Relational Deep Reinforcement Learning*

Zambaldi, Raposo, Santoro et al.

2018

1 What

- Box-World, a new environment targeting relational reasoning.
- Demonstration of the fact that agents who are able to produce relational representations using non-local computation (based on attention) show interesting generalisation behaviours.
- Application of the method for SCII mini-games and beating a grandmaster on some of them.

2 Why

As usual, DL and RL are on the rise, but we struggle to solve sample inefficiency, interpretability and generalisation issues. However, there is cool old school Relational RL (RRL) which allows us to combine generalisation capabilities of relation logic with RL that overfits. Let's combine all the latest DL/RL/DRL advances with this RRL research.

Moving from propositional to relational logic opens new horizons for generalisation and incorporating background knowledge into your ULTRADEEP nets. As far as I understand, the difference between propositional and relational logic is an ability to work with predicates: statements that might be right or wrong given the incoming variables. 'Juergen Schmidhuber is one of the LSTM's creators' in propositional logic might be turned in 'X is LSTM's creator' or even to creator(X,Y) in relational logic.

3 How

As the authors say, they were guided by two following principles: *non-local* computations using a shared function and *iterative computation*. They assume that an agent which can reason about the objects in terms of their relations without

*Notes by Vitaly Kurin `https://yobibyte.github.io/`

being tied to their spatial proximity is better suited for learning important stuff about these relations. They also claim that iterative computation might capture higher-order interactions between entities.

The authors make use of self-attention [Vaswani et al., 2017] to do these non-local computations with a shared function. They use 'query, key, value' terminology I encountered in [Vaswani et al., 2017]. As far as I get it [1], we get the output of the CNN block, slice it depth-wise and project to the vector [q,k,v] and do all the computations with this vector.

The attention matrix has the following structure:

$$A = softmax(\frac{QK^T}{\sqrt{d}})V, \tag{1}$$

where everything in front of V is attention weights. There is a slight confusion in the notation here, if I'm not mistaken, V here stands for attention weights only, however, we also have V for the value function, which is the final element of the network output. Intuitively, I parse this formula as weighting keys K with query Q which takes more of similar values and less of non-similar ones. And then we just weight the values we have based on this similarity. I'll copypaste the architecture figure from the paper below:

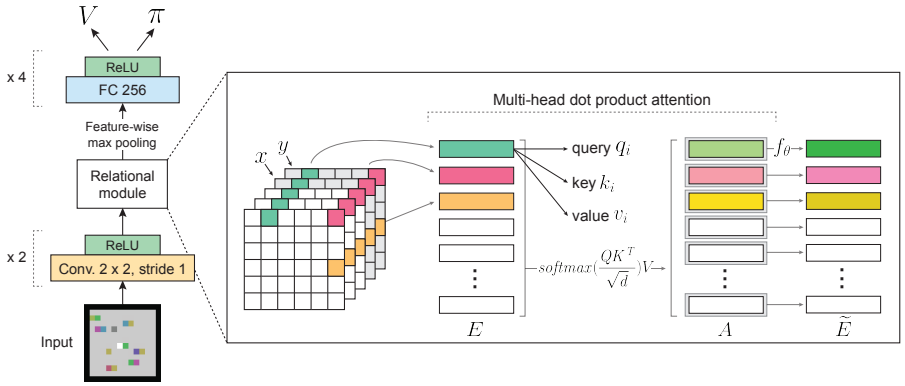

Figure 1: Model architecture

Nice. The attention mechanism and the network architecture is more or less clear. But how does all of this relates to the promised relational logic? Here we assume that all the entities have their position in space. We add x, y coordinates to the CNN output feature maps so that every pixel has its location on the map (so, we just have two additional channels for coordinates). The model uses multiple heads for attention as in [Vaswani et al., 2017]. As far as I get it, the hope is that each head learns its own semantics for the relation.

The authors use IMPALA [Espeholt et al., 2018] agent to learn the policy and the value functions. I will not talk about agents architectures here, you

[1] `https://www.reddit.com/r/MachineLearning/comments/6kc7py/d_where_does_the_query_keys_and_values_come_from/`

can always find them in the paper. Interestingly, the Starcraft agent does not use any of the fancy memory models but uses just an LSTM to manage the temporal nature of the task.

4 Evaluation

4.1 Box World

The environment is a 12×12 grid with pixels of different colour denoting keys, boxes and the agent. A key is a pixel not adjacent to any of the coloured pixels. A box is two adjacent pixels denoting a lock and content of the box. Any lock has a colour and you need a key with corresponding colour to open it. The content of the box is accessible only when you open it. The agent can move in four directions. The agent needs to walk over the tile to interact with the object on it. The current key the agent has is present in the observation space.

There is a unique sequence of boxes you need to open to complete the level (reach the gem). If you open a wrong box (a distractor), you won't be able to complete the level since you can use a key only once.

The relational agent was able to solve most of the levels, whereas the baseline (just a neural net) was able to solve about 75% of the tasks.

The authors decide to visualise what the agent attends to. They show that some attention head attends to the locks of the same colour the agent has the key of. Another head attends to the agent pixels, thus relating it to the other objects on the scene. The agent also attends to the gem and the other way around.

So, the information above tells us that the network learns something like $unlocks(key, lock)$ The authors test another two scenarios: checking whether an agent can solve problems with longer sequences and checking the key-lock combinations the agent has never seen. The relational agent generalises to both of the tasks, whereas the baseline becomes completely helpless.

4.2 Starcraft II

All the experiments in questions are the experiments with so-called mini-games from SCIILE [Vinyals et al., 2017]. An important moment here is that the authors report the mean performance, not the best one. It would be also interesting to see the standard error of the mean for that.

There is also the generalisation check for the SCII task when the authors add more units to collect minerals while training only with two units. Ideally, the model should understand that it can deploy all the units it has access to. Some networks of medium size were able to show generalisation, however, the results are highly variable. Moreover, the generalisation capabilities are diminishing with the larger model. More work has to be done.

5 Comments

- I like how the authors put their contributions without overselling and saying that their agents generalise **better** than all state-of-the-art super-duper approaches. They are very accurate with words and this is amazing.
- Has Starcraft been solved? No. Grand-masters are very good at balancing macro and micro. The current work shows only mini-games. That's like unit tests for your code. Let's hope we will see integrations tests soon. Again, that's amazing that the authors do not make bold claims about solving SC!
- More and more DeepMind papers make use of IMPALA agent which makes people who want to replicate the results and build on top sad. A really cool thing about working in such a big company like DeepMind is not even about the compute, it's about the whole infrastructure which make use of this compute effective.
- The paper is very detailed regarding the architecture, hyperparameters and similar peculiarities. That's amazing. However, I'd like to see more of the motivation, intuition and reasoning which led to these achievements.
- Apart from the details I mentioned above, the paper is very generous for references and this is also very nice for those who would like to go deeper.
- I like this paper for another reason as well. It is another clear example that we need inductive biases to achieve more. It's very naive to think that throwing all your data into one big model will produce something reasonable after yobiyears of processor time.
- I really like the generalisation experiments in the Box World. Poking your model and observing its behaviour feeds your curiosity and provides you with a ton of new insights.
- It's amazing that the authors create a simple environment to interpret the model behaviour and show that the model works as intended. It might have been a great debugging tool as well, I suppose.
- The paper has a really useful and rich 'future work' paragraphs in the 'Conclusion' section. It makes it really by head and shoulders above what we usually see nowadays.
- I might have missed something, but I haven't seen anything on iterative attending apart from couple of mentions in the paper.
- I might be again missing something, but it looks vague to how we expect the relational logic to be imposed here. Do we expect it to evolve and do the empirical checks that it does? What are the components which make our attention heads learn the relation properties?

References

[Espeholt et al., 2018] Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., et al. (2018). Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. *arXiv preprint arXiv:1802.01561*.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010.

[Vinyals et al., 2017] Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhnevets, A. S., Yeo, M., Makhzani, A., Küttler, H., Agapiou, J., Schrittwieser, J., et al. (2017). Starcraft ii: a new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782*.