goals is a pivotal future direction to provide fine-grained game adaptations in open-world digital games.

Lastly, in regard to goal recognition in educational games, it will be important to investigate the relationships between the players’ goals and learning outcomes (Baikadi, 2014). A goal recognition system that detects off-task behaviors and gaming-the-system behaviors can collaborate with an automated knowledge assessment system that measures students’ learning. It might be also interesting to examine if there are distinctive patterns in achieved goal sequences between high-performing students and low-performing students using a sequence mining technique; a post-hoc analysis of relating achieved goals to students’ learning gains could inform which problem-solving strategies offer a more principled way to promote one’s learning outcomes. Multimodal sensors (e.g., gaze, facial action units, posture, galvanic skin responses) have been widely investigated in game-based learning environments (Min, Wiggins, et al., 2016), often to measure learners’ affective and cognitive states. These real-time, multi-channel data sources generated from students have potential to serve as strong predictive features for goal recognition. Also, players’ inferred affective models and goals together would constitute a rich player model, which can be used in service of personalized game adaptations. Additionally, static student profiles such as demographics, pre-knowledge, or prior gameplay experience can also serve as individualized
explanatory variables to recognize player goals. Finally, not constrained to educational games but in general open-world digital game environments, it will be important to investigate how goal recognition models operate at run-time to most effectively drive gameplay personalization in player-adaptive games.

8.3 Concluding Remarks

Open-world digital games provide an ideal laboratory for investigating computational models of goal recognition, a core player modeling functionality. A key challenge in these environments is devising predictive models that accurately recognize player goals based on game interaction logs containing noisy data due to player exploration and minimally goal-directed behavior. This dissertation has introduced the GOALIE goal recognition framework that offers a scalable solution for devising reliable goal recognition models in open-world digital games through valid, multidimensional evaluations.

Through empirical evaluations on two different open-world educational games, we have found significant promise of GOALIE as a generalized goal recognition framework that does not require any domain-specific modifications. With respect to the two novel evaluation metrics, we have demonstrated that standardized convergence point can resolve an issue about the inherent tension between the convergence point and convergence rate, and the $n$-early convergence rate can serve as a proxy for measuring models’ early prediction by extending the classic convergence rate, simultaneously measuring models’ reliability on the later action predictions that are often more important. In summary, a series of evaluations has exhibited the distributed long short-term memory networks (LSTMs) achieve the highest
results in multiple categories. Notably, LSTMs automatically extract predictive features in the form of an end-to-end trainable model without labor-intensive manual feature engineering that is often required by other machine learning approaches, thereby proving themselves as a promising, scalable goal modeling technique.

We have discussed several promising future directions. Among these, it will be important to investigate various computational modeling techniques that incorporate the distributed representations trained in the distributed LSTMs. The distributed representations have been found to not only improve the models’ predictive performance but also possess representational benefits in an ablated experiment that has compared the distributed LSTMs to the discrete LSTMs that employ one-hot encoding instead. This empirical result along with various successful computational works that harness deep learning’s salient feature extraction capacity within other computational techniques provides an insight into the next-generation goal recognition modeling.

Our research to date has been informed by extensive analytical, empirical, and computational investigations of intent modeling, knowledge modeling, dialogue modeling and natural language processing, which have resulted in 15 peer-reviewed publications led- or co-written by the author. Intent modeling that constitutes the main body of this dissertation has been investigated in the context of players’ goal recognition in open-world digital games (Min, Baikadi, et al., 2016; Min, Ha, et al., 2014; Min, Mott, et al., 2016); knowledge modeling has been conducted for (1) problem-solving sequence modeling that examined collaborative filtering techniques such as probabilistic principal component analysis and non-
negative matrix factorization (Min et al., 2013), (2) performance modeling that investigated a semi-supervised learning technique to effectively deal with missing labels in open-world game-based learning environments (Min, Mott, Rowe, et al., 2014), (3) skill and knowledge scaffolding (Min, Mott, & Lester, 2014), (4) stealth assessment that measures students’ knowledge in an unobtrusive and non-disruptive manner utilizing a sequence of problem-solving interactions with game-based learning environments (Min et al., 2015), and (5) students’ drawing and writing performance predictions (Smith et al., 2016; Smith, Min, Mott, & Lester, 2015). Another line of work has investigated dialogue modeling for intelligent virtual agents in educational games (Min, Wiggins, et al., 2016). In this work, we analyzed students’ game trace logs and multimodal data streams that were revealed during the interaction with the game, in order to devise a computational model that predicts the virtual agent’s next dialogue act, the underlying intention (e.g., greeting, question, suggestion). Lastly, the work conducted to address natural language processing tasks such as text normalization and part-of-speech tagging has inspired the current dissertation. Min and Mott have investigated LSTMs to normalize noisy texts in Twitter, and their submission to the constrained track of the ACL 2015 W-NUT shared task for text normalization achieved the second place in the competition (Min & Mott, 2015). This work utilizes a projection layer that projects each character represented in a one-hot encoding scheme onto an n-dimensional continuous space, which has significantly influenced the current direction of goal recognition described in this dissertation. In addition to the four major research thrusts, we have
investigated a Unity-based toolkit for creating expressive user interfaces in game-based learning environments (Mott, Rowe, Min, Taylor, & Lester, 2014).

We expect that the work described in the dissertation will yield insights into the effective designing, modeling, and evaluating of goal recognition models for open-world digital games, inspire game designers who plan to build player goal driven adaptive games, and eventually improve player experience and enhance replayability. Also, we hope this computational work makes a contribution to the plan, activity, and intent recognition research field, and enhances game AI and the player modeling research.
REFERENCES


Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in neural information processing systems* (pp. 2933–2941).


Firaxis Games. (2005). Civilization IV.


MIT Press.


Le, Q., & Mikolov, T. (2014). Distributed Representations of Sentences and Documents. In


Lester, J., Ha, E., Lee, S., Mott, B., Rowe, J., & Sabourin, J. (2013). Serious games get


Min, W., Mott, B., & Lester, J. C. (2014). Adaptive Scaffolding in an Intelligent Game-Based Learning Environment for Computer Science. In The Second Workshop on AI-supported Education for Computer Science (pp. 41–50).


Deep Learning and Unsupervised Feature Learning Workshop (pp. 1–9).


