Player Goal Recognition in Open-World Digital Games with Long Short-Term Memory Networks

Wookhee Min, Bradford Mott, Jonathan Rowe, Barry Liu, James Lester
Department of Computer Science
North Carolina State University
Raleigh, North Carolina 27695, USA
{wmin, bwmott, jprowe, bliu12, lester}@ncsu.edu

Abstract

Recent years have seen a growing interest in player modeling for digital games. Goal recognition, which aims to accurately recognize players' goals from observations of low-level player actions, is a key problem in player modeling. However, player goal recognition poses significant challenges because of the inherent complexity and uncertainty pervading gameplay. In this paper, we formulate player goal recognition as a sequence labeling task and introduce a goal recognition framework based on long short-term memory (LSTM) networks. Results show that LSTM-based goal recognition is significantly more accurate than previous state-of-the-art methods, including *n*-gram encoded feedforward neural networks pre-trained with stacked denoising autoencoders, as well as Markov logic network-based models. Because of increased goal recognition accuracy and the elimination of labor-intensive feature engineering, LSTM-based goal recognition provides an effective solution to a central problem in player modeling for open-world digital games.

1 Introduction

Open-world digital games enable players to explore and pursue gameplay objectives within expansive virtual worlds [Squire, 2008]. In contrast to linear games, which prescribe a particular sequence of gameplay objectives for players to accomplish, open-world games support vast numbers of possible paths through game environments. Players select or formulate their next objectives and devise plans to accomplish them. Open-world games provide an ideal laboratory for investigating computational techniques for plan, activity, and intent recognition because they can support a broad range of goal-directed player behavior, and they can be fully instrumented via telemetry functionalities [Harrison *et al.*, 2015; Kabanza *et al.*, 2013; Yannakakis *et al.*, 2013]. 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 [Riedl and Bulitko, 2013; Min *et al.*, 2014].

Previous work has investigated these challenges in the context of player modeling, often with the goal of creating computational models for dynamically adapting gameplay to players' cognitive, behavioral, and affective states [Yannakakis *et al.*, 2013]. Such player-adaptive games have incorporated computational techniques for interactive narrative [Riedl and Bulitko, 2013], procedural content generation [Shaker *et al.*, 2015], game balancing [Lopes and Bidarra, 2011] and personalized learning in educational games [Ha *et al.*, 2011; Min *et al.*, 2014].

A central problem in player modeling is *goal recognition*, which is a restricted form of plan recognition. Goal recognition focuses on identifying a player's high-level objectives in a game world, given a series of low-level gameplay actions [Ha *et al.*, 2011; Harrison *et al.*, 2015; Min *et al.*, 2014]. Goal recognition models compute predictions of players' dynamically changing goals and inform run-time adaptation of gameplay to enhance players' experience [Kabanza *et al.*, 2013].

Prior work on plan, activity, and intent recognition has largely focused on observation sequences in which agents' actions are rationally motivated by well-defined objectives. For instance, smart space activity monitoring [Duong *et al.*, 2009], network security [Geib and Goldman, 2009], and natural language story understanding [Singla and Mooney, 2011] have served as testbed applications for plan, activity, and intent recognition. These 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.

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 *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, goal recognition in open-world digital games is characterized by considerable uncertainty, and goal recognition models must be able to operate robustly even in the face of highly noisy sequences of low-level player actions.

This paper presents a computational framework for player goal recognition based on long short-term memory networks (LSTMs) [Hochreiter and Schmidhuber, 1997], a variant of recurrent neural networks. We formalize goal recognition as the task of predicting the player's next gameplay objective in service of completing the overarching game. Our goal recognition framework is evaluated with game interaction logs from over 100 players interacting with CRYSTAL ISLAND, an open-world educational game for middle school science. Findings from the evaluation suggest that LSTM-based goal recognition significantly outperforms the previous state-of-the-art method based on *n*-gram encoded feedforward neural networks pre-trained with stacked denoising autoencoders [Min *et al.*, 2014]. Further, we find that LSTM-based goal recognition outperforms non-deep learning methods, including a discovery event-based Markov logic network model [Baikadi *et al.*, 2014], with respect to predictive accuracy and convergence rate. Notably, the 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.

2 Related Work

Plan, activity, and intent recognition 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; Kautz and Allen, 1986; Sukthankar *et al.*, 2014]. It has been actively investigated with a variety of machine learning techniques such as Bayesian inference models [Geib and Goldman, 2009], dynamic Bayesian networks [Duong *et al.*, 2009], and Markov logic networks [Baikadi *et al.*, 2014; Ha *et al.*, 2011; Singla and Mooney, 2011].

Digital games have proven to be a valuable testbed for investigating plan, activity, and intent recognition. [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. Closely related to this work, [Ha *et al.*, 2011] used Markov logic networks (MLNs) to recognize player goals in an educational game, yielding significant improvements in goal recognition accuracy relative to previous *n*-gram and Bayesian network approaches [Mott *et al.*, 2006]. MLN-based goal recognition has been further explored by [Baikadi *et al.*, 2014]. Their work investigated narrative discovery events, domain-specific representations of player progress through the game, in MLNs, and demonstrated improved performance relative to the models reported in [Ha *et al.*, 2011].

Recent years have witnessed an explosion of interest in deep learning, which has contributed to significant advances in computer vision, speech recognition and natural language processing [LeCun *et al.*, 2015]. Deep learning has also been investigated in goal and plan recognition. [Bisson *et al.*, 2015] presented recursive neural network-based decision models and evaluated the approach on three plan recognition benchmark domains including StarCraft, a real-time strategy game. Recursive neural networks [Socher *et al.*, 2010] have demonstrated potential as a computational plan recognizer by outperforming a probabilistic plan-library based approach and an inverse planning approach. [Min *et al.*, 2014] investigated stacked denoising autoencoder pre-trained feedforward neural networks for goal recognition in an educational game, which significantly outperformed the previous state-of-the-art MLN models in [Ha *et al.*, 2011]. 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. The approach presented in this paper is the first to examine goal recognition with long short-term memory networks leveraging distributed action representations, which effectively model temporal information characterized in observation sequences of player behavior.

Figure 1. CRYSTAL ISLAND virtual environment.

3 Goal Recognition Corpus

CRYSTAL ISLAND (Figure 1) is an open-world educational game for middle school science. In the game players are tasked with identifying the cause of an illness afflicting a team of scientists on a remote island research camp. Implemented as a Half-Life 2 mod, CRYSTAL ISLAND's gameplay is similar to many exploration-centric games in which players experience the world from a first-person viewpoint and perform actions such as navigating from one location to another, discovering important items, picking up objects, and talking with non-player characters. CRYSTAL ISLAND's non-linear narrative consists of seven key goals that players must accomplish to complete the game. Five of these goals involve speaking with non-player characters about the spreading illness, while the remaining two involve testing contaminated food in the camp's laboratory and submitting a completed diagnosis to the camp nurse. CRYSTAL ISLAND has been shown to provide substantial increases in both learning and motivation [Rowe *et al.*, 2011] through a series of empirical studies.

During gameplay, CRYSTAL ISLAND logs all player actions, which can be retrieved for offline data analysis. The data used in the evaluation of our goal recognition model was collected during a study involving 153 eighth grade students from a middle school. Due to incomplete data or prior experience playing the game, data from 16 participants was removed. Our goal recognition corpus is composed of data from the remaining 137 participants.

It should be noted that players' action sequences do not necessarily represent optimal paths for achieving goals in CRYSTAL ISLAND. Rather, action sequences are often sub-optimal or 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. For example, in pursuit of a goal, *Test the Contaminated Food*, one player in our dataset performed 86 actions. These actions began with six actions encoded as *Talk with the Camp Nurse*, each corresponding to a dialogue turn during a branching conversation with one of the game's non-player characters. After several more actions in the infirmary, the player embarked on an exploration of the gameworld using multiple *Move* actions to different locations. The player actions continued in this manner: the player made gradual but circuitous progress toward the next objective, eventually culminating with the final action that achieved the goal.

4 Long Short-Term Memory Network-Based Goal Recognition

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]. 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 inspire our investigation of long short-term memory networks (LSTMs) [Hochreiter and Schmidhuber, 1997] for goal recognition. In this work, players' sequential interaction data is encoded using distributed action representations, which serve as the input to LSTMs. We present a brief overview of LSTMs, the input data encoding method, the distributed representation learning procedure for the action-based input data, and the LSTM-based goal recognition framework.

4.1 LSTM Background

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 effectivly addressing the vanishing gradient problem [Graves, 2012].

LSTMs (Figure 2A) feature a sequence of memory blocks that include one or more self-connected memory cells along with three gating units: an input gate, a forget gate, and an output gate [Graves, 2012]. In LSTMs, the input and output gates modulate the incoming and outgoing signals to the memory cell, and the forget gate controls whether the previous state of the memory cell is remembered or forgotten.

In this LSTM implementation, the input gate (i_t), forget gate (f_t), and candidate memory cell state ($\widetilde{c_t}$) at time *t* are computed by Equations (1) – (3), respectively, in which *W* and *U* are weight matrices for 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 σ and *tanh* are the logistic sigmoid and hyperbolic tangent function, respectively.

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (1)$$
$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (2)$$
$$\widetilde{c_t} = tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (3)$$

Once these three vectors are computed, the current memory cell's state is updated to a new state (c_t) by modulating the current memory candidate value ($\widetilde{c_t}$) 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 block decides whether to keep or forget the previous memory cell state via the forget gate and regulates the candidate of the current memory cell state via the input gate. This step is described in Equation (4):

$$c_t = i_t \widetilde{c_t} + f_t c_{t-1} \quad (4)$$

In Equation (5), the output gate (o_t), similarly calculated as in Equations (1) – (3), is utilized to compute the memory cell output (h_t) of the LSTM memory block at time *t*, based on the updated cell state (c_t) as in Equation (6):

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (5)$$
$$h_t = o_t \, tanh(c_t) \quad (6)$$

Once the cell output (h_t) is calculated at time *t*, the next step is to use the computed cell output vector to predict the label of the current training example. For player goal recognition, we use the final cell output vector, assuming that h_t captures long-term dependencies from the previous time steps.

4.2 Data Encoding for Goal Recognition

We cast goal recognition as a multiclass classification problem in which a trained classifier predicts the most likely goal associated with the currently observed sequence of actions after the previously observed goal. We assume 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

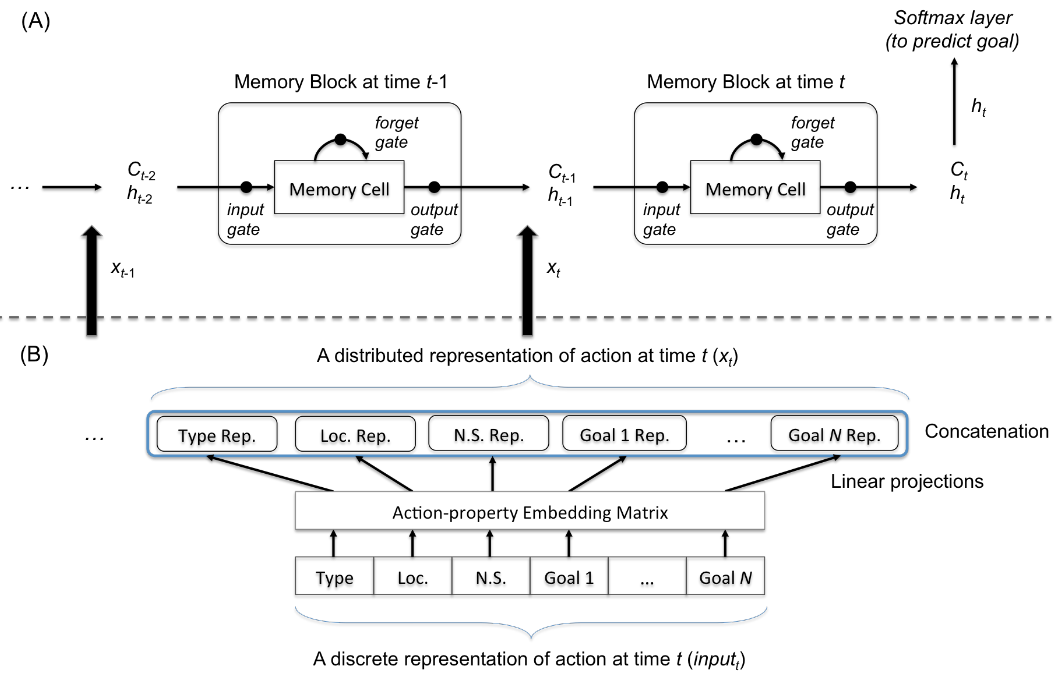

Figure 2. (A) An illustration of LSTMs [Graves, 2012] for goal recognition. The cell output vector, h_t, is utilized to predict the goal associated with x_t at time t using a softmax layer. (B) A distributed action representation (x_t) generated from a discrete action representation ($input_t$). An action-property embedding matrix linearly maps each action property (i.e., action type, action location, narrative state, previously achieved goals) in a discrete vector space onto a continuous vector space, and all N+3 continuous vectors are concatenated to generate a single distributed action representation, x_t (N=7 in this work). The induced x_t is fed into the LSTM memory block at time t.

lend itself to this type of analysis. In this setting, all actions that precede the current goal, but follow the previous goal, are labeled with the current goal, and all actions taken after achievement of the last goal are ignored, since these actions might not be directly related to gameplay goal pursuit.

A player action is encoded with four properties: *action type*, *action location*, *narrative state* (the player's progress in solving the game's narrative), and previously *achieved goals* [Min *et al.*, 2014]. In CRYSTAL ISLAND, a player action can be "*Talk* (action type) with the Camp Nurse at the *Camp Infirmary* (action location), after having achieved *Speak with the Camp Cook about Recently Eaten Food* (previously achieved goal) while having just completed the narrative milestone of *Submit a Diagnosis to the Camp Nurse* (narrative state)." In our work, action types include 19 distinct types of player actions, action location includes 39 non-overlapping sub-locations within the gameworld, narrative states contain 8 possible values based on the interactive storyline's plot structure, and there are 8 possible goals that could be previously achieved, which includes '*None*' in case the player has not yet achieved any goals. Note that the set of these four action properties was initially devised to be scalable to general digital games [Ha *et al.*, 2011] and can be adjusted according to characteristics of the game being investigated by adding specific properties that are expected to contribute to goal recognition or removing properties that are not applicable to the game.

To represent an action input in our work, we utilize a 10-dimensional discrete vector (Figure 2B). 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 seven dimensions represent a sequence of previously achieved goals (seven goals in total in this work) 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 is 86.4 according to [Ha *et al.*, 2011]), and (2) action sequences longer than a threshold (k) are cut to utilize the last k actions for the training efficiency in this work, we explicitly encode the previously achieved goals in the input.

4.3 Distributed Action Representations

Inspired by advances in distributed vector representations of text (e.g., words, sentences) [Bengio *et al.*, 2003; Le and Mikolov, 2014], this work investigates distributed representations of actions for goal recognition. Distributed representations of words have been examined with a range of neural networks [Bengio *et al.*, 2003; Le and Mikolov, 2014], and they have been investigated in a wide range of natural language processing tasks, such as language modeling [Bengio *et al.*, 2003], language parsing [Socher *et al.*, 2010] and sentiment analysis [Le and Mikolov, 2014].

In our work, an action consists of multiple properties, so distributed action representations are managed on a per-property basis. To operationalize this, a comprehensive action-property embedding matrix, the size of which is 74 (the total number of possible values of action properties computed as 19+39+8+8) by d (embedding size), is created

(Figure 2B). The action-property embedding matrix 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. We denote the first approach as *supervised embedding* and the second approach as *pre-trained embedding*. For the pre-trained embedding approach, we utilize another LSTM model to predict the next action based on a sequence of prior actions, and train the model using only player action data without goal labels.

4.4 LSTM-Based Goal Recognition

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. Each action in the sequence is encoded with a single distributed representation by concatenating the ten action property-based representations that constitute the action. The process for creating this distributed action representation is shown in Figure 2B. The action property embedding matrix manages the linear mapping from a discrete action property index to a d-dimensional continuous vector space. Since a distributed action representation is created by concatenating all ten d-dimensional vectors, the size of a distributed representation becomes ten times d for the action, x_t.

At recognition time, a sequence of actions is sequentially fed into the LSTM model 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 2A) is used to predict the most likely goal for the sequence of actions in a softmax layer, which is interpreted as a calculation of posterior probabilities of goals.

For hyperparameter optimization of our LSTM-based goal recognition models, we perform a grid search and empirically determine an optimal configuration of the networks through cross validation. In this work, 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 [LeCun *et al.*, 2015] among {0.5, 0.75}, all of which have significant potential to influence the proposed LSTM-based goal recognition performance. 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 a stochastic optimizer. 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 pre-trained embedding, an additional LSTM is trained to learn initial action property representations prior to building the LSTM goal recognizer. For this model, we use a sigmoid layer for predicting the next action for a given sequence of actions along with the binary cross entropy for the loss function, casting it as a multi-label classification task predicting the ten features that constitute an action. All other hyperparameters are identically applied as in the goal recognition model, except that the pre-training process does not use early stopping but instead uses a fixed number of epochs (100), since this unsupervised approach is not likely to exhibit overfitting, and the pre-trained representations will be fine-tuned in the following supervised learning step for the goal recognition model.

5 Evaluation

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% ("Speaking with the camp nurse") to 26.6% ("Running laboratory test on contaminated food"), which is the majority class baseline [Ha *et al.*, 2011]. The goal recognition performance of LSTMs is evaluated relative to two competitive baselines: (1) n-gram encoded feedforward neural networks (FFNN) pre-trained with stacked denoising autoencoders [Min *et al.*, 2014], a previous state-of-the-art approach and (2) Markov logic networks (MLN) reported in [Baikadi *et al.*, 2014], another previous state-of-the-art approach. All models are evaluated using 10-fold cross validation using the same data split for pairwise comparisons.

Goal recognition model performance is measured using *accuracy rate, convergence rate,* and *convergence point* [Baikadi *et al.*, 2014; Min *et al.*, 2014] in order to capture the multi-dimensional nature of goal recognition performance. Convergence rate refers to the percentage of observation sequences in which the final goal prediction is correct. A higher number is better for this metric. Convergence point refers to the proportion of an action sequence after which the goal recognizer begins to make correct goal predictions for the remainder of the sequence. More formally, convergence point is calculated by $\sum_{i=1}^{m}(k_i/n_i)\,/m$, 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. This metric indicates how early a goal recognizer can inform the system about the player's current goal. A lower score is better for this metric.

In the evaluation, the FFNN model was initialized using hyperparameter search. The MLN model was developed through manual feature engineering by a domain expert. Two sets of LSTM results are reported. The results vary based upon whether the LSTM utilized pre-training of action property representations. The two models are denoted as LSTM-S (supervised embedding) and LSTM-P (pre-trained embedding).

Table 1 presents results for the three computational goal recognition approaches. LSTM-S achieves the highest goal recognition accuracy of 66.3%. In this table, only the model that achieves the highest cross-validation accuracy per approach is reported. Convergence rate and convergence point are calculated based on the model with the highest accuracy.

	MLN	FFNN	LSTM-P	LSTM-S
Accuracy Rates	54.63	62.30	66.25	**66.34**
Convergence Rates	50.06	70.90	76.25	**78.54**
Convergence Points	**35.86**	42.82	68.13	69.66

Table 1. Accuracy rate (%), convergence rate (%) and convergence point (%) results for the best performing LSTM, FFNN, and MLN models under 10-fold cross validation.

Both the LSTM-S and LSTM-P models achieve the greatest accuracy when configured with an action property embedding size of 20, 100 hidden units, and a dropout rate of 0.75. For pairwise comparisons of the three computational approaches (MLN, FFNN, and LSTM-S), we run a Friedman non-parametric statistical test followed by Wilcoxon non-parametric post-hoc tests based on the fold-level cross-validation results. The Friedman test finds a significant difference in accuracy rates across the models, $\chi^2(2)=14.6$, $p=0.001$. Wilcoxon tests applied with the Holm-Bonferroni correction indicate that there are significant improvements in accuracy rates for LSTM-S over FFNN ($p=0.022$), LSTM-S over MLN ($p=0.005$), and FFNN over MLN ($p=0.013$).

Additionally, LSTM-based goal recognition is found to make more accurate predictions on the last action (LSTM-S: 78.5%, LSTM-P: 76.3%) than competitive baselines. On the other hand, LSTMs are found to have the latest convergence point, which indicates that the models' predictions slowly converge to the correct goal after observing longer sequences of actions. This result can be partially explained by the inherent tension between a high convergence rate and low convergence point. 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.

It is notable that LSTM-S (66.34%) and LSTM-P (66.25%) are competitive to each other. We did not find evidence of a statistically significant difference in predictive accuracy between the two models. Considering that both models achieve the highest accuracy with the same set of network hyperparameters, it may be that hyperparameter selection has a more significant influence on model performance than pre-training in inducing distributed action representations. To examine the individual impacts of the three hyperparameters optimized through grid search, the two highest performing LSTMs, both with (embedding size, hidden units, dropout rate) of (20, 100, 0.75) are compared to the their alternate models in terms of the 10-fold cross-validation results (Table 2).

Among the hyperparameters, only the number of LSTM hidden units commonly results in sizable difference in accuracy rate. This finding is echoed in [Bergstra and Bengio, 2012], which found that only a few hyperparameters substantially impact model performance. Identifying important hyperparameters informs directions for future work to obtain more accurate goal recognizers.

	Dropout (0.5)	Hidden units (200)	Embedding (10)	Highest
LSTM-S	65.13	**62.21**	65.36	**66.34**
LSTM-P	64.44	**63.31**	64.57	**66.25**

Table 2. Averaged hyperparameter level accuracy rates (%) where *Dropout (0.5)* denotes (20, 100, 0.5), *Hidden units (200)* denotes (20, 200, 0.75), and *Embedding (10)* denotes (10, 100, 0.75).

6 Conclusion

Open-world digital games provide an ideal laboratory for investigating computational models of goal recognition. A key challenge in goal recognition 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 paper has introduced a goal recognition framework that leverages long short-term memory networks (LSTMs) and distributed action representations, which significantly outperforms previous state-of-the-art approaches with respect to predictive accuracy. Additionally, LSTMs demonstrate higher convergence rates over competitive deep learning and non-deep learning-based baselines. The LSTM goal recognition framework achieves these results without requiring labor-intensive manual feature engineering often required by other machine learning approaches.

In the future it will be important to investigate forms of recurrent neural networks that use fewer parameters. In addition, model optimization can be examined either by utilizing different optimizers or identifying an optimal configuration of hyperparameters. Further, it will be important to devise additional metrics that quantify unmeasured aspects of goal recognizer performance. For example, in spite of its broad utilization, the convergence point metric does not thoroughly measure models' early prediction capability because it ignores action sequences that do not converge. It will also be important to investigate the impact of distributed action representations on LSTM goal recognition performance. In this paper, we did not isolate the effects of the LSTM models' action-property embedding layer. Finally, it will be important to investigate how goal recognition models operate at run-time within open-world digital game environments to understand how goal recognizers can most effectively drive gameplay personalization in player-adaptive games.

References

[Baker *et al.*, 2009] C. Baker, R. Saxe, and J. Tenenbaum. Action understanding as inverse planning. *Cognition*, 113(3):329–349, 2009.

[Baikadi *et al.*, 2014] A. Baikadi, J. Rowe, B. Mott, and J. Lester. Generalizability of Goal Recognition Models in Narrative-Centered Learning Environments. In *Proceedings of the 22nd Conference on User Modeling, Adaptation, and Personalization*, pages 278–289, 2014.

[Bengio *et al.*, 2003] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. 2003. A Neural Probabilistic Language Model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.

[Bergstra and Bengio, 2012] J. Bergstra and Y. Bengio. Random Search for Hyper-Parameter Optimization. *Journal of Machine Learning Research*, 13:281–305, 2012.

[Bisson *et al.*, 2015] F. Bisson, H. Larochelle, and F. Kabanza. Using a Recursive Neural Network to Learn an Agent's Decision Model for Plan Recognition. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, pages 918–924, 2015.

[Duong *et al.*, 2009] T. Duong, D. Phung, H. Bui, and S. Venkatesh. Efficient duration and hierarchical modeling for human activity recognition. *Artificial Intelligence*, 173(7-8):830–856, 2009.

[Geib and Goldman, 2009] C. Geib and R. Goldman. A probabilistic plan recognition algorithm based on plan tree grammars. *Artificial Intelligence*, 173(11):1101–1132, 2009.

[Graves, 2012] A. Graves. *Supervised Sequence Labelling with Recurrent Neural Networks*. Studies in Computational Intelligence, Springer, 2012.

[Ha *et al.*, 2011] E. Ha, J. Rowe, B. Mott, and J. Lester. Goal Recognition with Markov Logic Networks for Player-Adaptive Games. In *Proceedings of the Seventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, pages 32–39, 2011.

[Harrison *et al.*, 2015] B. Harrison, S. Ware, M. Fendt, and D. Roberts. A Survey and Analysis of Techniques for Player Behavior Prediction in Massively Multiplayer Online Role-Playing Games. *IEEE Transactions on Emerging Topics in Computing*, 3(2): 260–274, 2015.

[Hochreiter and Schmidhuber, 1997] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1–32, 1997.

[Kabanza *et al.*, 2013] F. Kabanza, J. Filion, A. Rezak Benaskeur, and H. Irandoust. Controlling the hypothesis space in probabilistic plan recognition. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pages 2306–2312, 2013.

[Kautz and Allen, 1986] H. Kautz and J. Allen. Generalized Plan Recognition. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 32–38, 1986.

[Le and Mikolov, 2014] Q. Le and T. Mikolov. Distributed Representations of Sentences and Documents. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1188–1196, 2014.

[LeCun *et al.*, 2015] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553): 436–444, 2015.

[Lopes and Bidarra, 2011] R. Lopes and R. Bidarra. Adaptivity challenges in games and simulations: A survey. *IEEE Transactions on Computational Intelligence and AI in Games*, 3(2):85–99, 2011.

[Min *et al.*, 2014] W. Min, E. Ha, J. Rowe, B. Mott and J. Lester. Deep Learning-Based Goal Recognition in Open-Ended Digital Games. In *Proceedings of the 10th AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, pages 37–43, 2014.

[Mott *et al.*, 2006] B. Mott, S. Lee, and J. Lester. Probabilistic goal recognition in interactive narrative environments. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 187–192, 2006.

[Riedl and Bulitko, 2013] M. Riedl and V. Bulitko. Interactive Narrative: An Intelligent Systems Approach. *AI Magazine*, 34(1):67–77, 2013.

[Rowe *et al.*, 2011] J. Rowe, L. Shores, B. Mott, and J. Lester. Integrating learning, problem solving, and engagement in narrative-centered learning environments. *International Journal of Artificial Intelligence in Education*. 21(1-2):115–133, 2011.

[Shaker *et al.*, 2015] N. Shaker, J. Togelius, and M. J Nelson. *Procedural Content Generation in Games: A Textbook and an Overview of Current Research*. Springer, 2015.

[Singla and Mooney, 2011] P. Singla and R. Mooney. Abductive Markov logic for plan recognition. In *Proceedings of the 25th National Conference on Artificial Intelligence*, pages 1069–1075, 2011.

[Socher *et al.*, 2010] R. Socher, C. Manning, and A. Ng. Learning continuous phrase representations and syntactic parsing with recursive neural networks. In *Proceedings of the NIPS-2010 Deep Learning and Unsupervised Feature Learning Workshop*, pages 1–9, 2010.

[Squire, 2008] K. Squire. Open-ended video games: A model for developing learning for the interactive age. *The ecology of games: Connecting youth, games, and learning*, pages 167–198, 2008.

[Sukthankar *et al.*, 2014] G. Sukthankar, C. Geib, H. Bui, D. Pynadath, and R. Goldman. *Plan, activity, and intent recognition: Theory and practice*. Newnes, 2014.

[Yannakakis *et al.*, 2013] G. Yannakakis, P. Spronck, D. Loiacono, and E. André. Player Modeling. *Artificial and Computational Intelligence in Games*, 6:45–59, 2013.