Symbolic Relation Networks for Reinforcement Learning

Dhaval Adjodah
IBM Watson Research
Yorktown Heights, NY
dval@mit.edu

Tim Klinger
IBM Watson Research
Yorktown Heights, NY
tklinger@us.ibm.com

Joshua Joseph
MIT Quest for Intelligence
Cambridge, MA
jmjoseph@mit.edu

Abstract

In recent years, reinforcement learning techniques have enjoyed considerable success in a variety of challenging domains, but are typically sample inefficient and often fail to generalize well to new environments or tasks. Humans, by contrast, are able to learn robust skills with orders of magnitude less training. One hypothesis for this discrepancy is that humans view the world in terms of objects and relations between them. Such a bias may be useful reducing sample complexity and improving interpretability and generalization. In this paper, we present a novel relational architecture which has multiple neural network sub-modules called *relational units* which operate on objects and output values in the unit interval. Our model transforms the input state representation into a relational representation, which is then supplied as input to a Q-learner. Experiments on a goal-seeking game with random boards show better performance over several baselines: a multi-headed attention model, a standard MLP, a pixel MLP and a symbolic RL model. We also find that the relations learned in the network are interpretable.

1 Introduction

Deep Learning techniques have proven invaluable in tackling many problems which previously required tricky feature engineering. Reinforcement learning, in particular, has benefited from deep learning, achieving notable successes in a variety of challenging domains [13, 18]. Unfortunately, these approaches are not always sample efficient, often requiring hundreds of thousands of episodes or more of training. More troubling still, current techniques often do not yield general or transferable models [8]. Humans, by contrast, are much better at learning efficiently, learning generalizable strategies from orders of magnitude less data [11]. One potential explanation for this success is that humans view the world in terms of objects and their relations [2]. For example, in viewing an image we may see that the cup is on the table (a *binary relation* between two objects) or the wall is green (a *unary relation* on one object).

This work investigates the effectiveness of the *relational hypothesis* in RL (discussed in more detail in [2]) that a relational inductive bias leads to improvements in learning efficiency, effectiveness, generality and interpretability. Recent work [16, 20, 2, 14] has added support for the relational hypothesis in domains such as visual Q/A [16] and game playing which requires reasoning [20].

In this paper, we present a novel architecture in which each relation is learned by a separate relational unit which operates on a fixed number of objects (1 or 2 in our experiments) and produces a scalar value in the unit interval[1]. We experiment on the stochastic goal-seeking game of [7] in which each episode has a random placement of multiple goal objects, some with positive rewards and some with negative rewards. This game, while simple, is challenging because the agent must learn a

[1]The logical community refers to these as 'fuzzy' relations.

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

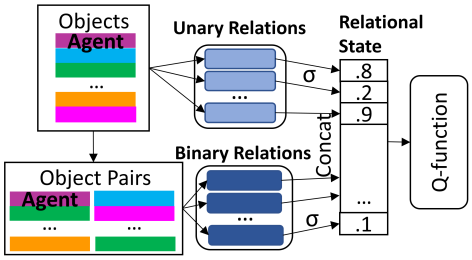

Figure 1: The SRN: Object state representations are provided as input and passed directly to each unary unit. Object pairs are computed and passed to each binary unit. The output of the relational units is collected in a *relational state* and supplied as input to a Q-function.

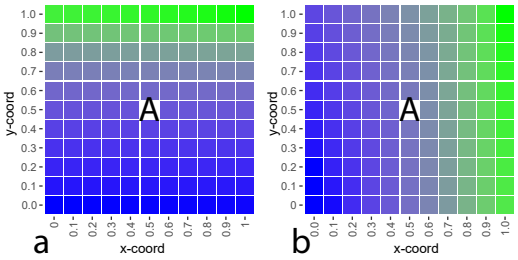

Figure 2: Visualization of the output of two binary units when the agent is placed at the center and a goal is placed on each tile which (a) shows the degree of 'aboveness' of the goal from the agent, and (b) shows degree of 'rightness' of the goal from the agent.

location-relative policy to succeed. We find that the relations learned are interpretable. Our model has the ability to learn a binary relation that indicates to what degree an object is, for example, to the right of the agent (it learns relative, not absolute, position relations), while another binary relation can learn 'aboveness' relative to the agent, and a unary relation can be learned that indicates the type of each object. We show that with a suitable number of relational units (a hyper-parameter of our model), we can achieve performance gains over a multi-headed attention model [20], a standard MLP, a pixel MLP, and the symbolic RL model of [7] on this problem.

2 Related Work

Early work in the area of relational reasoning used symbolic representations and tabular learning [5]. More recently [16] introduced a simple relational model with a single relational function for supervised relational learning. A version using the multi-headed attention network [19] is used in [20] to learn a policy and value function. Here, each attention head computes a relation by a applying a scaled dot-product between pairs of objects, normalizing with softmax, and using the result as weighting to update the object representations. Our approach differs in several respects. First, we consider both unary (single argument) and binary (two argument) relations. Second, we allow multiple modules for each type of relation (unary and binary). Third, we do not compute a probability distribution over the attention scores between an object and every other, but instead compute values in the unit interval for each comparison independently. This encodes the intuition that the same relation may hold independently between an object and many others (for example 'above'). Finally, we do not use attention scores to update the object representations but instead supply the concatenated relational values (which we call the *relational state*) directly to the Q-function. Our relational unit can be viewed as implementing a fuzzy logical relation between its arguments. In this respect, our approach connects to the neuro-symbolic reasoning work of [17, 15, 6, 4, 1].

3 Model

Since we are focusing on the relations between objects in this work, we assume that the objects have already been extracted upstream (for example using the approach of [7] or [10]), and we have available an $m \times 4$ tensor of m object representations ($m - 1$ goals, 1 agent). Each object representation is of size 4, consisting of the (x, y) coordinates, the type, and a binary 'existence' marker of whether a goal has been captured by the agent. We consider only agent-object pairs and assume that the agent representation is supplied first within the tensor of objects.

As shown in Figure 1, our architecture explicitly represents multiple unary (single object) and binary (two objects) relations on objects in the state[2] and concatenates the output values of those relations to form a relational state which is supplied to the Q-function. In essence, our model can be seen

[2]Relations between more than two objects are straightforwardly incorporated into our model but due to the combinatorial increase in computational expense we elected to not considered them in our current work.

as taking the state and pre-processing it into a relational state that can be used with any deep RL algorithm. Specifically, we use differentiable neural relation modules (units which take a vector of objects as input, and output a value in $[0, 1]$) and concatenate their outputs to pass as input to the state action value layers. We call this architecture a *Symbolic Relation Network* (SRN). The unary unit is of the form: $\sigma(W_1 \texttt{ReLU}(W_2 A^T + b_1) + b_2)$ where σ is the sigmoid function, $W_1 \in \mathcal{R}^{1\times k}$ and $W_2 \in \mathcal{R}^{k\times n}$ are weight tensors, $A \in \mathcal{R}^{m\times n}$ is the dimensional input object tensor with m n-dimensional objects, and $b_1 \in \mathcal{R}^{k\times m}$ and $b_2 \in \mathcal{R}^{1\times m}$ are biases. The hidden dimension k is taken to be the same as the object dimension n which is 4 in our experiments. The output is of dimension m, one relation value in $[0, 1]$ for each of the m input objects (i.e. the unit learns a representation of each object). The binary unit is constructed similarly but we form the input tensor by concatenating N pairs of objects to form an $N \times 2n$ input tensor. The state-action value module is a simple linear layer which takes in the concatenation of the relational unit outputs. In implementation, we find that our SRN model can be quite small (< 1000 total parameters) and fast.

4 Methods

In this work, we only vary the neural network model that is used to approximate the Q-function $Q(s, a)$, allowing us to fairly compare the different models, while keeping the environment the same[3]. We test the SRN and baseline models using the environment introduced by [7] where an agent can move up, down, left and right and must reach 'good' goals (green circles) that provide a positive reward (+10), and avoid 'bad' goals (red circles) that provide a negative reward (-10). The game ends when the maximum number of steps is reached (100 in our experiments) or when the agent captures all of the good goals. Each step taken by the agent incurs a small penalty (-0.1). Each episode of training generates a new board with a randomly placed good and bad objects.

We use off-the-shelf Deep Q-learning (DQN) [12] as our reinforcement learning algorithm with decaying exploration (parameterized through ϵ-greedy action selection), decaying learning rate for the Adam optimizer [9] and a batch size of 256. We also normalize rewards during learning to be in $[-1, 1]$. We did not find that using the difference of frames and representing goal positions as relative to the agent (all positions fed to our model are absolute) were necessary or helpful for learning.

We employ the testing procedure used in [3, 12, 7]: every 10 training episodes, we pause training and test our model on 10 new random boards for 100 steps (or until the agent finds all the good goals) and record the average test performance over these 10 random boards. We repeat this procedure for at least 10 independent trials and report the average testing performance over these independent trials.

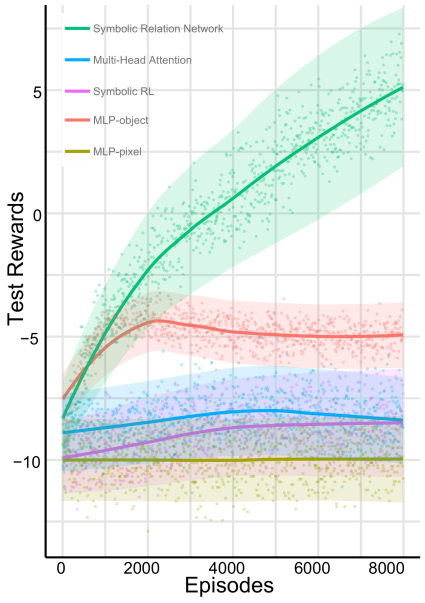

Figure 3: Testing performance of different models. Shaded areas show [5-95] confidence interval.

We test our SRN model against a number of baseline models, always ensuring that the baselines have approximately the same number of parameters. Our baselines consists of: 1) a simple Multi-layer Perceptron comprising of three linear layers with ReLU activation learning from the object representation; 2) a relation network based on multi-head dot-product attention [20]; 3) the symbolic RL model of [7]; 4) an MLP which learns directly from pixels instead of from our object based representation consisting of 3 convolutional layers (with ReLU activations and batch normalization in between them) followed by a final linear layer.

[3] We plan to share the code for our environment and training procedure, saved trained models, and hyperparameter configurations.

5 Results

5.1 Interpretability

To evaluate the interpretability of the learned relations, we trained a Symbolic Relation Network with one unary unit and two binary units with DQN on a board with only 1 goal (good and bad) and, after convergence, evaluated the output of each relational unit. We found that the unary unit learns to recognize good and bad objects. Specifically, a unique value between 0 and 1 is learned consistently for each type of goal. We suspect that if goals provided more continuous levels of reward (as opposed to just two levels, good and bad), the unary relationship would learn a more continuous representation of the goal type, but have yet to perform this experiment.

Figure 2 shows the evaluation for each binary unit. To create these plots, we placed the agent in the center of the board and evaluated the relationship between the agent and an object placed at each of the colored grid locations. Each such goal location is colored on a blue-green spectrum to indicate the value of the relationship (the output of the binary unit's softmax) when applied to the agent and that goal. Blue represents low values; green represents high values. We evaluated the relation using both good and bad objects and found that the value of the relationship learned was invariant to the type of object. We hypothesize that, since the unary relation learned the type, it was presumably unnecessary for the binary relationship to learn it as well. In the Figure 2, (a) shows the agent has learned a notion of degree of goal 'aboveness' of a goal relative to itself while (b) shows the agent has learned a degree of goal 'rightness' relative to itself. Additionally, we found that the same complementary relations can be seen even when the agent is not placed in the center. With these three independent relations, `type`(X), `above`(agent, X), and `right`(agent, X), for objects X, the agent is able to learn a policy that is location relative and hence robust to the random placement of the objects and agent each game.

5.2 Learning Performance

As can be seen in Figure 3, our Symbolic Relation Network (with 8 binary units and 1 unary unit) obtains the highest reward (statistically significant over a [5-95] confidence interval) and outperforms all of the baseline models of the same size (same number of parameters), namely a multi-layer perceptron, a multi-head dot-product attention (MHA) [20] and the symbolic RL model of [7]. In some exploratory experiments, we observe that MHA models with 100 times the number of parameters as our SRN can start competing in performance with our SRN. We also compare our SRN against a larger (with 100 times the number of parameters) Multi-layer Perceptron that learns directly from pixels, and find very little learning (in agreement with [7]).

Model	Reward	Recall	Precision	F1
SRN	**7.27**	**0.50**	0.71	**0.59**
MHA	-5.07	0.31	**0.75**	0.44
Symbolic RL	-5.51	0.32	0.69	0.44
MLP-objects	-3.04	0.35	0.52	0.41
MLP-pixel	-7.35	0.17	0.29	0.21

Table 1: Performance of the Symbolic Relation Network compared to other baselines.

We also compute three other metrics for performance: the *recall* (proportion of good goals found out of the total number of good goals), the *precision* (proportion of good goals found out of the total number of goals found) and their harmonic mean (F1 score), $F1 = \left(\frac{recall^{-1}+precision^{-1}}{2}\right)^{-1}$ which is a standard combined measure. As we can see in Table 1, our model outperforms other baseline model in terms of reward, recall, and F1 score and is close to MHA in precision. The results show that our Symbolic Relation Network not only strongly outperforms other models of comparable size in both reward and F1 measures, but is also more interpretable, learning three complementary relations useful for solving the task.

6 Conclusion

In this paper we introduced a neural architecture for relation learning with multiple unary and binary relational units. The outputs of the units are concatenated into a relational state vector which is supplied directly as input to a Q-function. Our experiments on a simple but challenging stochastic, goal-seeking environment show higher performance over a multi-headed attention model, a standard MLP, a pixel MLP and a symbolic RL model. Additionally, the learned relations are interpretable and complementary.

References

[1] Masataro Asai and Alex Fukunaga. Classical planning in deep latent space: Bridging the subsymbolic-symbolic boundary. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, Louisiana, USA, February 2-7, 2018*, 2018.

[2] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Caglar Gulcehre, Francis Song, Andrew Ballard, Justin Gilmer, George Dahl, Ashish Vaswani, Kelsey Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matt Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks, 2018.

[3] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

[4] Andres Campero, Aldo Pareja, Tim Klinger, Josh Tenenbaum, and Sebastian Riedel. Logical rule induction and theory learning using neural theorem proving, 2018.

[5] Sašo Džeroski, Luc De Raedt, and Kurt Driessens. Relational reinforcement learning. *Mach. Learn.*, 43(1-2):7–52, April 2001.

[6] Richard Evans and Edward Grefenstette. Learning explanatory rules from noisy data. *CoRR*, abs/1711.04574, 2017.

[7] Marta Garnelo, Kai Arulkumaran, and Murray Shanahan. Towards deep symbolic reinforcement learning, 2016.

[8] Çaglar Gülçehre and Yoshua Bengio. Knowledge matters: Importance of prior information for optimization. *CoRR*, abs/1301.4083, 2013.

[9] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[10] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *Advances in neural information processing systems*, pages 3675–3683, 2016.

[11] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40, 2017.

[12] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[13] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[14] Rasmus Berg Palm, Ulrich Paquet, and Ole Winther. Recurrent relational networks, 2017.

[15] Tim Rocktäschel and Sebastian Riedel. End-to-end differentiable proving, 2017.

[16] Adam Santoro, David Raposo, David G. T. Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning, 2017.

[17] Luciano Serafini and Artur d'Avila Garcez. Logic tensor networks: Deep learning and logical reasoning from data and knowledge, 2016.

[18] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, January 2016.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[20] Vinicius Zambaldi, David Raposo, Adam Santoro, Victor Bapst, Yujia Li, Igor Babuschkin, Karl Tuyls, David Reichert, Timothy Lillicrap, Edward Lockhart, Murray Shanahan, Victoria Langston, Razvan Pascanu, Matthew Botvinick, Oriol Vinyals, and Peter Battaglia. Relational deep reinforcement learning, 2018.