Reinforcement Learning for Automatic Cars

Lihang Liu
EID:ll32632
lihangliu@utexas.edu

1 Introduction

In this report, we will introduce how we apply Reinforcement Learning to the problem of automatic car driving. The basic Q Learning algorithm, SARSA, is used to train our automatic car such that the car is able to avoid possible accidents and reach the goal as fast as possible. The details of the SARSA algorithm is not coverred in this report. Instead, we focus on how to design appropriate states, actions and rewards. In the experimental environment, there are several kinds of objects: cars, traffic lights, parkings and pedestrians, which all need to be avoid. The experiments demonstrate the performance of our algorithm.

2 Environments

In this section, we will introduce the overall environmental assumptions, as well as the behavior of our auto car.

Lanes There are four lanes in the environment with two lanes in each direction. The length of each lane is 15.

Other Cars There are two kinds of cars with opposite directions. The cars are able to change speed from 0 step to 2 steps, as well as chaning lanes. However, they can't drive into a lane with different direction. The actions of these cars for next steps are randomly chosen.

Parkings Parkings here are regarded as obstacles in the fixed position from start to end.

Pedestrians The Pedesrians always appear from one side of the lanes and head for the other side. They are able to stop or move one step forward. The actions of these pedestrians for next steps are also randomly chosen.

Traffic Light The traffic lights have periodical behaviors. They loop along the yellow-green-red patterns and stay at each patterns for 2 time steps.

The Automatic Car The automatic car is initialized in the starting of the lanes with six actions associated: stop, go left, go right, go 1 step forward, go 2 step forward and go 3 step forward. The option of going 3 step forward makes the automatic car more flexiable to avoid other moving objects. The goal of our automatic car is to avoid possible accidents and reach the end of the lanes as fast as possible.

3 Methods

In this section, we will introduce several modules:

1. Two modules of SARSA with different state and reward definitions, the Local-SARSA and the Global-SARSA.

2. A protocol designed to combine these two modules to work simultaneously and cooperate.
3. A module of Improved-Local-SARSA to corparate the Global-SARSA into the Local-SARSA.

3.1 Local-SARSA

The Local-SARSA is designed to perceive local informations around our automatic car in order to avoid possible obstacles or accidents. Note that reaching the goal as fast as possible is not considered here.

3.1.1 States

In order to represent the local information more efficiently, we use a total of 6 binary features, which all concern about the environment configurations of the next step.

1. Whether the current position will be hit by other objects in the next step.
2. Whether the left position will be hit by other objects in the next step.
3. Whether the right position will be hit by other objects in the next step.
4. Whether the position 1 step ahead will be hit by other objects in the next step.
5. Whether the position 2 step ahead will be hit by other objects in the next step.
6. Whether the position 3 step ahead will be hit by other objects in the next step.

The above 6 binary features count for a total of 64 states. The number of available actions as described in the previous section is 6. Thus the resulting Q table is with size of 64×6.

3.1.2 Rewards

As the goal of this module is to avoid accidents, the only rewards used here is:

- Negative reward if the automatic car collides with any other objects after performing the selected action.

3.1.3 Select the Next Action

To balance the exploration and exploitation, we use a ϵ-greedy algorithm to select the next action: the car chooses the action that it believes has the best Q value with probability $1 - \epsilon$, and it chooses an action uniformly at random, otherwise.

3.2 Global-SARSA

The Global-SARSA is designed to perceive the global information of our automatic car in order to reach the goal as fast as possible. Note that avoiding possible obstacles or accidents is not considered here.

3.2.1 States

In order to represent the global information of our automatic car, we encode the current position of the automatic car in the lanes as the states. More concretely, we take each position of the lanes as individual states. When the automatic car is in that position, it's in that state. In our settings, there are totally $4 \times 15 = 60$ states. The number of available actions as described in the previous section is 6. Thus the resulting Q table is with size of 60×6.

3.2.2 Rewards

As the goal of this module is to reach the goal as fast as possible, the only rewards used here is:

- Positive reward if the automatic car reached at the destination (the end of the lanes).

3.2.3 Select the Next Action

The ϵ-greedy policy used here is the same as that described in the Local-SARSA.

3.3 Combine Local-SARSA and Global-SARSA

The Local-SARSA and Global-SARSA only concerns about their own part of task, avoiding accidents and reaching the destination as fast as possible, respectively. In order to combine these two modules, we design a protocol to enable them to work simultaneously and cooperate together.

These two modules have their own Q table, states and reward defintions as described above. The difference is in how to select the next action. Denote a as a set of actions, $s^{(l)}$ as the set of states of Local-SARSA and $s^{(g)}$ as the set of states of Global-SARSA. Also, we have Q table $Q^{(l)}$ for Local-SARSA and $Q^{(g)}$ for Global-SARSA. Assume for time step t, the automatic car is in state $s_i^{(l)}$ for Local-SARSA and $s_j^{(g)}$ for Global-SARSA, we select the next action:

$$a_k = \arg\max_{a_k \in a}(Q^{(l)}_{s_i^{(l)},a_k} + Q^{(g)}_{s_j^{(g)},a_k}) \tag{1}$$

with probability $1 - \epsilon$, otherwise it chooses an action uniformly at random, similar to ϵ-greedy policy.

3.4 Improved-Local-SARSA

The above protocol combines the local information and the global information separately. However, a better module should consider the local and global information simultaneously instead of separately. In this section, we introduce a Improved-Local-SARSA module which consider both avoiding accidents and reaching the destination as fast as possible.

3.4.1 States

The states used for Improved-Local-SARSA is the same as Local-SARSA.

3.4.2 Rewards

In order to reach the destination as fast as possible as well as avoiding accidents, we design the rewards heuristically.

- -2 reward deduction if the selected action is "stop".
- -1 reward deduction if the selected action is "left" or "right".
- $+1$ reward credit if the selected action is "1 step forward".
- $+2$ reward credit if the selected action is "2 step forward".
- $+3$ reward credit if the selected action is "3 step forward".
- -10 reward deduction if the automatic car collides with any other objects after performing the selected action.

The first 5 rewards encourge the automatic car to move forward as fast as possible to reach the destination, while the last reward enforces the avoiding of accidents.

3.4.3 Select the Next Action

The ϵ-greedy policy used here is the same as that described in the Local-SARSA.

4 Results

In this section, we show the experiments on automatic car driving by the SARSA algorithm. In the environments, there are total 2 parkings, 6 cars, 2 pedestrians and 2 traffic lights. If any object reaches its end and get off the lanes, a new object of the same category will be generated and placed in a random starting position. A demo of Improved-Local-SARSA after 10,000 training iterations has also been attached.

The four forementioned methods are compared:

Local the Local-SARSA.

Global the Global-SARSA.

Local-Global the protocol combining Local-Global and Global-SARSA.

Improved-Local the Improved-Local-SARSA.

Two important metrics are used to measure their performance:

- the average time steps took to reach the destination.
- the average accidents encountered before reaching the destination.

For each methods, we tested under different training iterations. The results are shown in Fig. 1 and 2. Also Table 1 is shown below to give some details.

From the figures and the table, we have the following observations and conclusions:

- The Local-SARSA has good performance in avoiding accidents, however, it has the worst performance in getting into the destination as fast as possible.
- The Global-SARSA has sucessfully learned how to reach the destination with the smallest time steps. However, it has the largest accident numbers, as it has no idea about accidents.
- The Local-Global finds the balance between the Local-SARSA and the Global-SARSA. The average time steps took to reach the destination has largely decreased compared to Local-SARSA while its ability in avoiding accidents is as good as the Local-SARSA.
- The Improved-Local-SARSA better utilized the local and the global information compared to the other methods. It has the best performance in avoiding accidents and the second smallest average time steps to reach the destination. The larger time steps costed compared to the Global-SARSA is necessary sometimes to avoid the accidents. This demonstrates the efficiency of our Improved-Local-SARSA algorithm with nice defined states and reward functions.

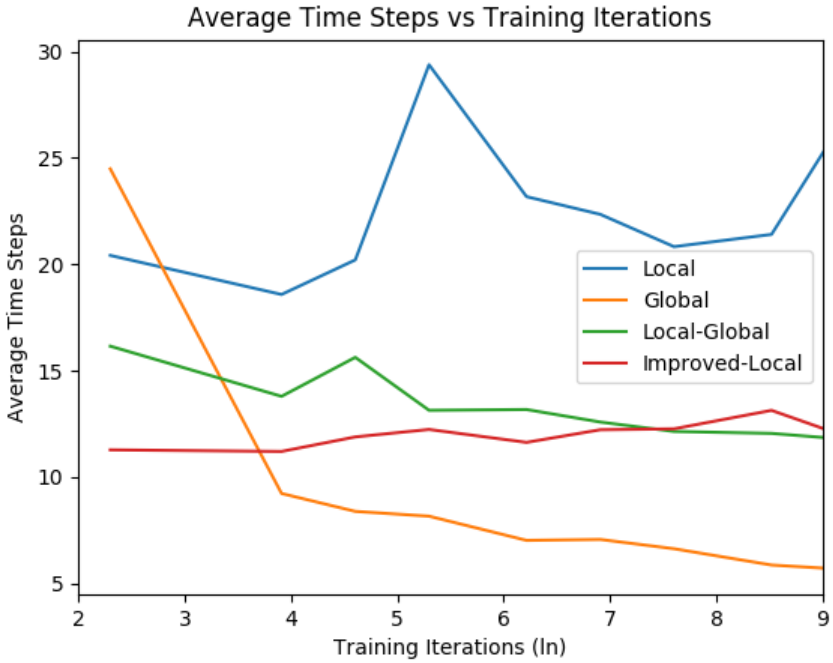

Figure 1: Average time steps took to reach the destination under different training iterations

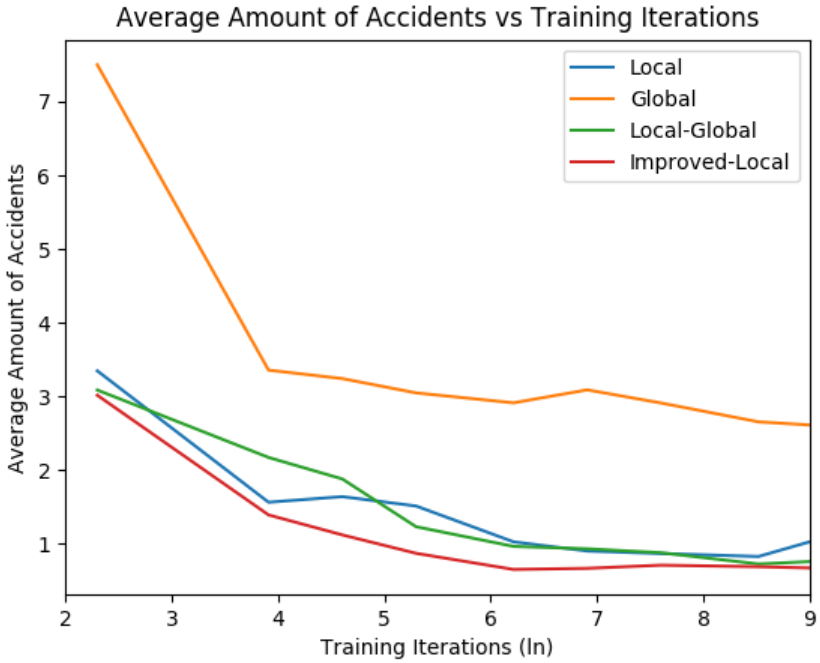

Figure 2: Average amount of accidents encountered before reaching the destination under different training iterations

Table 1: Average time steps and average amount of accidents under 10,000 training iterations

Methods	Average Time Steps	Average Amount of Accidents
Local	26.912	1.115
Global	**5.652**	2.594
Local-Global	11.774	0.773
Improved-Local	11.912	**0.665**

5 Conclusion

In this report, we show how Reinforcement Learning can apply to the problem of automatic car driving. In the experiments, we show that our automatic car can effectively avoid other objects and maintain only an average of 0.665 while there are totally 12 random objects and in the lanes of size 4×16.