Transfer Learning in Reinforcement learning with application to Robotics.

Monica Patel

Abstract— **Autonomous robots have achieved high levels of performance and reliability at specific tasks. However it is important for a robot agent to be able to adapt to the new environment and learn varying tasks. In Reinforcement learning agent learns by interacting with the environment and gathering data. Therefore learning different tasks in isolation can be very expensive for a physical agent both in terms of computation and actual physical cost of the Robot. Transfer learning can be used in such cases to learn in simulated environment and using the learned knowledge on actual physical robot to avoid damage as well as to speed up vanilla RL algorithms. This report focuses on understanding transfer learning problem in reinforcement learning domain and applying it to robot navigation task.**

I. INTRODUCTION

Robot agents need to be able to learn from their experience, both in order to deal with changing environments and to adapt to new problems. Reinforcement learning paradigm is popular for such agents. Several methods aim at improving learning by introducing additional knowledge into the exploration choices. This report studies few central ideas of transfer learning and demonstrate how it can aid in learning process. A policy reuse method is used in the project as transfer method where state action values are used as knowledge transfer from previously learned task. The project also demonstrate working of PRQ- algorithm on robot navigation task where new policy is learned from set of older policies. Evaluation metrics for transfer learning are also studied in the report and results are evaluated using two of these metrics.

II. RELATED WORK

Many methods are proposed for knowledge transfer using different aspects for transfer learning. Some transfer knowledge between task by initializing state action values of new task using previously learned action values.[4,5]. Many other algorithm focus on source task to target task mapping [6,7]

III. MAZE NAVIGATION TASK

Finding a path in unknown environment is classic problem in Robotics. For the course of simplicity and to visualize the working and results of the reinforcement learning algorithm the optimal path finding problem is simplified. In the experiment the world is NxM grid with agent allowed for discrete actions - moving $\{Up, Down, Left, Right.\}$ Agent begins at start location and receives a reward of 100 upon reaching the goal. For every step that agents takes it receives a negative reward of -1 and if while traveling the agents bumps into the wall it receives the negative reward of -100 and stays at its own location.

There are many ways by which source and target task differ in transfer learning problem. This has significant impact on abilities of transfer algorithm. For the project two problems are taken into consideration-

- When the goal location is different in the target task than source task.
- Location of the walls in the target task are different than the source task.

IV. WHAT CAN BE TRANSFERRED

The transfer learning methods are classified on the basis of type and level of knowledge transfer. There can be lower level knowledge transfer like $< s, a, s', r >$ instances, policies, action-value functions or the complete model. There can also be higher level knowledge transfer like partial policies eg. options [9] such as options or reward shaping.
The project focuses on lower level knowledge transfer which can be directly used by the agent in target task to fully define initial policy.

V. TRANSFER LEARNING ALGORITHM

Fernandez and Veloso [1] proposed a method called as *Probabilistic Policy Reuse* in 2006 which can be used in the maze task where the goal location in target task

is different than the source task. The algorithm uses Q-learning as its base reinforcement learning algorithm to learn the source task. Both Q- learning and Policy reuse are described in following sections.

A. Q - Learning

Q- learning is *off-policy* TD control algorithm [8] defined by following update rule:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[R_{t+1} + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t)] \quad (1)$$

B. Policy Reuse

In reinforcement learning a Markov Decision process is defined as tuple $< S,A,R,T >$ where S is set of states, A is set of action R is stochastic reward function $R : SxA \Rightarrow Real\ Value$, T is stochastic state transition function, $T : SxAxA \Rightarrow Rea\ value$. In addition to above formalism for the transfer problem, task specific reward function is introduced. A domain D is characterized by tuple $< S,A,T >$ task is defines by Ω as tuple $< D,R_\Omega >$ where R_Ω is stochastic reward function for that task.

In a transfer problem we are given a target task Ω to be solved, i.e we want to learn π^*_Ω and there is an oracle which returns best matching old policy π_{old} the π-reuse exploration strategy gives the algorithm to use π_{old} to best learn the optimal policy on the target task.

The learned policy π_{old} is learned using direct using direct RL method, like Q learning. The goal of $\pi - reuse$ is to balance random exploration, exploitation of new policy and exploitation of old policy. This is done using the following equation.

$$a = \begin{cases} \pi_{old}(s), & w/prob.\psi \\ \varepsilon - greedy(\pi_{new}(s)), & w/prob.(1-\psi) \end{cases}$$

The π-reuse strategy follows the past policy with probability ψ and it exploits the new policy with probability $1-\psi$. It exploits the new policy using $\varepsilon - greddy$ strategy. The following shows the algorithm to learn new policy using π-reuse. The procedure gets input π_past runs for K number of episodes with H maximum steps per episode. The episode also ends if the agent reaches the goal.

Algorithm 1 $\pi - reuse(\pi_{past}$,K,H,ψ,v)

Initialize $Q(s,a)^{\pi_{new}} = 0\ \forall_s \in S, \forall_a \in A$
For k=0 to K-1
  Set the initial state s, randomly.
  Set $\psi_1 \leftarrow \psi$
  for h = 1 to H
    With Probability of $\psi_h, a = \pi_{past}(s)$
    With Probability of $1-\psi_h, a = \varepsilon - greedy(\pi_{new}(s))$
    Receive the next state s' and reward $r_{k,h}$
    Update Q^π_{new} using Q-learning update rule from equation 1
    Set $\psi_{h+1} \leftarrow \psi_h v$
    s $\leftarrow$ s'
W = $\frac{1}{k}\sum_{k=0}^{K}\sum_{h=0}^{H}\gamma^h r_{k,h}$
Return W, $Q^\pi_{new}(s,a)$ and Π_{new}

VI. LEARNING FROM POLICY LIBRARY

The $\pi - reuse$ algorithm assumes that there is an oracle which returns the best matching policy from library of policy to be used. This is done using similarity measures between the policies. For the two problem studied in the project this can be done using using two different ways.

- For the problem where the wall location of target task is different than source task, the best old policy would be the one which is learned in environment which is closely represented as target task. Meaning the wall position of the source task is similar to target task. Here domain level advice can be used to make selection decision about the best old policy to be used.
- For the problem where goal location is different in target task than source task, policy selection function can be used which can be learned online while learning the new policy.

A. Exploiting Domain Knowledge

As described earlier for the problem where the wall location of target task is different than source task domain level advice can be used to select best old policy. In the Navigation task the world can be represented as grid which each grid having probability of being occupied or unoccupied [3]. This grid-map is given to robot as a gray scale image with while in grey image shows unoccupied space and black shows occupied spaces and gray area unexplored territory as shown in the Figure 1. If source policies are learned over such images of maze, the policy which learned on image

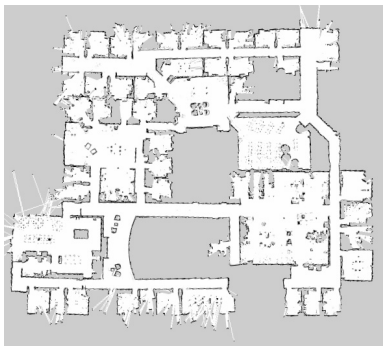


Fig. 1: Willow garage grid map

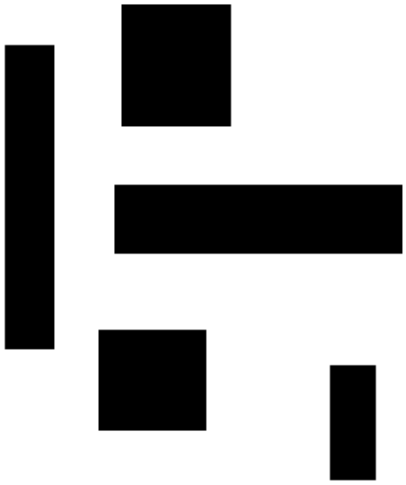


Fig. 2: $\pi - reuse$ results with one policy

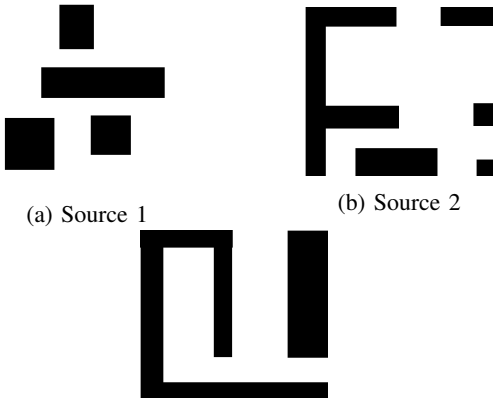
with similar black white and gray area to target task will perform best while reuse. Therefore best policy can be selected using image similarity measures like Mean Squared Error method over pixel values. Choose the policy learned on the environment which minimizes the MSE value in following equation.

$$MSE = 1/mn \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [Target(i,j) - Source(i,j)] \quad (2)$$

For the toy-world used in project if the target world is given by Figure 2 and other worlds are in Figure 3a 3b 3c the policy learned on world Figure 3a is chosen for reuse.

B. PRQ - Learning Algorithm

When we have policy library two main questions to needed to be answered are: (1) Given a set of policy which policy is exploited? (2) Once the policy is chosen what exploration and exploitation policy should be followed? To answer first question a similarity function between policies is defined.



(a) Source 1 (b) Source 2

(c) Source 3

Fig. 3: Source Tasks

Definition Given a policy π_i that solves a task $\Omega_i =< D, R_i >$ and a new task $\Omega =< D, R_\Omega >$, the Reuse gain of the policy π_i on the task Ω, W_i is gain obtained when applying the $\pi - reuse$ exploration strategy with the policy π_i to learn policy π. The most useful policy to reuse π_k then is which maximizes this gain.

$$\pi_k = arg_{\pi_i} \max(W_i), i = 1, ...n \quad (3)$$

This can be found online using PRQ-Algorithm. In PRQ algorithm, let W_i be the Reuse Gain of the policy policy π_i on the task Ω, τ be the temperature parameter and $\delta\tau$ be the decay parameter. A softmax strategy defined by using W value given by equation 4 can we used to select policy.

$$P(\pi_j) = \frac{e^{\tau W_j}}{\sum_{p=0}^{n} eW_p} \quad (4)$$

The policy from the library are selected using the probability obtained from above equation. Algorithm for PRQ-Learning is given below:

Algorithm 2 $\pi - reuse(\pi_past$,K,H,ψ,v)

Initialize Q(s,a) = 0
Initialize W_i = 0
Initialize the number of time a particular policy i is chosen U_i = 0
For k = 1 to K do
    Choose a policy using the probability given by equation 4.
    Execute the learning episode using policy chosen π_k. Update the new action value function Q^{π}_{new} and observe reward R.
    Set $W_k = \frac{W_k U_k + R}{U_k + 1}$
    Set $U_k = U_k + 1$
    Set $\tau = \tau + \delta\tau$

VII. EVALUTION METRIC

There are many metrics to measure the benefits of transfer in the target task.

1) The **jumpstart** is the difference in initial performance of an agent using transfer, relative to learning without transfer.
2) The **asymptotic performance** (i.e., the agents final learned performance) may be changed via transfer.
3) The **total reward accumulated** by an agent (i.e., the area under the learning curve) may differ if it uses transfer, compared to learning without transfer.
4) The **transfer ratio** is defined as the total reward accumulated by the transfer learner divided by the total reward accumulated by the non-transfer learner
5) The **time to threshold** measures the learning time (or samples) needed by the agent to achieve a pre-specified performance level.

The results obtained by the project shows jump-start and asymptotic performance as in Figure 4 5

Parameters for Q - learning were as follows: $\alpha = 0.05$ $\gamma = 0.95$ $\varepsilon = 0.05$ Number of episode K = 1000 and max number of steps per episode H = 2000
Additional parameters for $\pi - reuse$ were as follows: $\psi_i nitial = 1$ at every step it was multiplied by v = 0.95
Additional parameters for PRQ were as follows: Initial Temperature parameter τ = 1 and decay parameter $\delta\tau$ = 0.05

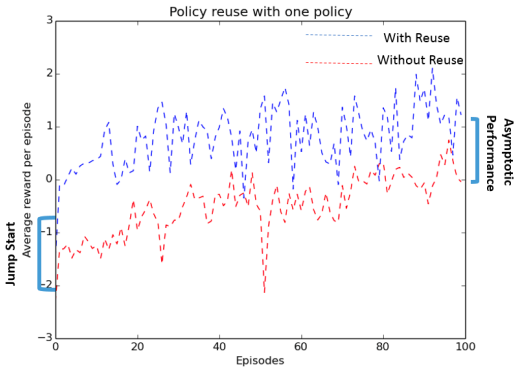


Fig. 4: $\pi - reuse$ results with one policy

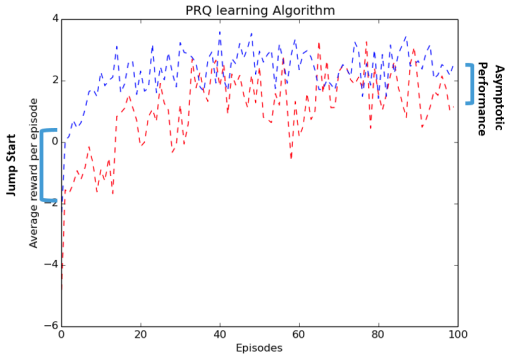


Fig. 5: PRQ algorithm Results

VIII. CONCLUSIONS

Transfer learning paired with reinforcement learning shows significant improvement in the task of learning optimal policy. It can be used when a robot agent interacts with dynamic environment which changes with time. Past learned knowledge can aid to better and faster learning of robot agent in such scenarios. Many other approaches can be used for knowledge representation and transfer to improve result qualities.
PRQ algorithm allows to probabilistically bias exploration learning process. It can further improved to reuse across different representational frameworks. Its can also be expanded over multiple agents.

REFERENCES

[1] Fernando Fernandez, Manuela Veloso, Probabilistic Policy Reuse in a Reinforcement Learning Agent.

[2] Matthew E. Taylor, Peter Stone, An Introduction to Inter-task Transfer for Reinforcement Learning.

[3] Frank Dellaert ,Dieter Fox, Wolfram Burgard, Sebastian Thrun, Monte Carlo Localization for Mobile Robots.

[4] J. Carroll and T. Peterson. Fixed vs. dynamic sub-transfer in reinforcement learning. In Proceedings of the International Conference on Machine Learning and Applications, 2002.

[5] M. G. Madden and T. Howley. Transfer of experience between reinforcement learning environments with progressive difficulty. Artificial Intelligence Review, 21:375398, 2004.

[6] M. E. Taylor and P. Stone. Behavior transfer for value-function-based reinforcement learning. In The Fourth International Joint Conference on Autonomous Agents and Multiagent Systems, July 2005.

[7] M. E. Taylor, P. Stone, and Y. Liu. Value functions for RL-based behavior transfer: A comparative study. In Proceedings of the Twentieth National Conference on Artificial Intelligence, July 2005.

[8] Richard S. Sutton and Andrew G. Barto Reinforcement Learning: An Introduction.

[9] Richard S. Sutton, Doina Precup, Satinder Singh, Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning.