Learning Neuro-Symbolic World Models with Conversational Proprioception

Anonymous ACL submission

Abstract

The recent emergence of Neuro-Symbolic Agent (NeSA) approaches to natural language-based interactions calls for the investigation of model-based approaches. In contrast to model-free approaches, which existing NeSAs take, learning an explicit world model has an interesting potential especially in the explainability, which is one of the key selling points of NeSA. To learn useful world models, we leverage one of the recent neuro-symbolic architectures, Logical Neural Networks (LNN). Here, we describe a method that can learn neuro-symbolic world models on the TextWorld-Commonsense set of games. We then show how this can be improved further by adding a proprioception that gives better tracking of the internal logic state and model. Also, the game-solving agents performance in a TextWorld setting shows a great advantage over the baseline with 85% average steps reduction and $\times$2.3 average scoring.

1 Introduction

Recent emergence of neuro-symbolic (NS) approaches include natural language-based sequential decision making (Kimura et al., 2021b; Chaudhury et al., 2021; Kimura et al., 2021a). They propose a model-free approach of learning a logical policy, and tested with interactive-text games (Narasimhan et al., 2015; Côté et al., 2018; Hausknecht et al., 2020; Murugesan et al., 2021), which have become an interesting benchmark in the intersection of natural language processing and sequential decision making. NS approaches give the direct explainability of what is learned and allow natural integration of external knowledge as logic. Despite that, existing NS approaches are of model-free reinforcement learning (RL) but it would be useful if we could have model-based approaches that are potentially more sample efficient and can reach higher cumulative rewards as shown by neural world models (Hafner et al., 2019; Łukasz Kaiser et al., 2020). In contrast to these, a logical world model learned using NS approaches would allow an agent to use logical reasoning which enables us to obtain a trace of logical steps for better explainability. In fact, several sets of benchmarks and game environments have been proposed such as TextWorld (Côté et al., 2018), Jericho (Hausknecht et al., 2020) and TextWorld Commonsense (TWC) (Murugesan et al., 2021), which are far too complicated to solve without reasoning and common sense, compared to the original game setting (Narasimhan et al., 2015). Also, existing implementation of NS approaches do not really interpret a natural language but rely on the logical facts provided from the game engines.

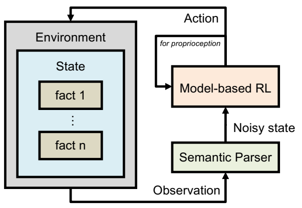

Figure 1: Overview of proposed method

In this paper, we focus on the problem of learning logical world models in NS methods. The main research question to be addressed is then how we can learn such models for text-based games using a general semantic parser. As a state-of-the-art interactive-text agent, GATA (Adhikari et al., 2020) constructs belief graphs used to enhance deep RL methods. In contrast to understanding the world state in a latent space, we want to explicitly use the logical world models to plan optimal action sequences and to provide direct explainability of the decision making policy. For the explainability purpose, we leverage general semantic parsing, following one of the early work constructing knowledge graphs (Ammanabrolu and Riedl, 2019).

An overview of our proposed method is depicted by Figure 1. The left side depicts that the environment state can be sufficiently approximated as a set of logical facts. Continuing in the bottom right, the agent can get textual observations of the environment. We assume that we have a *semantic parser* (Drozdov et al., 2022) that converts these observations into a logical form. In the real situation the semantic parsing is good, but won't be perfect, hence we require that our agent should be capable of handling noisy logical states. From such states, our agent should produce suitable actions for accomplishing its tasks in the environment.

The main contributions of this paper are: the proposal of a novel world model-learning method with a neuro-symbolic approach, and its experimental results with TWC.

2 Problem Definition

Text-based games are often modelled with the RL problem setting in mind as Partially Observable - Markov Decision Processes (PO-MDP) (Côté et al., 2018; Hausknecht et al., 2020). As a first approach, we add an assumption - that the semantic parser can remove partial observability and that we are dealing with an MDP. At each time step the agent uses the information in a state, s, to take an action, a, which transitions the state to the new state, s' according to the state transition function T such that $s' = T(s, a)$. While acting in this environment the agent also gets rewards, r, according to an unknown reward function, R, such that $r = R(s, a)$. In the model-free RL setting, the agent learns a policy or value function which directly governs the actions. Here, we are interested in the model-based RL setting where the agent learns a model of the world which usually consists of both T and R. This model can then be used with planning methods to find the optimal actions.

Based on the classical model-based RL setting, our problem has two more important specifications. First, we assume that our environment is *relational*, similar to (Lang et al., 2012). This means that all actions and states are composed of relational logic. They may be in the propositional form but there must be a corresponding lifted form that has a consistent *meaning*. For example, the propositional state, *on(book,table)* can be abstracted or *lifted* into *on(x,y)* with predicate, on, and the variables, (x, y). The first assumption is that all states and actions handled by the agent are in this relational lifted form. This assumption can be handled as a design specification of the semantic parser. The second assumption is that the goal state is given. This is a weaker assumption that is already used in current RL research, the so-called *goal-conditioned RL*. Here, it allows us to concentrate only on learning T since R is no longer required for planning when we are given the goal state.

3 Learning Logical World Models

The problem of learning logical rules that explain a given set of logical examples can be cast into the general problem called Inductive Logic Programming (ILP) (Muggleton and De Raedt, 1994). What needs to be done is then to cast our relational model-based RL problem into ILP form. But before going into that detail, it is important to note that relying on classical ILP has significant failings. In particular, it is not well suited to noisy data to the extent that a single erroneous data point may cause the whole system to fail.

However, newer methods that leverage neural networks have shown great promise on working even with noisy data (Evans and Grefenstette, 2018). These are sometimes called neural ILP, differentiable ILP or neuro-symbolic ILP. These advances are the main impetus for us to research on the learning of logical world models.

We may use any such ILP method that is noise-resistant but here we use the Logical Neural Network (LNN) (Riegel et al., 2020) as a Neuro-Symbolic AI framework. It is an end-to-end differentiable system that enables scalable gradient-based learning and it has a real-valued logic representation of each neuron that enables logical reasoning (Riegel et al., 2020).

Action ILP with LNN

Now, getting back to the task of expressing our relational model-based RL problem as ILP, we first gather data samples which are triples of lifted logic, (s, a, s'). This is gathered by using an exploration policy to generate actions. Here, we used a policy that uniformly randomly samples the action space but better exploration methods may be used, such as that outlined in (Lang et al., 2012). This data collection may be done in an offline or online RL setting but we assume that a large enough *batch* is available in the online RL setting before we start the learning procedure.

Given a batch of data samples, the learning procedure must produce an estimate of T. This T will be the *hypothesis* to be generated by our ILP. This is a set of logical rules that best fits the data. To make learning more efficient we need to narrow down the definition of T. Because we are ultimately interested in using T for planning, we define it as a set of planning operators where each one is a quadruple of $(\alpha, \beta, \gamma, \sigma)$. Each element is a set of logical conditions. The conditions (α, β) are preconditions where α are conditions that must be true for the action to be executable, β are ones that must be false. The conditions (γ, σ) are post-conditions where γ are ones made true by the action and σ are ones made false. These conditions are the lifted logic statements that comprise a state, s, and the set of all possible conditions is P.

We model each of the operator elements as an LNN conjunction operator whose inputs are P. The LNN learning procedure can learn weights for each of these inputs that correspond to real-valued logic (Riegel et al., 2020; Sen et al., 2021). For the LNNs of α and β, the inputs are given the corresponding logical values of the conditions in s. The output is true when action, a, corresponds and $s \neq s'$ otherwise it is false. For the LNNs of γ and σ, the inputs are given the logical values corresponding to the difference in the conditions of s and s' such that γ are the the conditions made true and σ those that are made false. The output is true when action, a, corresponds otherwise it is false.

Using these inputs and outputs to the LNN, gradient-based optimization can be used for supervised learning (Riegel et al., 2020; Sen et al., 2021). When learning converges, we have a set of weights for each of the corresponding elements. These may be interpreted as probabilistic transitions but here we simply threshold them and maintain a deterministic transition system for our final estimate of T. Given this operator transition model and the goal, we can be in any state and use classical planning methods to find a series of actions to reach the goal.

Conversational Proprioception

Proprioception (Tuthill and Azim, 2018) is the sensation of body position and movement critical to human experience, while it is typically absent from conscious perception. This concept is used in imitation learning (Torabi et al., 2019) and in robotics (Cong et al., 2022), however to our knowledge it has not been proposed for text-based games or tasks with logical state represenatations.

In general, proprioception is a prediction of the next state, $s' = \hat{T}(s, a)$, based on the existing knowledge of one's body dynamics in the form of the transition model estimate, $\hat{T}$, the current state, s, and the action taken, a. This additional information is crucial to help us disambiguate and better locate the next state. For our task where T is a logical model, we propose to augment our learned T with a set of proprioception rules, $\epsilon(s, a)$, such that our T will now be defined as $(\alpha, \beta, \gamma, \sigma, \epsilon(s, a))$. For our agent, we define ϵ very generally such that it only consists of 2 rules. First, it tracks state-action pairs that were already tried and augments the state with this information. Second, it adds a precondition that prevents state-action pairs from repeating which is a common problem of conversational agents. These 2 rules are general enough to apply to any TWC environment and possibly beyond. We leave the design of further proprioception rules as a possible future work.

4 Experiment and Discussion

For evaluating the quality of world model learned, we first qualitatively analyze the learned action models. Then we see the interactive-text agent performance with a TextWorld benchmark. In this paper, we experiment on the TextWorld Commonsense (TWC) set of games (Murugesan et al., 2021) with the same experimental settings.

Once we have a logical world model, we can use it with a planner. Here, we use the Fast-Downward systsem (Helmert, 2006). For convenience we convert the learned operators into the PDDL (Planning Domain Definition Language) format by combining $(\alpha, \beta, \epsilon)$ into the preconditions and (γ, σ) into the effects. We also augment the state with $\epsilon(s, a)$.

Learned Models

We confirmed that the world models were meaningfully learned by any of model-based approaches. Figure 2 shows example learned action models in a converted PDDL form for an action *insert_into* (insert XX into YY) by model-based approaches from AMR-based logical facts. For our results, we first show some examples of the learned rules in our logical world model in Figure 2. Here, we can visually inspect the validity of the rules. For example with the left case, the effect would be that the object *v0* is at/in/on the container *v1* (*has_location-2*) but now it is no longer in the inventory (*carry-1*). This level of explainability is inherent in logical models although it requires careful inspection.

The effect of proprioception can be seen in the right-hand side of Figure 2. The predicate of *tried_insert_into* is from an AMR-based pred-

Table 1: Scores on the TextWorld Commonsense(TWC) set of games

	Semantic parsing	Handicap	Easy		Medium		Hard	
			Valid	Test	Valid	Test	Valid	Test
TWC agent (DL-only) [AAAI 2021]	Word embedding	(these are common) Admissible action Inventory Curated Common Sense	17.65 ± 3.62 85% ± 7%	18.00 ± 3.24 87% ± 5%	37.18 ± 4.86 72% ± 7%	43.08 ± 4.13 54% ± 17%	49.36 ± 7.50 46% ± 10%	49.96 ± 0.00 22% ± 0%
Model-free NeSA based on [EMNLP 2021]	Skipped	Game-engine facts	-	15.00 100%	-	28.60 100%	-	-
Model-free NeSA (REINFORCE)	AMR-based facts	-	-	32.28 ± 3.24 63% ± 5%	-	43.68 ± 5.36 38% ± 25%	-	49.48 ± 1.04 28% ± 13%
Planning (Model-based NeSA)	Skipped	Action transition Game-engine facts	2.4 100%	2.4 100%	4.4 100%	3.6 100%	13.6 100%	14.0 100%
Model-based NeSA (Learned action transition)	Skipped	Game-engine facts	2.4 ± 0.0 100%	2.4 ± 0.0 100%	4.4 ± 0.0 100%	3.6 ± 0.0 100%	13.6 ± 0.0 100%	28.4 ± 0.0 60.6%
Model-based NeSA	AMR-based facts	-	21.4 ± 0.0 57.1%	21.2 ± 0.0 42.9%	31.6 ± 0.0 38.5%	31.6 ± 0.0 50.0%	42.8 ± 0.0 20.6%	42.8 ± 0.0 24.2%
Model-based NeSA w/ proprioception	AMR-based facts	-	3.6 ± 0.0 100%	**4.0 ± 0.0 100%**	7.6 ± 0.0 100%	**5.6 ± 0.0 100%**	33.2 ± 0.0 64.7%	42.8 ± 0.0 24.2%

icate *insert_into* but with the intention modality of the agent, which is encoded in first-order logic. This recognition of an already-performed action *insert_into* should contribute to avoiding repeatedly performing failed actions.

TWC Performance

It would be more interesting if we take altogether to see if the learned rules allow us to plan optimal actions in the world. To answer this, we present our results in Table 1. Here, there are 7 methods: the first row is a deep-learning-only method, second and third are model-free neuro-symbolic methods, forth is a planning result which is an upper bound, fifth is a model-based RL method with game-engine facts, sixth is a model-based RL method, and seventh is a model-based RL method with proprioception.

For comparison, we show the best performing deep RL agent in (Murugesan et al., 2021) and the optimal agent using perfect game knowledge. Note that we have additional assumptions differing from the plain deep RL setting of the original setup in (Murugesan et al., 2021) but we give this as a reference on the potential improvement our overall approach might provide. The TWC games are categorized into Easy-Medium-Hard with a validation and testing set for each as shown in the columns.

Our results show that planning on our learned model can produce the same optimal actions for all the Easy and Medium games and for the validation set of the Hard games. An interesting limitation appears in the test set of the Hard games wherein novel predicates appear in the test set that do not appear in any of the training or validation set. This is a current limitation of our system.

The significant effects of AMR-originated noise or lack of information can be seen in the last two rows. However, thanks to the proprioception (the last row), it has been improved to be competitive to the model-based approach from game-engine provided logical facts (3rd-last row).

```
(:action insert_into
:parameters (?v0 ?v1)
:precondition
(and
   (carry-1 ?v0)
   (not (has_location-2 ?v0 ?v1)))
:effect
(and
   (has_location-2 ?v0 ?v1)
   (not (carry-1 ?v0))))
```

```
(:action insert_into
:parameters (?v0 ?v1)
:precondition
(and
   (not (tried_insert_into ?v0 ?v1))
   (carry-1 ?v0)
   (not (has_location-2 ?v0 ?v1)))
:effect
(and
   (has_location-2 ?v0 ?v1)
   (not (carry-1 ?v0))
   (tried_insert_into ?v0 ?v1)))
```

Figure 2: Examples of the learned action models

5 Conclusion

We proposed a model-based RL agent for text-based games which comprises of a semantic parser producing logical states, a neuro-symbolic ILP module for learning logical world models, and an off-the-shelf planning system to produce optimal actions in the game world. We presented results and experiments here on the key component which learns the logical world models.

6 Limitations

The experimental environment we used for testing our agents gives artificially generated natural language text, whose distribution of vocabulary, syntax, and semantic frames is controlled and limited to what the natural language text generators can provide. While we tried to include out of vocabulary for entities in our experiments, applying the proposed approach to natural language text in wild, such as chatbots working with human, must be faced with issues such as out of vocabulary for relations, etc. We believe, however, approaching from controlled "wildness" is an important direction of the work for interactive-text agents.

References

Ashutosh Adhikari, Xingdi Yuan, Marc-Alexandre Côté, Mikuláš Zelinka, Marc-Antoine Rondeau, Romain Laroche, Pascal Poupart, Jian Tang, Adam Trischler, and Will Hamilton. 2020. Learning dynamic belief graphs to generalize on text-based games. In *Advances in Neural Information Processing Systems*, volume 33, pages 3045–3057. Curran Associates, Inc.

Prithviraj Ammanabrolu and Mark Riedl. 2019. Playing text-adventure games with graph-based deep reinforcement learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3557–3565, Minneapolis, Minnesota. Association for Computational Linguistics.

Subhajit Chaudhury, Prithviraj Sen, Masaki Ono, Daiki Kimura, Michiaki Tatsubori, and Asim Munawar. 2021. Neuro-symbolic approaches for text-based policy learning. In *Conference on Empirical Methods in Natural Language Processing*, pages 3073–3078.

Lin Cong, Hongzhuo Liang, Philipp Ruppel, Yunlei Shi, Michael Görner, Norman Hendrich, and Jianwei Zhang. 2022. Reinforcement learning with vision-proprioception model for robot planar pushing. *Frontiers Neurorobotics*, 16:829437.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Ruo Yu Tao, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. 2018. Textworld: A learning environment for text-based games. *CoRR*, abs/1806.11532.

Andrew Drozdov, Jiawei Zhou, Radu Florian, Andrew McCallum, Tahira Naseem, Yoon Kim, and Ramón Fernandez Astudillo. 2022. Inducing and using alignments for transition-based AMR parsing. *CoRR*, abs/2205.01464. To appear in NAACL-22.

Richard Evans and Edward Grefenstette. 2018. Learning explanatory rules from noisy data. *Journal of Artificial Intelligence Research*, 61:1–64.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. 2019. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR.

Matthew Hausknecht, Prithviraj Ammanabrolu, Marc-Alexandre Côté, and Xingdi Yuan. 2020. Interactive fiction games: A colossal adventure. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7903–7910.

Malte Helmert. 2006. The fast downward planning system. *Journal of Artificial Intelligence Research*, 26:191–246.

Daiki Kimura, Subhajit Chaudhury, Masaki Ono, Michiaki Tatsubori, Don Joven Agravante, Asim Munawar, Akifumi Wachi, Ryosuke Kohita, and Alexander Gray. 2021a. LOA: Logical optimal actions for text-based interaction games. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 227–231.

Daiki Kimura, Masaki Ono, Subhajit Chaudhury, Ryosuke Kohita, Akifumi Wachi, Don Joven Agravante, Michiaki Tatsubori, Asim Munawar, and Alexander Gray. 2021b. Neuro-symbolic reinforcement learning with first-order logic. In *Conference on Empirical Methods in Natural Language Processing*, pages 3505–3511.

Tobias Lang, Marc Toussaint, and Kristian Kersting. 2012. Exploration in relational domains for model-based reinforcement learning. *Journal of Machine Learning Research*, 13(119):3725–3768.

Stephen Muggleton and Luc De Raedt. 1994. Inductive logic programming: Theory and methods. *The Journal of Logic Programming*, 19:629–679.

Keerthiram Murugesan, Mattia Atzeni, Pavan Kapanipathi, Pushkar Shukla, Sadhana Kumaravel, Gerald Tesauro, Kartik Talamadupula, Mrinmaya Sachan, and Murray Campbell. 2021. Text-based rl agents with commonsense knowledge: New challenges, environments and baselines. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(10):9018–9027.

Karthik Narasimhan, Tejas Kulkarni, and Regina Barzilay. 2015. Language understanding for text-based games using deep reinforcement learning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1–11, Lisbon, Portugal. Association for Computational Linguistics.

Ryan Riegel, Alexander G. Gray, Francois P. S. Luus, Naweed Khan, Ndivhuwo Makondo, Ismail Yunus Akhalwaya, Haifeng Qian, Ronald Fagin, Francisco Barahona, Udit Sharma, Shajith Ikbal, Hima Karanam, Sumit Neelam, Ankita Likhyani, and Santosh K. Srivastava. 2020. Logical neural networks. *CoRR*, abs/2006.13155.

Prithviraj Sen, Breno W. S. R. de Carvalho, Ryan Riegel, and Alexander G. Gray. 2021. Neuro-symbolic inductive logic programming with logical neural networks. *CoRR*, abs/2112.03324.

Faraz Torabi, Garrett Warnell, and Peter Stone. 2019. Imitation learning from video by leveraging proprioception. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 3585–3591. ijcai.org.

John C. Tuthill and Eiman Azim. 2018. Primer: Proprioception. *Current Biology*, 28:R187–R207.

Łukasz Kaiser, Mohammad Babaeizadeh, Piotr Miłos, Błażej Osiński, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, Afroz Mohiuddin, Ryan Sepassi, George Tucker, and Henryk Michalewski. 2020. Model based reinforcement learning for atari. In *International Conference on Learning Representations*.