Report on

"Multiple Cops Catching a Criminal using Multi Agent Reinforcement Learning"

Submitted in partial fulfillment of the requirements for ***Sem VI***

Topics in Deep Learning

Bachelor of Technology
in
Computer Science & Engineering

Submitted by:

Saahil Jain	**PES1201700243**
Jeevana R Hegde	**PES1201700633**
Rishabh Jain	**PES1201700099**

Under the guidance of

Srinivas K S
Professor
PES University, Bengaluru

January – May 2020

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

Project Description

In our project a group of cops are trained to chase and catch a criminal by reinforcement learning. The criminal is programmed to dodge cops. Four test cases have been taken, two for 2 cops and two for 3 cops. Out of the two, in one case the Q table is shared amongst the cops and in the other, they have their independent Q tables. The episode changes either when the criminal is caught or once 200 steps have been taken.

Reasons to Choose this Project

Reinforcement learning is an active area of research hence the project was opted in this domain. To learn and gain more insight about the cooperativeness of multi-agent reinforcement learning, this topic was chosen.

Uniqueness and Comparison of our Solutions

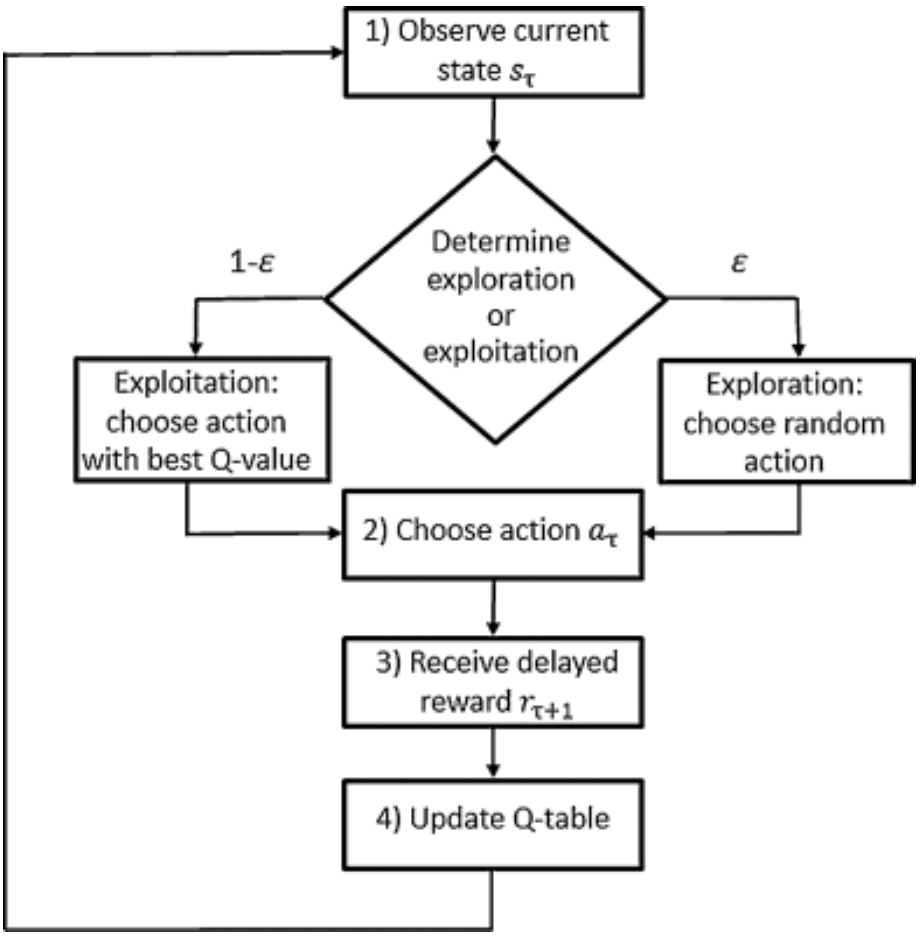
In generic cases of reinforcement learning, a common Q table is shared amongst the various agents and complete visibility is provided in a small environment like the one used in our project. However, to reduce the size of the Q table, partial visibility has been provided to the agents.

Scope of our Project and Additional Information

As and when technology and AI play a bigger role in real life, this could be a possible application for robo-cops, or autonomous drone clusters in army.

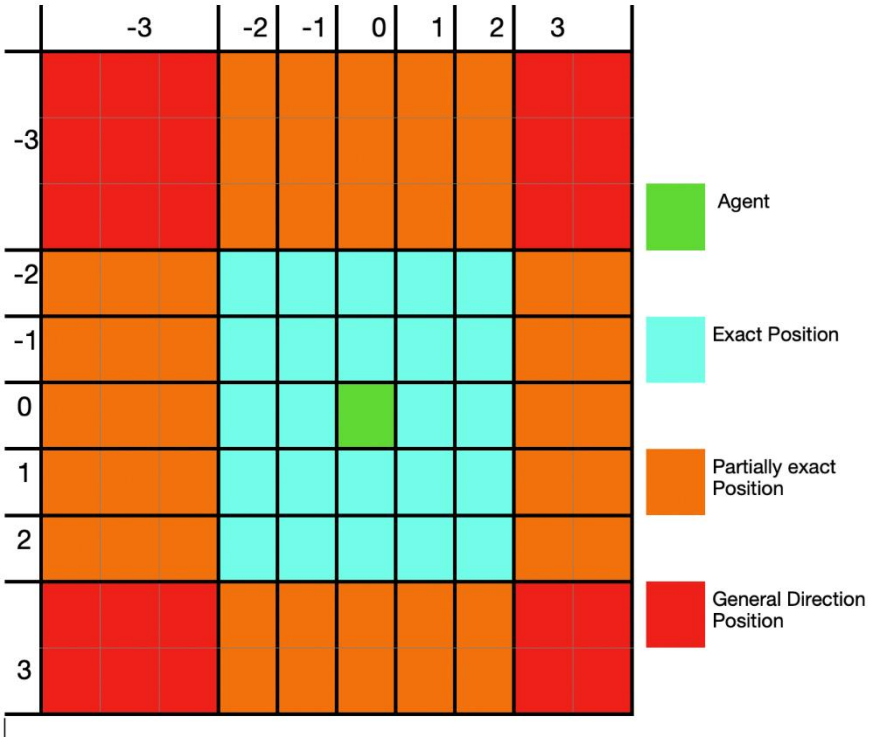
The reason for choosing shared Q tables is for quicker learning and smaller model size. The reason for choosing independent Q tables was assuming that different agents would opt for different roles.

Pictorial Representation



RELATIVE POSITION CALCULATION FOR AGENTS

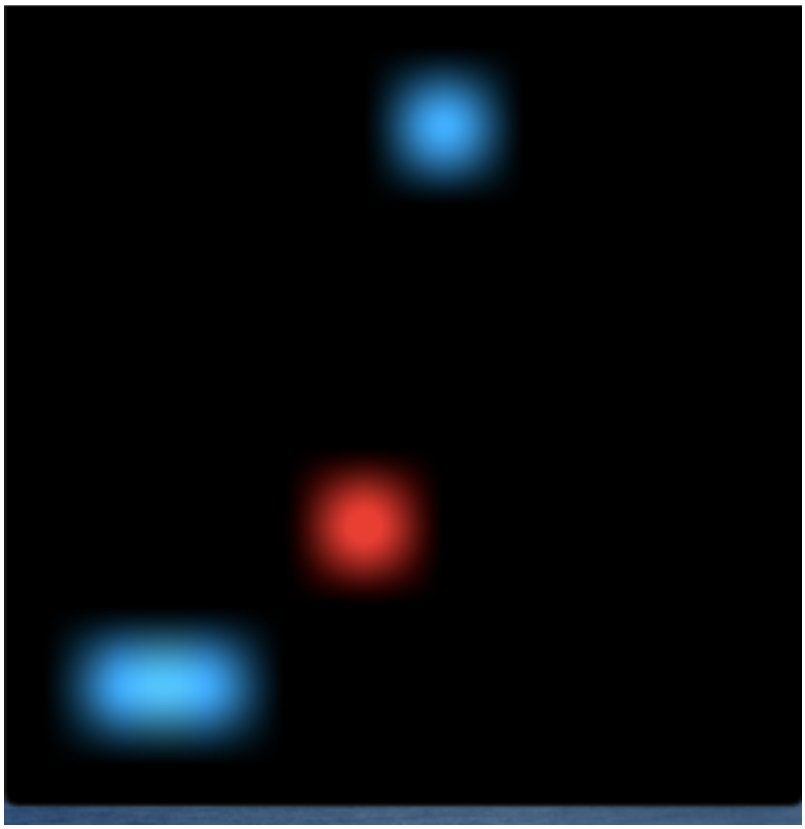
For a grid of size 10x10 and Sight being 3,
the visibility of the agent is as follows



Description:

- A Q table is used to make predictions
- The discrete state/observations that the agents see are represented as the rows of the Q table.
- The various actions it can perform which are UP, DOWN, LEFT or RIGHT are represented as columns.
- To reduce the grid space, relative positions of the other agents and criminal are provided to each agent.
- This is further reduced by limiting the sight of the agent as shown in the diagram.

Screenshot



```
Using Different Q-tables :
1 : STEPS : 30
2 : STEPS : 332
3 : STEPS : 35
4 : STEPS : 322
5 : STEPS : 3
Catching Percentage    : 60.0
Average Steps To Catch : 144.4

Using Same Q-tables :
1 : STEPS : 7
2 : STEPS : 3
3 : STEPS : 14
4 : STEPS : 6
5 : STEPS : 309
Catching Percentage    : 80.0
Average Steps To Catch : 67.80000000000001

3 AGENTS :

Using Different Q-tables :
1 : STEPS : 32
```

Solution

Our 4 solutions are:

1. 2 agents using shared Q table
2. 2 agents using separate Q table
3. 3 agents using shared Q table
4. 3 agents using separate Q table

The algorithm, math and testing carried out for all 4 cases are the same for all the solutions.

I. Algorithm -

The new reward for the current action in a particular state is the sum of a fraction (1 – learning_rate) of its current value and a fraction(learning_rate) times sum of current reward and maximum future reward.

II. Math –

Given a Q table Q with the possible discrete states as rows and possible actions as columns, updating Q table is as follows:
Given state S and action performed as A,

Q[S][A] = ((1-learning_rate)*Q[S][A])+(learning_rate*(current_reward + (discount * max(Q[perform_action(A)])))

perform_action(A) returns the new state when the agent performs the action.

III. Testing Carried Out –

Each of the 4 solutions were tested for 100000 iterations. The number of times the cops succeeded in catching the criminal is taken in percentage.

IV. Results and performance–

	2-shared	*2-Independent*	*3-shared*	*3-independent*
Catching Percentage	74.727	75.054	78.601	81.494
Average Steps to Catch	106.81956	105.29157	101.90124	93.26331

Constraints

Cop must catch the criminal within 200 steps. If the cops are unable to catch him, the criminal wins.

Assumptions

- At each step, cop and criminals have the same speed.
- Grid is assumed to be wrapped around (Ex: -1 is n-1 if cells are assumed to be numbered from 0 to n-1. True for x and y direction.)
- Grid is assumed to be free of obstacles
- Agent is unaware of whole picture of environment. They are rather known to have the relative positions of the criminal as well as other agents at all times.

Future Work Plans

- We can provide a more visual feed rather than relative positions. This would let the model be applicable to any number of agents.
- Environment can be made more realistic by taking into consideration some obstacles.

References

Lecture notes

[1] Srinivas K.S
https://drive.google.com/open?id=1x2w5T-zNNcBlKhuwpDJbt3xEiITtnucwL

Internet tutorial

[2] Sendex, Q-Learning introduction and Q Table - Reinforcement Learning
https://pythonprogramming.net/q-learning-reinforcement-learning-python-tutorial/