Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

SEAPoT-RL: Selective Exploration Algorithm for Policy Transfer in RL

Akshay Narayan,[†] Zhuoru Li,[†][*] Tze-Yun Leong[†][‡]
School of Computing, National University of Singapore[†]
School of Information Systems, Singapore Management University[‡]
anarayan@comp.nus.edu.sg, lizhuoru@google.com, leongty@smu.edu.sg

Abstract

We propose a new method for transferring a policy from a source task to a target task in model-based reinforcement learning. Our work is motivated by scenarios where a robotic agent operates in similar but challenging environments, such as hospital wards, differentiated by structural arrangements or obstacles, such as furniture. We address problems that require fast responses adapted from incomplete, prior knowledge of the agent in new scenarios. We present an efficient *selective exploration* strategy that maximally reuses the source task policy. Reuse efficiency is effected through identifying sub-spaces that are different in the target environment, thus limiting the exploration needed in the target task. We empirically show that SEAPoT performs better in terms of jump starts and cumulative average rewards, as compared to existing state-of-the-art policy reuse methods.

Introduction and Related Work

This work addresses a class of problems that requires the agent to respond quickly in a new environment, and at the same time adapt to changes. SEAPoT is a new mechanism for maximal transfer of the source task policy to the target task in model-based reinforcement learning. The algorithm detects changes in the target environment as compared to the source, extracts a sub-space around the changed region, limiting exploration to this sub-space, and continues to follow the source task policy when a "known" state with respect to the source environment is reached in the target task.

We address two major issues in this work: (i) How to ascertain that the agent's operational environment has changed? (ii) Once such a change is detected, how to efficiently learn a new policy to operate in the current environment with minimal exploration? We focus on the settings where the source and target tasks share the same state-action space but differ in the transition dynamics and/or reward functions. The similarities between the environments are captured in the shared state-action space, and the differences are represented in a distribution of environmental elements leading to different transition dynamics. In such settings, policy transfer often performs better than other transfer mechanisms because it requires strictly less information being transferred across the tasks.

[*]Currently affiliated with Google Korea, LLC

Existing research on policy reuse includes gathering expert suggestions related to spatial hints (Da Silva et al. 2010), extracting partial policies using structure of the policy space (Hawasly et al. 2013), policy reuse with probabilistic exploration (Fernández et al. 2010), reward shaping to define target task policy from source policy (Brys et al. 2015) and modeling policy selection as Bayesian optimization problem (Rosman et al. 2016). Unlike previous methods, our approach does not require expert suggestions; exploration is kept minimal in the target task, and no knowledge of the source policy structure is required for effective transfer. Applying change detection also eliminates the need to explicitly specify the task as source or target.

The SEAPoT Algorithm

Our policy transfer learning algorithm includes two major steps: (i) Detecting change in the agent's target environment, and (ii) Selective exploration by sub-space extraction.

Change detection: We model the agent's observations as time series, where each data point is derived from an episode in the agent's learning history. Bayesian change point detection (MacKay et al. 2007) is adopted to infer if the latest data point is a result of a change in the environment. This method can handle noisy data due to sensor errors or other environment factors. Change points split the time series into disjoint segments such that, given a change point, data appearing before and after the change are independent of each other. Figure 1 shows an example of how the change detection mechanism works with the observations from one particular state in the system. The red curve shows the raw sensor readings. The blue spike shows the change likelihood when an obstacle is introduced or removed.

Selective exploration: In the target problem, the agent performs exploration in a small sub-space, which is the location where the environment has changed and the agent's historical knowledge is insufficient to complete the task. Selective exploration comprises of three steps: (i) Extract sub-space MDP, M', that is smaller than the original task; (ii) Solve M' to obtain local policy; (iii) Compose the target task solution using the source task and the local policies obtained.

Sub-space extraction: We exploit the agent's knowledge of the source task in solving the target task. Based on

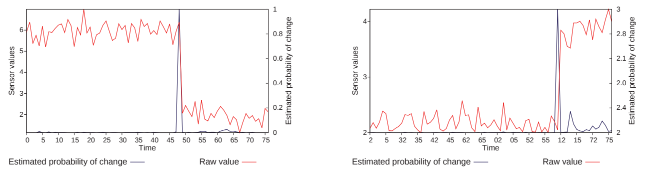
Figure 1: Change detection when an obstacle is added (left) or removed (right)

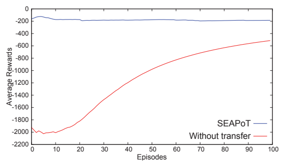
Figure 2: Performance of SEAPoT versus no transfer

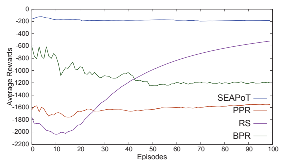
Figure 3: Comparison: SEAPoT, PPR, RS & BPR

a factored representation (Guestrin et al. 2002), where each state in the MDP is represented as a vector of state variables, we define *Adjacency* as: states s_i is adjacent to $s_j \Leftrightarrow \exists a$ such that $T(s_i, a, s_j) > 0$. The notion of *reachability* immediately follows. We extract a sub-space of the target task using the n-*step closure* ($\{s_j | s_j$ is reachable from s_i by taking utmost n actions$\}$). The states reached on executing n actions form the *frontier states* and act as local goals. Using a (sub)set of actions from the parent task and a domain dependent potential based reward function (Ng et al. 1999), which retains the total order of policies, we obtain a well-defined sub-space MDP.

Sub-space exploration: We use Factored-RMAX to explore in the sub-space extracted. The sub-space MDP is visible only in the agent-space; a table look up mechanism provides mapping between states in the parent task and those in the sub-space and vice-versa. Sub-space exploration yields a policy that circumvents the detected change in the environment.

Policy composition: We identify the location of change, and invoke the learned policy in that state. Once the agent is in the frontier state of the extracted sub-space (or the local goal), it can continue to follow the source task policy. An ε-greedy approach is used to enforce active exploration of the target task. If the extracted sub-space is too small, the sub-space policy may lead to a state that is still blocked, requiring multiple iterations of extracting and solving the sub-space.

Experiments, Discussion and Future Work

The experimental set-up is based on a simulated home environment in an assistive care setting. This environment comprises a 10×10 grid world, with four landmark locations (state-space size $= 8000$); the obstacles are walls and/or furniture. The environments differ in the placement of furniture in the home. The agent can perform a total of five actions, `forward`, `left`, `right`, `pick up` and `put down`. The task involves picking up objects from one location and dropping it off at any other location. Reward structure: -1 for each step; -10 for invalid pickup/put down; +20 for successful completion of the episode. The agent transits to the intended state 80% of the time, but has a 20% chance of moving in one of the directions perpendicular to the intended direction. We report the results averaged over 20 runs of 100 episodes each.

We compare our work with three state-of-the-art policy reuse methods (Fernández et al. 2010; Brys et al. 2015; Rosman et al. 2016). See Figure 3. Probabilistic policy reuse (PPR) applies the same action as the source policy, if the outcome of the decision is to use the source policy. Hence the agent draws negative rewards. Bayesian policy reuse (BPR) works well when the set of tasks are known beforehand. When a new task is presented, it selects a policy from the library and reuses it. The BPR agent draws negative rewards due to a similar reason as PPR. The reward shaping agent, on the other hand, has to learn the target task, albeit with help from the shaping signal that is obtained from the source task policy. Hence, the jump start is lost.

We attribute the performance improvement of SEAPoT to the following reasons: (i) The agent reuses the knowledge learned in the source to the maximum extent in the target task; (ii) there is minimal exploration in the target environment.

Our future work includes incorporating active exploration in the target task, policy selection based on context, and using simulation to identify task similarity and partial policy reuse in the target task.

Acknowledgments

This research was partially supported by a SMART Grant: Enhancing the Care for Elderly through Robotic Aides and Academic Research Grant: T1 251RES1211 from the Ministry of Education in Singapore.

References

Brys, T.; Harutyunyan, A.; Taylor, M. E.; and Nowé, A. 2015. Policy transfer using reward shaping. In *AAMAS*, 181–188.

Da Silva, B. N., and Mackworth, A. 2010. Using spatial hints to improve policy reuse in a reinforcement learning agent. In *AAMAS*, 317–324.

Fernández, F.; García, J.; and Veloso, M. 2010. Probabilistic policy reuse for inter-task transfer learning. *Robotics and Autonomous Systems* 58(7):866–871.

Guestrin, C.; Patrascu, R.; and Schuurmans, D. 2002. Algorithm-directed exploration for model-based reinforcement learning in factored MDPs. In *ICML*, 235–242.

Hawasly, M., and Ramamoorthy, S. 2013. Lifelong learning of structure in the space of policies. In *AAAI Spring Symposium: Lifelong Machine Learning*.

MacKay, D., and Adams, R. 2007. Bayesian online changepoint detection. Technical report, SEAS, Harvard. https://goo.gl/b27IfU.

Ng, A. Y.; Harada, D.; and Russell, S. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, 278–287.

Rosman, B.; Hawasly, M.; and Ramamoorthy, S. 2016. Bayesian policy reuse. *Machine Learning* 1–29.