Multi-Robot Training via Transfer-Learning

Aditya Prasad Mishra
ASU ID: 1211165121
User ID:amishr28
amishr28@asu.edu

Piyush Nolastname
ASU ID: 1211198375
User ID:ppapreja
ppapreja@asu.edu

[1]Department of Computer Science – Arizona State University

Abstract. *Training an agent to learn navigation, path planning and obstacle avoidance in a dynamic environment is a difficult problem in itself. It becomes even more tricky when we consider an agent with higher number of degrees of freedom. Reinforcement learning (RL) possesses the ability to find the most optimized action for the current state, through trial and error interactions. Deep reinforcement learning, a variant of RL, goes a step further, through using neural networks in providing solutions to a generalized variety of problems. However, learning a new skill requires considerable data and policy set construction. Both Reinforcement Learning and Deep Reinforcement learning take a huge amount of training time. The challenge here is to reduce the training time. Transfer learning is one of the possible solutions as it enables us to transfer task specific information as modules across morphologies with higher degrees of freedom. We explore the possibility of how to minimize the learning time by doing transfer learning in case we are provided with an already trained module. The task of navigation is our primary interest. We apply the supervised learning approach for this project to test the effectiveness of transfer learning in navigation.*

1. Introduction

This project is aimed towards finding the solution to the problem of knowledge transfer across agents. We approach this by dividing the task of navigation into sub-tasks such as path finding, collision avoidance, thereby separating the task-specific details of the robot specific details such as morphology information, sensor output information, etc. We then combine the learning of each sub-task to make an end to end robot navigation. The idea here is to train robots of varying morphologies in real world scenarios and achieve zero shot generalization with morphologically different robots in unknown environments by applying the concept of transfer learning.
The real world motivation for this project comes from learning approach we humans take. We learn not only from direct observation but also from knowledge transfer. This project tries to implement the idea from [6], which demonstrated the concept of transfer learning on robots with multiple degrees of freedom. However, we will try to tackle the problem of navigation using this approach.

2. Previous Approaches

There are a variety of approaches to make robots learn. The first thing that comes to mind is Imitation learning[10]. Although quite popular, there are several shortcomings in this approach. It is extremely difficult to provide the robot with all possible scenarios in demonstrations. Also, as the environment changes it is observed that the robot has to be trained again from scratch and reusing either the old data or model is not possible. In recent times, Reinforcement learning[7] and [9] has emerged as an interesting area of research towards making a robot learn more effectively. Transfer learning[11] is another popular approach. The importance of transfer learning in the context of reinforcement learning could be adjudged from the fact that an agent takes a considerable amount of time to optimize its path towards the reward. Transfer learning takes into account that some of the tasks have commonalities when transferred across worlds with varying degree of variation or across robots with varying degree of freedoms. This reduces the overall time in optimizing for the whole task again. Past works in transfer learning like the "The PG-Ella algorithm"[5], discuss the policy gradients for sequential multitask learning. [8] discusses a representation in problem-space that is Markov for the particular task at hand, and one in agent-space that may not be Markov but is retained across successive task instances. Our approach aims at reducing shortcomings mentioned in Reinforcement Learning and Imitation Learning and is complementary to approaches mentioned for Transfer Learning.

3. Our Approach

We decompose a policy into robot-specific and task-specific modules, wherein, the task-specific modules can be shared across robots of different morphologies and robot specific modules can be shared across different tasks [6]. Both of these modules are represented as neural networks.
We follow the sensor-based supervised learning approach for training. The task module is represented as a policy by a neural network, which learns on multiple iterations and provides us with the current deviation from straight movement. The proximity sensor readings from the robot from the input to our task module. The output from this module is the direction of the robot, which is used as an input for the robot specific module. The robot specific module using supervised learning or reinforcement learning would give as the output the motor controls for the robot. We then combine the task module with a different robot module to test the effectiveness of our approach.

4. Implementation

Our implementation workflow is as shown in figure 1. We start with preparing V-REP simulation environment and choose Pioneer p3dx as the robot. We use the Sensor Data to train our task module. The label for the sensor readings (9 sensors in total, with each sensor having three values, detectedState, detectedPoint and surfaceNormalVector) is the difference between the left and right motor controls, which intuitively translates to the direction in which the robot should move. After we have trained the task the module, we train the robot module by getting the directions from the task module and then using it along with the Orientation and location parameters as inputs. We also record the motor controls at the next time step. These recordings are used to train the Robot Module. Then, we integrate both of these modules and see how they work together in our testing phase
The input dimension of the task module is 63 (9 sensors with 7 values for each). The

output dimension of the task module is a 5 (5 value vector, with each value representing a direction such as slight left, left, straight, slight right and right. There are 5 hidden units. The Hidden units have Sigmoid activation. The output layer has a Softmax activation to give us the most probable direction at the current time step.
Our robot Module consists of a 7-dimensional input layer, 6 of which comprise of the current orientation and location parameters and 1 for the max predicted probable direction from the previous layer. Robot Module also consists of 4 hidden units and an output layer which would predict regression values for the motor speed at the next time step. These Modules are shown below in figure 2 and 3

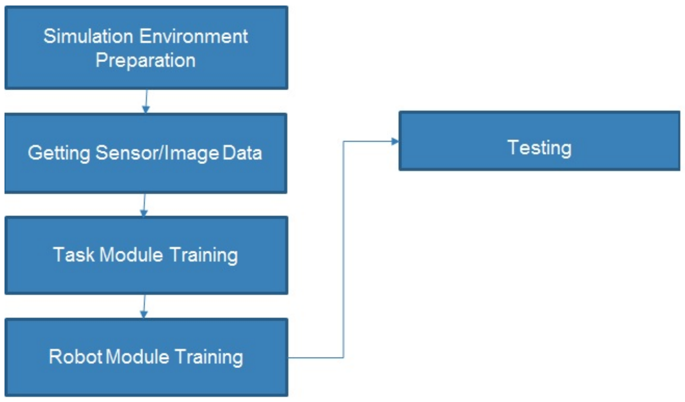

Figure 1. Work Flow

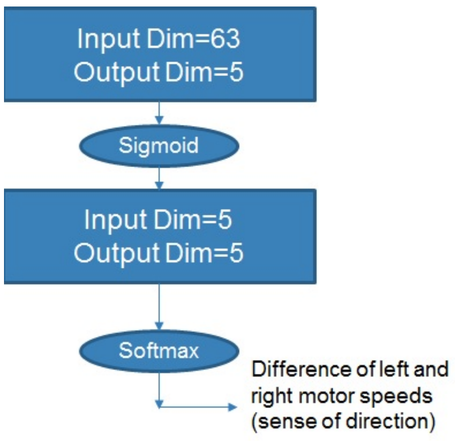
Input Dim=63
Output Dim=5
Sigmoid
Input Dim=5
Output Dim=5
Softmax
Difference of left and right motor speeds (sense of direction)

Figure 2. Task Module

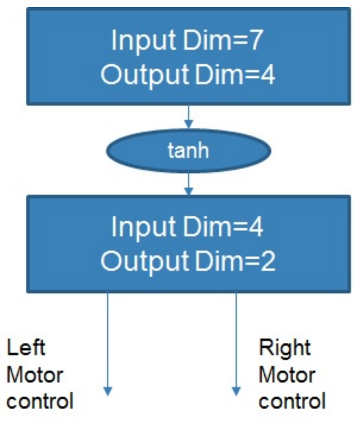

Figure 3. Robot Module

5. Results

Our recorded accuracy with our tested models for data classification is presented in Table 1. Although these are good models for data classification. But they did not provide us with a resulting model which would suitably run our Robot in unknown environments.

Table 1. Accuracy Table

	Training Accuracy	Testing Accuracy
Neural Network - Task Module	81%	84%
Neural Network - Robot Module	65%	71%

6. Challenges

One of the first and major challenges of this project was to select the perfect simulation environment to fit the needs of the project. To implement transfer learning, we needed a simulation environment with robots of considerable different morphology (such as a car and a drone) to test the working of our chosen approach. There are a number of simulation environments available like Udacity[3] self-driving car simulator, Microsoft Airsim[2], TORCS[4] and V-REP[1]. Neither of these simulation environments (except for VREP) fully completed our requirements. While the aforementioned simulators provided expansive environments, rich in detail and the data collection was easy in the form of images (Udacity simulator and Airim) and sensor data (TORCS), the range of vehicles to choose from was not large. Airsim has only drones, the Udacity simulator, only a single type of car and so does TORCS. So the availability of robots with different morphologies requirement ruled these simulators out. V-REP, on the other hand, provides with robots with all kinds of morphology such as p3dx, quadcopter. But it lacks in ease in terms of implementation aspects such as data collection from sensors, supervised training of the robot, everything has to be done manually, unlike the previously mentioned simulators which provide these functionalities off the shelf.
Furthermore, another challenge thrown by VREP is the time lag. The simulation time is different from the real time and this makes the task of data collection and processing arduous.

7. Conclusion and Future Work

The end result of this project left many things to be tried and implemented such as:
1. Reevaluation of the results of our current approach (Sensor based Supervised Learning) with more data and tweaks in our model.
2. Integration of CNN based task module to the existing robot module and compare its performance with sensor-based model.
3. Integration of the Reinforcement learning module to the Robot Module to learn the task of navigation as well as path planning.
4. Implementing this concept in real world scenarios.

8. References

References

[1] http://coppeliarobotics.com.

[2] https://github.com/microsoft/airsim.

[3] https://github.com/udacity/self-driving-car-sim.

[4] http://torcs.sourceforge.net.

[5] Haitham Bou Ammar, Eric Eaton, Paul Ruvolo, and Matthew Taylor. Online multi-task learning for policy gradient methods. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1206–1214, 2014.

[6] Coline Devin, Abhishek Gupta, Trevor Darrell, Pieter Abbeel, and Sergey Levine. Learning modular neural network policies for multi-task and multi-robot transfer. *arXiv preprint arXiv:1609.07088*, 2016.

[7] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[8] George Konidaris and Andrew G Barto. Building portable options: Skill transfer in reinforcement learning. In *IJCAI*, volume 7, pages 895–900, 2007.

[9] Jan Peters and Stefan Schaal. Reinforcement learning of motor skills with policy gradients. *Neural networks*, 21(4):682–697, 2008.

[10] Stefan Schaal. Is imitation learning the route to humanoid robots?

[11] Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(Jul):1633–1685, 2009.