Reinforcement Learning with Deep-Q-Networks to Train Soccer Agent

I. Abstract

Training an artificial intelligence to do humanlike task has always been a major goal for AI scientists. In this project, we try and use a reinforcement learning scheme to train an agent to play soccer in a simulated environment in Unity. In this project, various reward functions are used.

II. Environment

In the environment, there is a soccer field setup and 4 players, two for each team. Each player have their own individual movement that programmers can set. After each goal, the environment will be reset. As observation data, each player have 14 sensors, 11 of which are located in front of the player, the other 3 is on the back side of the player. Each of this sensor return datas that contain information of what is detected in each sensor and how far it is. The summary of all the information is as follows :

Name	Information
Environment	Unity
Number of Players	2 per team, 2 teams
Sensors per player	14, 11 front, 3 back
Sensors data	3 stack, each stack containing 8 data stating what is seen and the distance from it. Each stack of sensor represents reading at some point.
Reset Trigger	Goal or Overtime
Action Space for each Players	360 degree rotation, forward, backward, left, and right.

III. Models

A. First Model

The initial model for the soccer agent done by Jonathan Willianto is a single neural network system for both players. The observation input is done by appending all stack in an array. Each sensor reading represents 2 inputs, one is the distance, the other one is the index of the data representing what is seen. This gives a total of 168 input datas. The action space that each actor have is a single movement in a given direction (either forward, backward, rotate left, rotate right, left, or right). Therefore for two players, this gives 18 possible movements. The model is trained with reward -1 on suicide goal and +1 - time_penalty for goals. Summary is in the following table :

Name	Information
Neural Network Size	168 -> 256 -> 256 -> 18
Input Size	(Number of Sensor * Stack * 2 * 2) 14 * 3 * 2 *2 = 168
Output Size	Each agent can only have one action at one time. Eg. Agent 1 rotate left, Agent2 rotate right, etc. Size : 18
Reward	-1 for suicide goal +1 - time penalty for goals

This initial model yields poor result and is therefore conceded due to the poor result. Factors might include the size of the action spaces and the small choices of action offered.

B. Separation of Actors

Moving on, the second model is when the actors are separated. In this model, each teams have each one keeper and one striker. Reward functions and action spaces are defined differently for each actor. In addition, due to the fact that using sparse rewards results in slower training, in this model, a continuous reward is added. Additionally, the inputs for this model have also been changed as the previous input abstracts information away from the neural network. In this model, input are unparsed and are all given to the neural network. Strikers are allowed to move with in any direction and also rotate themselves while keepers cannot rotate and can only move forward, backward, turn left or right. In addition, striker and keeper can only do one action at a time, meaning they cannot rotate while moving forward or turn left while moving forward. In general the datas for this training is as follows :

Name	Information
Neural Network Size	336 -> Hidden Layer -> 4 or 6
Input Size	3*14*8 = 336
Output Size	4 for Keeper (No Rotation), 6 for Striker
Reward	+0.001 for Keeper every timestep - 0.001 for Stiker every timestep - 1 suicide goal for keeper - 0.1 suicide goal for striker + 1 goal for striker + 0.1 goal for keeper

C. Chase Ball and See Ball

Due to the slow training time using the rewards in the previous methods, the new model adds more human bias while keeping other parameters the same. Inputs are parsed in the same way while rewards are added. In this model more rewards are added. In general, adding more human bias will increase the learning speed as the machine will not need to learn the feature anymore.

First reward added being a reward given when the striker detects a ball in one of its sensor. The other reward added being a reward given when the striker is very close to the ball. This reward shows great improvement in training time but not in the direction of the goal as the striker can look at the ball or be near the ball without kicking it. Nevertheless, the idea motivated from this code soon will prove to increase the convergence speed of the neural network when modified to a kick ball reward in later models.

D. **Kick Ball**

The final reward is a modified version of the previous see ball and chase ball reward with rewards only given when the ball is kicked. In addition, the reward is spread over three observations. The other rewards are the same as the previous models with slight modification of the value (increase).

IV. Conclusions

In general, we can conclude that deep q learning can successfully train an agent to play football. Moreover, adding a human bias into the game such as the chase ball and kick ball can increase the convergence speed of the model by a great margin as results can be seen after training for a small amount of time or episodes.

V. Contributions

- Jonathan Willianto -> First code, chase ball, see ball, actor separation
- Abdirakhman Ismail -> Ideas on reward functions and action spaces, environment setup
- Oleksii Nasypanyi -> Final code and kick ball
- Tien Dat Nguyen -> Chase ball, see ball, actor separation
- Nattawong Chinworakij -> Final Presentation

VI. References

- Covert, Christoper, McMillan, Cameroon , and Pipatpinyopong, Patipan. CS230 Project Report. Cooperative - Competitive Multi Agent-Learning in Soccer Environments. <http://cs230.stanford.edu/projects_winter_2019/reports/15811878.pdf>