RELATIONAL FORWARD MODELS FOR MULTI-AGENT LEARNING

요약

- 다중 에이전트 환경에서의 강화학습을 더 잘할 수 있는 프레임을 제시한 논문이다.
- 미래의 상대방의 행동(혹은 상대방의 보상정도)를 예측하는 오라클을 만들수만 있다면, 이를 이용하여 재귀 강화을 학습을 하자는 아이디어
- 오라클을 어떻게 만들 것인가?
  - 일단 학습용 ground-truth는 있다고 본다.
  - 환경을 그래프로 보고, 그래프 기반 RNN 모델을 통해 지도학습을 한다.

도입

다중 에이전트 강화 학습 연구

난제 1. 어떻게 협력하게 할 것인가?

- 난제는 에이전트끼리의 협력적 행동 양태를 붇돋아(foster) 주냐는 것이다.
- hand-craft를 통해서 각 에이전트의 행위와 상호간의 역할 관계를 정해주는 approach가 있다.
- 학습 기반이라면 중앙화된 controller를 두는 approach도 있다.
  - 한계는 대규모 에이전트 기반으로의 확장성이 문제가 있다.
- 그러므로 컨트롤 타워가 없어도 **각자 스스로(on their own) 협력성을 증대시키는** 매커니즘이 필요하다고 하겠다.

난제 2. 어떻게 협력하고 있는지 측정할 것인가?

- 기존 연구는 개별 에이전트의 기능성 측정에만 초점이 있지, 협력의 측정에는 미흡하다.

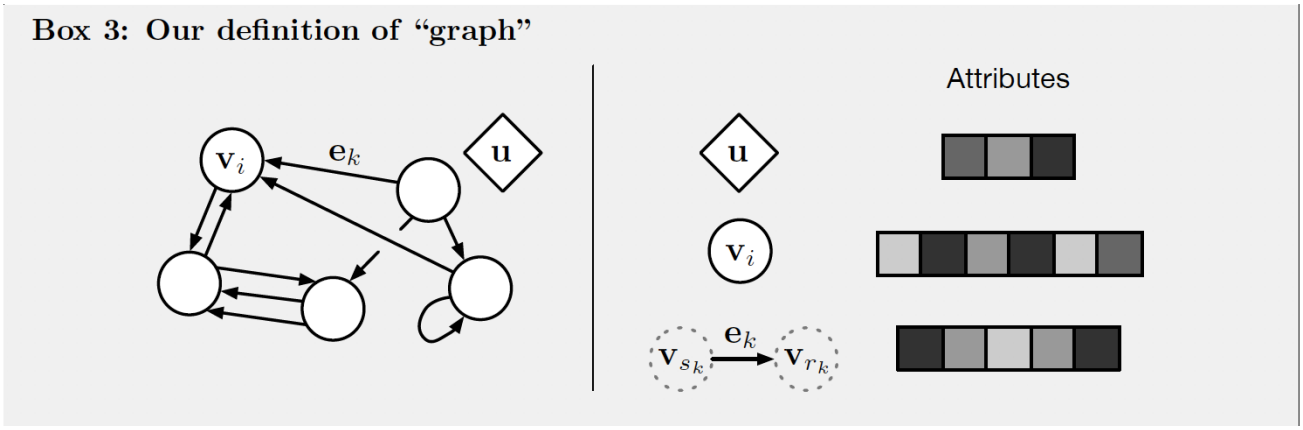
우리의 제안 : Relational Forward Model (RFM)

- 첫번째로 그래프 네트워크를 수단으로 해서, 다중 에이전트 시스템의 다이네멕스를 예측하는 모델을 학습하게 할 것이다.
  - 이는 일종의 관계 추론(relation reasoning)이다.
- 이 예측 모델은 설명 가능하다. (너무 거창함..)
  - 무엇이 각 에이전트의 행동을 드라이브하며,
  - 각 에이전트가 상호간에 영향을 주는 것을 추적하며
  - 어떤 환경적 요소가 사회적 상호작용을 강화시키는지 밝힌다.
- 두번째로 이 예측 모델을 기존 강화학습 프레임에 장착(augment)하여 상대방의 미래 행동을 예측하여 이를 관찰과 함께 정보로 활용하게 한다.
  - 이는 이 예측 정보를 가지지 못하는 baseline보다 월등하게 성능 향상을 도모한다.

DeepMind의 Graph Network (GN) 간략 소개

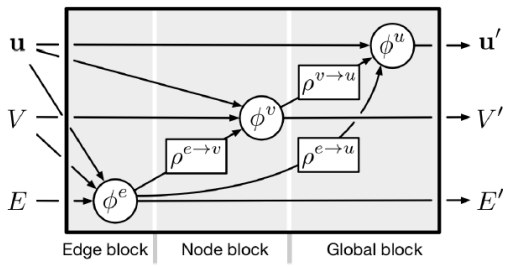
Relational inductive biases, deep learning, and graph networks, Battaglia, et al. 2018

구성 요소와 attribute

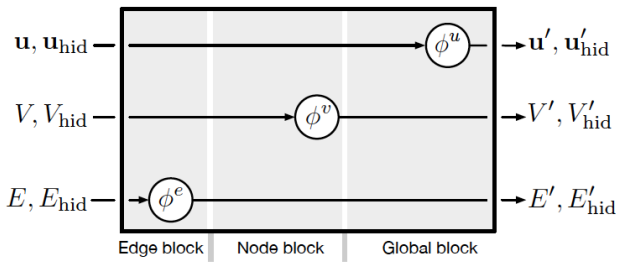


구성 요소끼리의 상호 작용 통한 업데이트

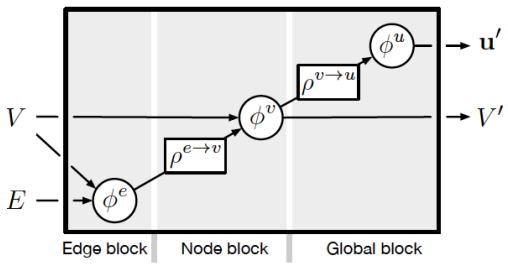
- 아래 그림에서 (a) Full GN block이 deepmind 방식



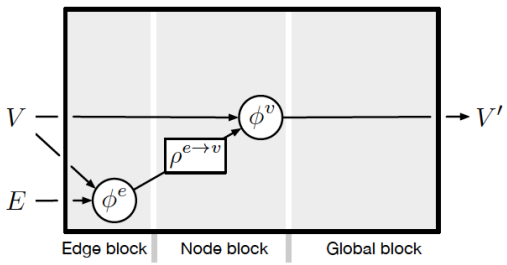
(a) Full GN block



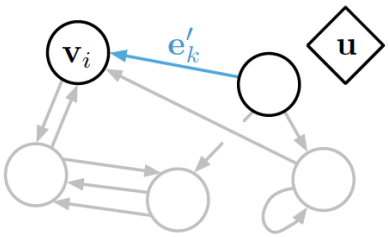
(b) Independent recurrent block



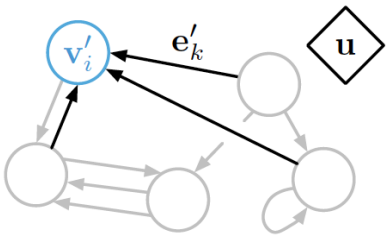
(c) Message-passing neural network



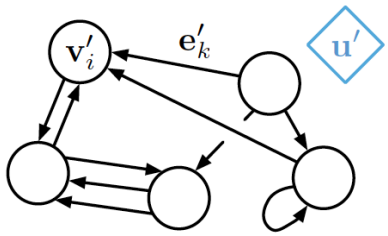
(d) Non-local neural network



(a) Edge update



(b) Node update



(c) Global update

Figure 3: Updates in a GN block. Blue indicates the element that is being updated, and black indicates other elements which are involved in the update (note that the pre-update value of the blue element is also used in the update). See Equation 1 for details on the notation.

A GN block contains three "update" functions, ϕ, and three "aggregation" functions, ρ,

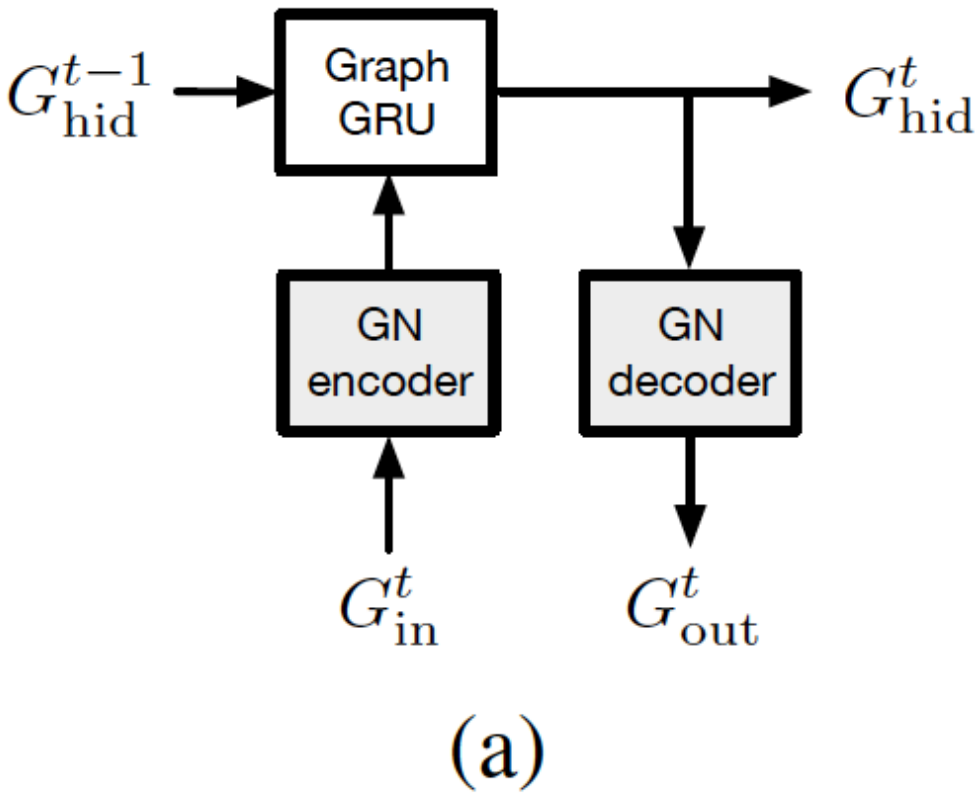
$$\begin{aligned} \mathbf{e}'_k &= \phi^e\left(\mathbf{e}_k, \mathbf{v}_{r_k}, \mathbf{v}_{s_k}, \mathbf{u}\right) & \bar{\mathbf{e}}'_i &= \rho^{e\to v}\left(E'_i\right) \\ \mathbf{v}'_i &= \phi^v\left(\bar{\mathbf{e}}'_i, \mathbf{v}_i, \mathbf{u}\right) & \bar{\mathbf{e}}' &= \rho^{e\to u}\left(E'\right) \\ \mathbf{u}' &= \phi^u\left(\bar{\mathbf{e}}', \bar{\mathbf{v}}', \mathbf{u}\right) & \bar{\mathbf{v}}' &= \rho^{v\to u}\left(V'\right) \end{aligned} \tag{1}$$

where $E'_i = \{(\mathbf{e}'_k, r_k, s_k)\}_{r_k=i,\, k=1:N^e}$, $V' = \{\mathbf{v}'_i\}_{i=1:N^v}$, and $E' = \bigcup_i E'_i = \{(\mathbf{e}'_k, r_k, s_k)\}_{k=1:N^e}$.

The ϕ^e is mapped across all edges to compute per-edge updates, the ϕ^v is mapped across all nodes to compute per-node updates, and the ϕ^u is applied once as the global update. The ρ functions each take a set as input, and reduce it to a single element which represents the aggregated information. Crucially, the ρ functions must be invariant to permutations of their inputs, and should

본 논문에서의 Graph Network 의 응용

- 복수 개의 GN block을 엮여서 사용(composition)
- encoder-decoder with GRU 느낌의 구조
- encoder/decoder 는 vertex ft, edge ft, graph ft을 3계층 MLP (outsize = 64)
  - 입력이 그래프 구조, 출력도 그래프 구조
- Graph GRU는 입력이 그래프 2개, 출력이 그래프 1개
- decoder 뒷단에는 최종 출력이 도출
  - 각 에이전트별 action distribution 이나
  - 각 에이전트별 reward 예상



(a)

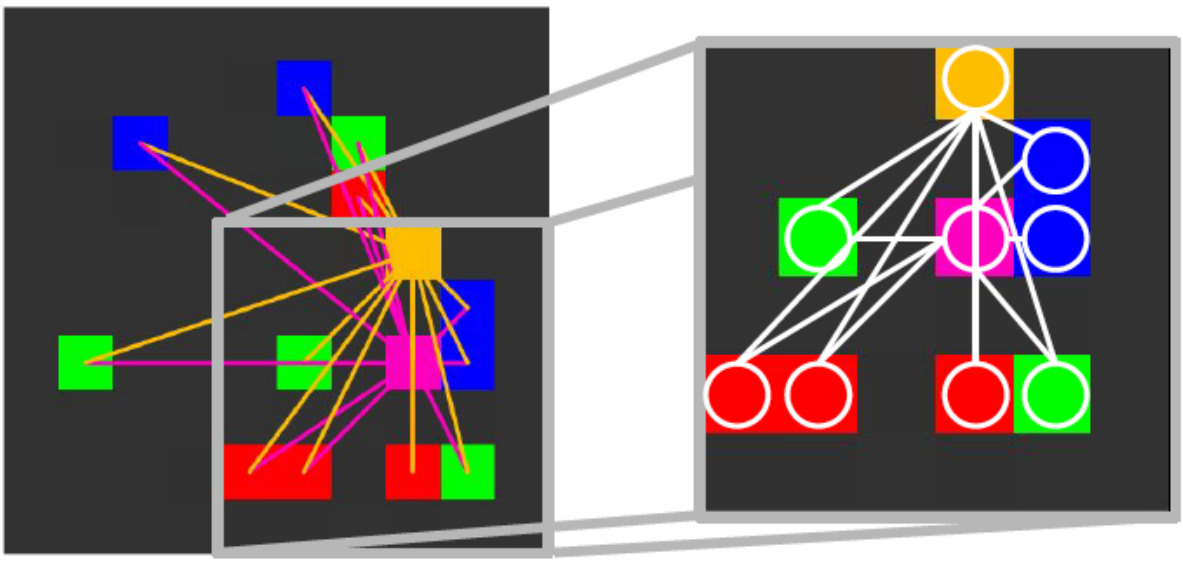
MARL 환경 예시

협력 항해 문제

- 2 agent
- 6 x 6 grid
- 2개의 보상 tile
- 다중 에이전트가 동시에 두 개의 tile에 도착시 +1
- 20 step max

동전 먹기 문제

- 2 agent
- 8 x 8 grid
- 10 step max
- 12개 동전(빨/초/파 색상별로 4개씩)
- 두 가지 색상은 reward제공, 나머지 색상은 punishment
- 상대방이 꺼리는 색상을 초반에 간파하면 보상이 높아지니, 타인 관찰 (혹은 협력)이 중요



(b)

사슴 사냥 문제

- 2 or 4 agent
- 3마리 사슴(stag)과 12개 사과
- 32 step max
- 협력해야만 사슴을 잡을 수 있고, 이때 보상이 사과따는 보상에 비해 매우 크다.
- 가끔 안개가 껴서 사슴과 사과가 안보일 때가 확률적으로 있다.
  - 이러면 목표가 사라지만 협력의 필요성이 약해지게 된다.

pre-trained agent를 통한 ground-truth 데이터 확보

- By multi-agent versioned importance-weighted actor-learning
  - Human-level performance in first-person multiplayer games with population-based deep reinforcement learning, deepmind 2018
- 각 환경별 500,000 에피소드

그래프 데이터로의 표현

- 각 time step별로 환경의 상태와 각 에이전트가 취한 행동, 그때의 보상을 취합하고 이를 그래프 요소요소에 녹여낸다.
- 각 에이젠트들과 entity(ex. 사과나 사슴)들이 노드가 된다.
  - 노드의 속성은 위치, 종류(entity or agent), entity 상태(available or collected), 마지막 행동
- 엣지는 entity에서 모든 agent로 연결되는 것과 에이전트끼리 연결되는 두가지가 있다.
  - 엣지의 속성은 input graph에서는 sender/receiver 표시만 있었다가
  - 학습이 되면서 output graph에서는 없던 엣지 속성이 형성이 된다.(?)
  - 이렇게 학습된 엣지 속성이 상호 작용 설명의 근거가 된다.

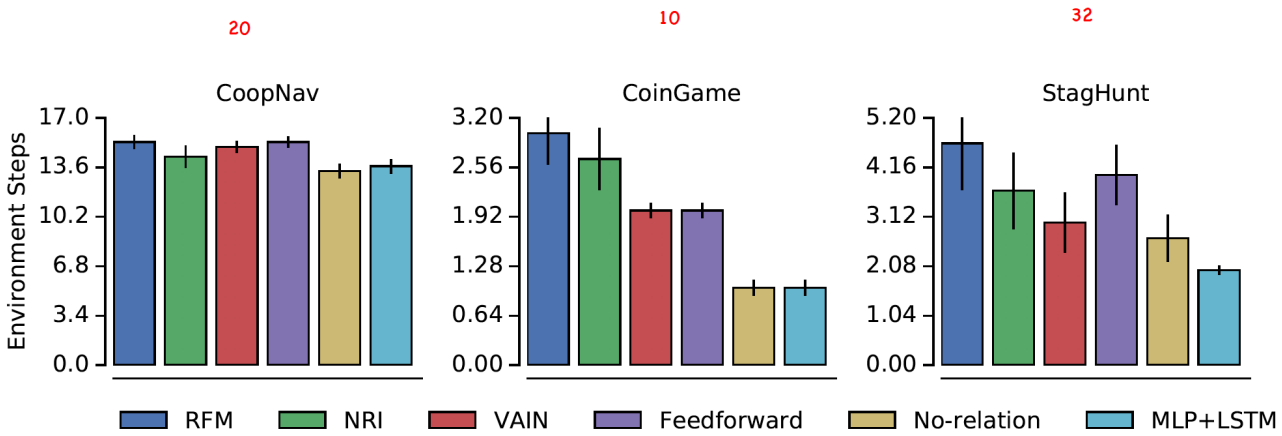
상대방 행동 예측 모델 학습 결과

ablated baseline

- FeedForward Model
  - GraphGRU가 없는 모형, 일종의 그냥 autoencoder 컨셉
- No-relation model
  - 모델은 Full RFM과 같지만 edge가 없는 그래프 이용
- MLP + LSTM
  - 모델은 Full RFM과 같지만 그래프가 아닌 vector 형태의 입력 데이터 이용

학습 결과 비교

- 협력 항해의 경우 75% 예측 정확도, 동전 문제는 30%, 사슴 사냥은 16% 예측 정확도
- 시간이 지나면서 협력의 중요성이 부각되는 문제인 동전 문제와 사슴 사냥 문제에서 특히 우리 것이 다른 것보다 뛰어났다.
- 우리의 ablated baseline보다 좋았다. 즉 그래프를 통한 관계 형성이 주요했다.

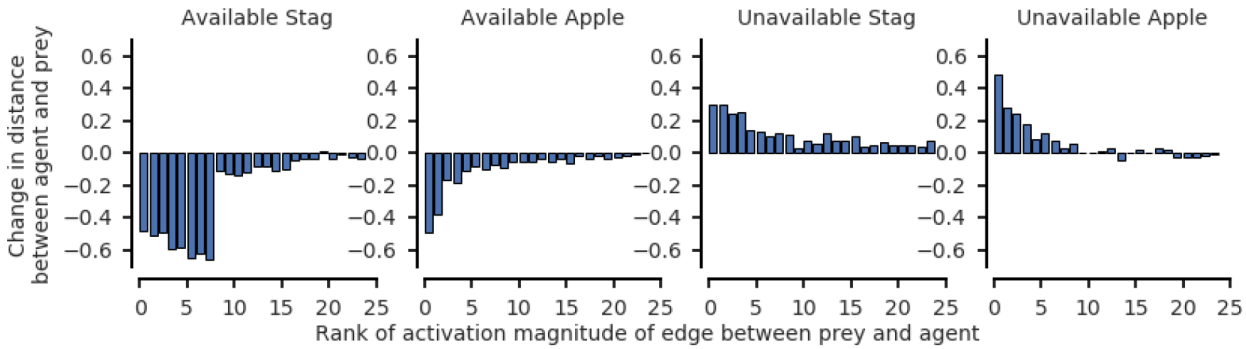


사슴 사냥 사례를 통해 본 상대방 행동 예측모델의 관계 분석

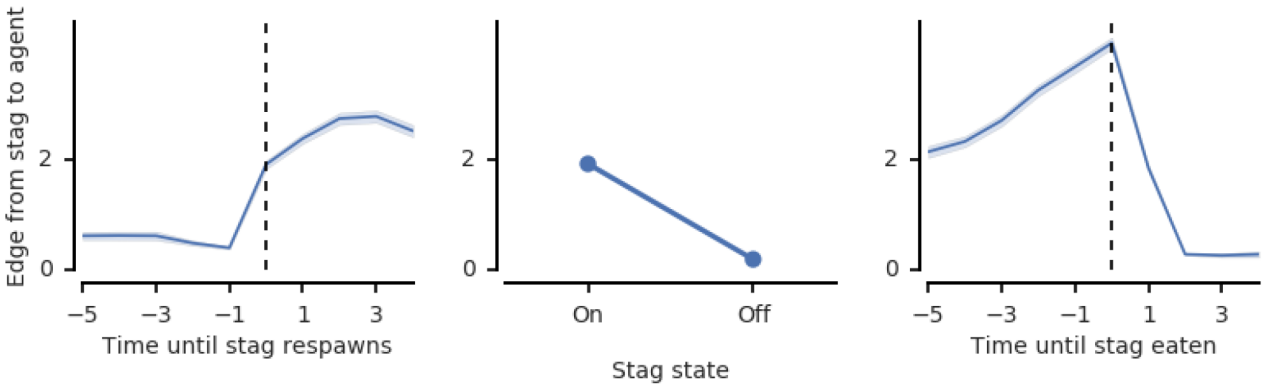
엣지의 메세지 강도로 통해 본 관계 형성

- edge norm from sender entity to a receving agent 정보는 해당 에이전트가 다음에 어떤 행동을 할지에 대한 가늠자(predictive)가 된다.
- 아래 그림에서 에이전트의 행동으로 인해 사냥 가능한 사슴 객체로의 거리가 많이 가까워질수록 연결 강도가 강하다.
  - 사냥 불가능한 사슴 객체와의 연결 강도는 반대이다.
  - 상대적으로 사과에 대한 강도는 약하다.

(a) Edge activation magnitude is predictive of future behavior.

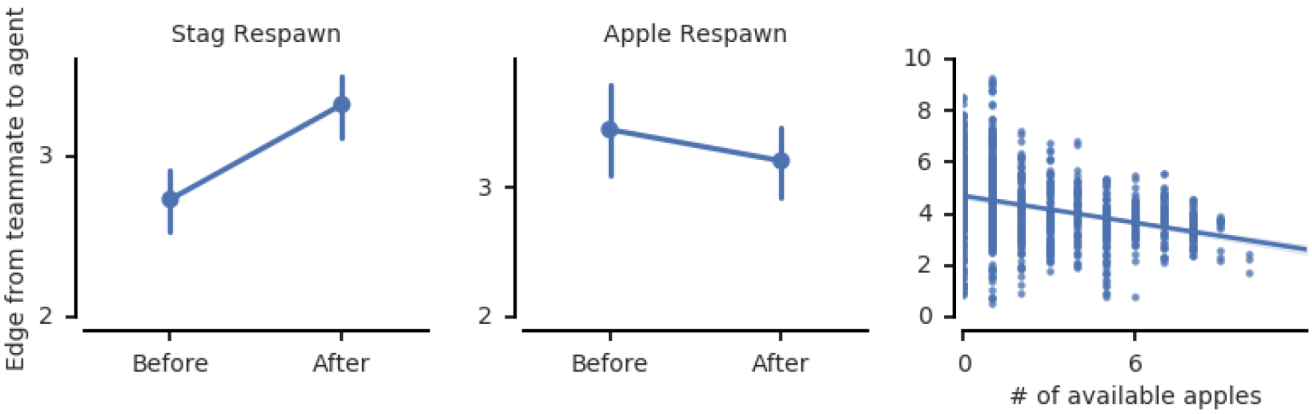


- 아래 그림에서는 특정 사슴과 특정 에이전트끼리의 연결 강도를 시간 변화에 따라 표현한 것이다.
- 상황 변화가 상호 작용의 정도를 변화시킨다.
  - 안개에 갇혀 있을 때는 연결 강도가 약하다가 안개가 걷히면 강해짐
  - 사냥을 당하면 약해짐



- 아래 그림은 에이전트끼리의 상호 작용을 표시한다.
- 협력해서 사냥할 사슴이 나타나면 연결이 강해진다. 상대적으로 사과는 개인 플레이므로 연결 강도가 낮아진다.
- 먹을 사과 총 개수가 늘어나면 날수록 협력의 필요성이 낮아지니 연결 강도가 낮아진다.

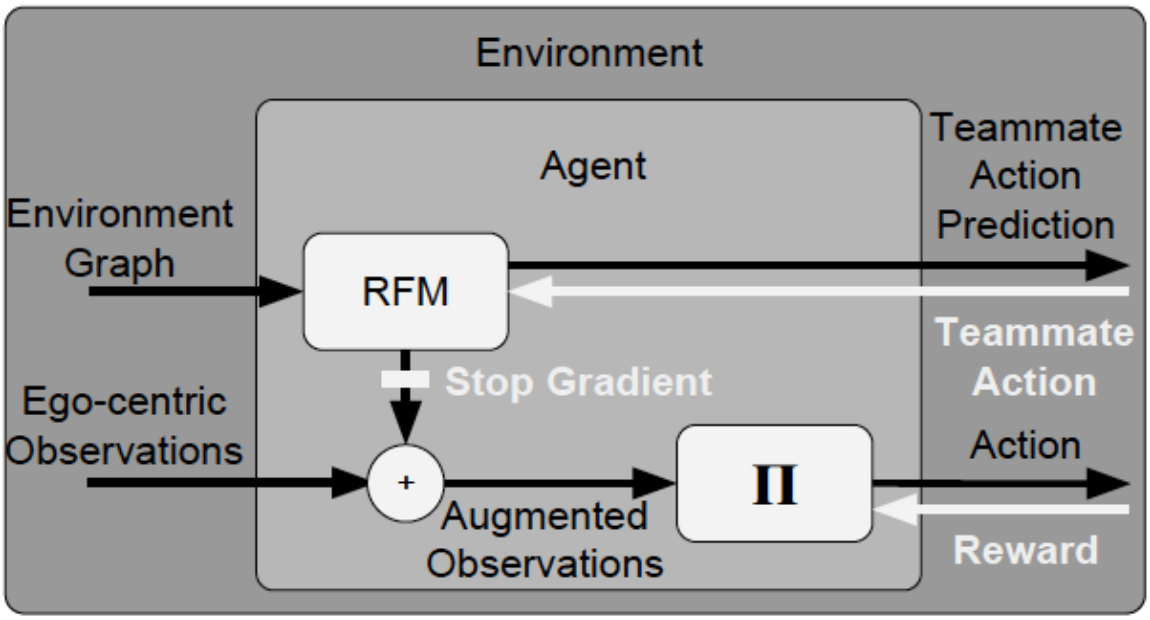
(c) Edge activation magnitude discovers situations that alter agents' social influence.



RFM-Augmented Agents

- 타인 행동 예측 모델을 이용해서 MARL 에이전트의 학습 가속화를 시킬 수 있지 않을까?
- MARL agent에는 타인 행동 예측 모델을 통해서 얻은 타인의 다음 위치 정보를 이미지화해서 추가로(augment) 강화학습에 사용할 수 있게 한다.

- 정확한 절차
  1. A2C로 학습한 agent가 있다.
  2. 이를 이용해서 offline 으로 많은 에피소드를 얻고, 이를 통해 타인 행동 예측 모델을 학습한다.
  3. 4-player 사냥 게임이라면, 1명만 learning A2C agent가 되고, 나머지는 freezed AC2 agent로 구성하고 플레이 시킨다.
  4. learning AC2 agent는 타인 행동 예측 모델을 이용해서 더 가속화해서 강화학습을 하게 된다.



(c)

결과 비교

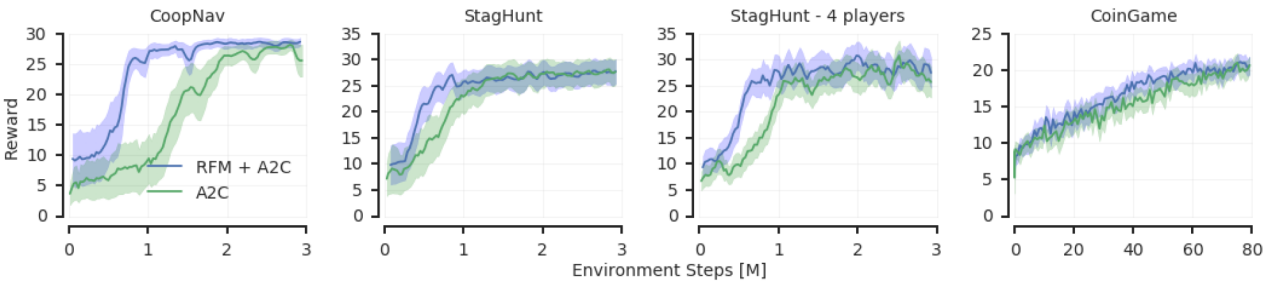


Figure 5: Training curves for A2C agents with and without on-board RFM modules. Allowing agents to access the output of a RFM module results in agents that learn to coordinate faster than baseline agents. This also scales to different number of agents. Importantly, the on-board RFM module is trained alongside the policy network, and there is no sharing of parameters or gradients between the agents.

In [ ]: