Floyd-Warshall Reinforcement Learning: Combining advantages from model-free and model-based RL

Vikas Dhiman

Abstract

Reinforcement Learning (RL) algorithms are often discriminated on the basis of whether a state-transition model is learned implicitly (*model-free*) or explicitly (*model-based*). In the implicit case i.e. *model-free* RL, algorithms such as Q-learning and policy gradients learn a joint state-action function that maps state-action pairs to expected future reward in an agent's environment. The reward distribution that governs an agent's environment is thus implicity memorized in the learning of this function. Due to this combined learning of state-transition dynamics and the reward distribution, learned behaviors in *model-free* RL are hard to transfer to environments where the state-transition model remains the same while the reward-distribution changes. In contrast, *model-based* RL explicity learns the state transition function allowing for the decoupling of learned behaviours to environments in which only the reward distribution changes. Oftentimes, this explicit modelling leads to the requirement of an additional planning step to compute model dynamics which can make policy computation in such algorithms expensive. Inspired by both these RL paradigms and the Floyd-Warshall algorithm for path-planning on graphs, we propose Floyd-Warshall Reinforcement Learning (FWRL), a novel algorithm that combines advantages from both *model-based* and *model-free* approaches. Our main contribution is an algorithm that learns a goal-conditioned value function which transitions from model-based behavior to model-free behavior in well explored regions of an agent's state space. We show that FWRL transfers knowledge about the environment when the reward location is dynamic compared to a model-based Q-learning baseline. We show (TODO: fill in the results when exp done) that FWRL meets the ground between model-free and model-based algorithms by being model-free in the more frequently visited regions while being model-based on less visited paths.

Introduction

The Reinforcement learning algorithms are classified on the basis of whether the state-transition model is learned explicitly into *model-based* and *model-free* algorithms. The *model-free* algorithms like Q-learning and policy gradients are easier to learn in cases when model is more complex than the policy. However, model-free algorithms are harder to transfer in cases when the reward function changes while the state-transition model remains the same. One example of such a problem is traveling salesman problem. Although classical traveling salesman problem is a posed as a planning problem where the space has already been explored, however it is not hard to recognize that someone has to create a map of the map through exploration and sometimes it has to be the agent them-self. In such a scenario, the traveling salesman has a dynamic reward with a static map that needs to be explored. The model-free approach fails to generalize to new rewards in these scenarios, because of the conflated representation of environment model and resulting reward. In spite of these limitations, model-free approaches have been applied to multi-goal navigation scenarios with considerable success **?**. However, all these methods depend upon exploring the entire space being a candidate goal space. Not only model-free approaches are sample inefficient, multi-goal navigation problem exaggerates this problem.

The *model-based* algorithms learn the state-transition model explicitly hence making it easier to separate the environment transition from the reward distribution. Also model-based algorithms are known to be sample efficient than model-free algorithms especially in cases when the state-transition model is translation invariant in the environment. However, model-based algorithms require an additional planning step on the model dynamics which make it expensive to compute the policy on the fly. Although model-based algorithms fail to find an optimal policy, because small errors in models can add up and lead to wrong prediction over larger state spaces.

Taking inspiration from the Floyd-Warshall algorithm in graph-based path planning domain, we propose Floyd-Warshall reinforcement learning (FWRL) algorithm that combines the benefits of both model-based and model-free methods. We define the Floyd-Warshall function F for a goal state s_j starting from a state-action pair s_i, a_i as the expected reward through all possible paths p_{ij} between the

starting and destination state.

$$F(s_j|s_i, a_k) = \mathbb{E}_{p_{ij}\sim\pi}\left[\sum_{s,a\in p_{ij}} \gamma^t r(s,a) \middle| p_{ij} = (s_i, a_k, \ldots, s_J)\right]. \quad (1)$$

For an optimal policy

$$F^*(s_i, a_i, s_j) = \max_{p_{ij}} \sum_{k\in p_{ij}} \gamma^t r(s_k, a_k). \quad (2)$$

In terms of Q-function and Value function the Floyd-Warshall function has the following relationships

$$V(s_i) = \mathbb{E}_a[Q(s_i, a)|s_i] \quad (3)$$
$$F(s_i, a_i, s_j) = V(s_j) - Q(s_i, a_i). \quad (4)$$

For the optimal policy the expectation is replaced by the max operator, that is $F^*(s_i, a_i, s_j) = \max_a Q^*(s_j, a) - Q^*(s_i, a_i)$.

Due to this formulation, the optimal Floyd-Warshall function should satisfy the transitive property

$$F^*(s_i, a_i, s_j) = \max_{s_k}\left[F^*(s_i, a_i, s_k) + \max_{a_k} F^*(s_k, a_k, s_j)\right] \quad (5)$$

The advantage of this formulation is that as soon as the goal state s_g, the Q-function and hence the policy can be estimated from $Q(s, a) = F(s, a, s_g)+R(s_g)$. Even if the path between two states has not been ever traversed, the value function can be computed using the transitive property in Eq 5.

In our experiments, we show how our approach is better than both model-free Q-learning and model-based approaches. We also show how our approach can be switch between either approaches depending upon traversal experience in the state space.

Claims

- Using FWRL is better than model-free learning like Q-learning in terms of generalizing to dynamic goals, static maps.
- Using FWRL is better than model-based learning because of less accumulation over long term, hence more accurate value functions.
- FWRL is better than Hindsight Experience Replay (HER) Andrychowicz and Kurach (2016) because it is a parametric representation of hindsight.
- FWRL is better than TDM Pong et al. (2018) because it does not require the distance reward function.

Related work

Goal-conditioned value functions

The idea of goal-conditioned value function is not new but has got attention because of revival of reinforcement learning based on deep neural networks. We build upon the goal-conditioned value functions of Schaul et al. (2015). Schaul et al. (2015) proposed an architecture and a matrix factorization based algorithm for faster learning of UVFA (Universal value function approximators). UVFA focused on fast estimation of goal-conditioned value functions using sparse matrix factorization but not on bridging the gap between model-based and model-free algorithms.

Andrychowicz and Kurach (2016) introduced the idea of Hindsight experience replay (HER) two learn about the model from previous episodes even when the goal location has changed or not been achieved. Our method can be seen as parametric approximation of "Hindsight memory", that can help compress information from previous episodes instead of maintaining the entire history of replay memory.

Pong et al. (2018) propose temporal difference models that estimate goal directed Q function for a specific kind of reward function, in particular the distance from the goal and in contrast with limited temporal horizon.

Combining model-based and model-free methods

Navigation with mapping

(1) CMP from Saurabh Gupta: is metric, might not working in continuous spaces. (2) Semi-parameteric Topological mapping: is not end to end. (3) Neural Map: Is actually not mapping

Model free DRL

does not generalize to multi-goal environments.

Model based DRL

Needs more exploration. Find the paper that shows that Model based DRL can actually compete with Model free DRL as long as it models uncertainty.

Multi-goal navigation based papers

Mirowski 2017, 2018: No one shot map learning, does not generalizes to new maps.

Method

See Alg 1

Future work

Items to improve the algorithm:

- TODO: Justify the computational cost of constraint The cost of going through the entire state space. How do you extend to a network? and large state spaces.
  1. Observation 1: If there is only one goal, then the computation should not be any more than Q-learning. This can be accomplished by assuming that transitivity is satisfied till s_{t-1} and needs to be extend to only the next step. This sounds similar to the Floyd-Warshall dynamic programming update. However, this assumes that s_t is being visited for the first time. If the state s_t is being visited for the second time, the earlier value may be the shorter path for it.
- TODO: Drop Q Value Maintains another Q-value. Why do you need Q-value apart from FW-value? We need only goal reward and add it to the entire state space.

Algorithm 1: How to solve small windy grid world with randomized goals?

Let $r_g \leftarrow 10$;
Initialize $F(s_i, a_i, s_j; \theta_F) \leftarrow -100$;
Let minimum path reward $F_0 \leftarrow -0.05$;
Initialize empty replay memory $\mathcal{R} \leftarrow \{\}$;
Initialize empty key frames $\mathcal{S}_K \leftarrow \{\}$;
Observe z_0 ;
$s_0 = \Phi_o(z_0; \theta_E)$;
for $t \leftarrow 1$ **to** T **do**
  Take action a_{t-1} ;
  Observe z_t, r_t ;
  $s_t = \Phi_o(z_t; \theta_E)$;
  if $r_t > r_g$ **then**
    $s_g = s_t$;
    break;
  Record transition $\mathcal{R} \leftarrow \{(s_{t-1}, a_{t-1}, r_t, s_t)\} \cup \mathcal{R}$;
  $F(s_{t-1}, a_{t-1}, s_t) \leftarrow r_t + F_0$;
  for $s_k \in \mathcal{R}[s < s_t], a_k \in \mathcal{A}$ **do**
    $F(s_k, a_k, s_t) \leftarrow$
    $\max\{F(s_k, a_k, s_t), F(s_k, a_k, s_{t-1}) +$
    $F(s_{t-1}, a_t, s_t)\}$;
    if $s_t \in \mathcal{R}$ **then**
      $\mathcal{S}_K \leftarrow \{s_t\} \cup \mathcal{S}_K$;
      for $s_g \in \mathcal{R}[s > s_t]$ **do**
        `/* Floyd-Warshall update */`
        $F(s_k, a_k, s_g) \leftarrow$
        $\max\{F(s_k, a_k, s_g), F(s_k, a_k, s_t) +$
        $\max_{a \in \mathcal{A}} F(s_t, a, s_g)\}$;

Result: To follow the shortest path s_i to s_j, follow the neighbors with highest Q;
$\chi(s_k) = \arg\max_{a_k \in \mathcal{A}} F(s_k, a_k, s_g)$;

Experiments

- Grid world: Set up a random goal static maze scenario, compare with normal Q-learning.
- TODO: Lava world: Set up a Lava world like Schaul et al. (2015) and test on it.
- TODO: Atari games: Compare performance with normal Q-learning.
- TODO: Deepmind Lab: Set up a random goal static maze scenario, compare with normal Q-learning.

Grid world with dynamic goals

We setup a randomly generated grid world and select a randomly generated goal state.

References

Andrychowicz, M., and Kurach, K. 2016. Learning efficient algorithms with hierarchical attentive memory. *arXiv preprint arXiv:1602.03218*.

Pong, V.; Gu, S.; Dalal, M.; and Levine, S. 2018. Temporal difference models: Model-free deep rl for model-based control. *arXiv preprint arXiv:1802.09081*.

Schaul, T.; Horgan, D.; Gregor, K.; and Silver, D. 2015. Universal value function approximators. In *International Conference on Machine Learning*, 1312–1320.