Relational Forward Models For Multi-Agent Learning*

Tacchetti, Song, Mediano et al.

2018

1 What

Graph networks [Battaglia et al., 2018] for Multi-Agent RL (MARL). Relational Forward Model (RFM) is a new type of models which predict the forward dynamics of a multi-agent system and produce intermediate analysable representations. Plugging RFM into MARL agents improves the speed of learning.

2 Why

As the authors claim, fostering coordination between agents in MARL is one of the outstanding challenges. Another essential problem is in how to analyse the behaviour of a multi-agent system. RFMs address both of the challenges.

3 How

RFMs are based on Graph Networks [Battaglia et al., 2018] (GN). RFM takes a semantic graph as an input and outputs either an action prediction for agents or the cumulative reward till the end of the episode.

No notes for GN background here, please have a look at my notes[1].

GN has an encoder-decoder architecture with a GRU. The authors introduce recurrence to do relational reasoning both on the input graph and its representation in time as well. It's as if the network how the relations developed over time, not only how they look now.

Encoding GN works as an input for a GRU block. We keep the output of a GRU for the next time step and process it through a GN decoder to obtain an output.

*Notes by Vitaly Kurin `https://yobibyte.github.io/`. Thanks to most of the authors for the paper sources (I copypaste long formulas and figures sometimes). And thanks to you all for feedback. Stay tuned!

[1] Also have a look at my notes here: `https://yobibyte.github.io/files/paper_notes/gn.pdf`

In all the environments, the authors trained an A2C agent till convergence and then collected the trajectories and all the semantics (to form the graph).

How would one represent semantics as a graph? All the agents and static entities are nodes with their features (e.g. position). Invalid attributes are padded with zeroes (e.g. last action of an apple). All agents are connected to each other and all the static entities. Input edges have no attributes (there is only a sender-receiver relation though).

The authors held out some set of trajectories for analysis.

4 Evaluation

The authors benchmark RFM on the following three environments: Cooperative Navigation [Lowe et al., 2017], Coin Game [Raileanu et al., 2018] and Stag Hunt [Peysakhovich and Lerer, 2017].

4.1 Action Prediction

All the models perform similarly well on *Cooperative Navigation*, which is not surprising given its simplicity. In *Coin Game* and *Stag Hunt*, RFM beats all, though NRI is close (RFM can keep the predictions correct for an additional step). Surprisingly, a feedforward graph network performs also quite good. This shows that GNs help a lot in learning.

4.2 Stag Hunt Relational Analysis: Actions

The authors find that the Euclidean norm of a message vector ($||e_k||$) shows how the sender influences the receiver, i.e. it's more likely that the sender will interact with the receiver at the next step.

4.3 Stag Hunt Relational Analysis: Return

Removing/adding an edge between agents influences the return estimation the model gives us. In particular, when we have information about the second agent, the return for the first one is higher. This means that the second agent is helping the first.

It is also intriguing to see that the estimated utility of a teammate decreases when the stag is caught. However, I'm not sure if that's very surprising. The total return estimate should decrease after the stag is captured. Before: 10 (on my own), 20 (with a teammate). After: 2 (on my own), 3 (with a teammate). The numbers above are entirely random. However, I just wanted to say that delta between small numbers is lower than the delta between larger numbers (should we measure the percentage here?).

4.4 RFM-augmented agents

Finally, it's time to check how RFM agents perform in a MARL setup. RFM's output goes to a policy. So, the RFM predicts actions of the teammates, the policy acts. No communication between the agents, each of them had their own RFM/policy net. The agents in question were trained along with the agents pre-trained in a supervised way. RFM conditioned policy performs better than A2C.

5 Comments

- I'm amazed how the GN abstraction helps us to describe quite a complicated model in such a simple way. All the GN stuff is abstracted, and we have three blocks.
- I don't quite get why the authors compared RFMs with NRI [Kipf et al., 2018]/FeedForward GNs only on action prediction performance, but not on the real environment performance later on.
- The authors say that RFM is identical to NRI, except for the initial graph structure inference step. That looks interesting, need to read the NRI paper.
- RTFM would be much a cooler name though. Relational Teammate Forward Model maybe?

References

[Battaglia et al., 2018] Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gulcehre, C., Song, F., Ballard, A., Gilmer, J., Dahl, G., Vaswani, A., Allen, K., Nash, C., Langston, V., Dyer, C., Heess, N., Wierstra, D., Kohli, P., Botvinick, M., Vinyals, O., Li, Y., and Pascanu, R. (2018). Relational inductive biases, deep learning, and graph networks.

[Kipf et al., 2018] Kipf, T., Fetaya, E., Wang, K.-C., Welling, M., and Zemel, R. (2018). Neural relational inference for interacting systems. In Dy, J. and Krause, A., editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2688–2697, Stockholmsmässan, Stockholm Sweden. PMLR.

[Lowe et al., 2017] Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., and Mordatch, I. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments.

[Peysakhovich and Lerer, 2017] Peysakhovich, A. and Lerer, A. (2017). Consequentialist conditional cooperation in social dilemmas with imperfect information.

[Raileanu et al., 2018] Raileanu, R., Denton, E., Szlam, A., and Fergus, R. (2018). Modeling others using oneself in multi-agent reinforcement learning.