Team 9: Artificial Intelligence (Project Report)

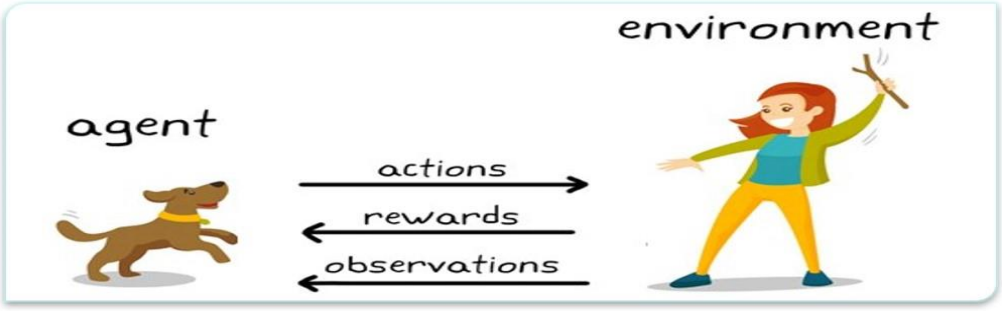
Project Name: Reinforcement learning on Snake Game

Team Members: Khurram Hajizada, Shafahat Sardarov, Rashad Gasimov

Reinforcement Learning for Snake game

Snake is a sole player video game where the player controls a line to eat items by running into them with the head of the line. Each item eaten makes the snake longer, so controlling is progressively more difficult. It becomes primary obstacle to itself as it grows. Here our agent is the snake and our environment is a bordered plane. The aim is here to get highest score. For getting the highest score we have to concern about staying alive and finding food.

The solution of snake game is Reinforcement Learning algorithm. Reinforcement learning (RL) is an area of machine learning concerned with how software agents ought to take actions in an environment in order to maximize some notion of cumulative reward. In this project, snake should learn environment based on actions. After learning environment of canvas, the aim of snake to find food. And finally it gets reward from environment. The process is similar to below picture:



```
function game_Run(){

        // rl initialize Environment
        if(lv_init){
           lv_init = '';
           environment = new rl_Snake();

           paint();
           lv_state   = environment.getState();
           lv_Act = environment.getAction(lv_state);
           environment.implementAction(lv_Act);
           paint();

        }
        else{
            if(!lv_Reset){
                environment.reward(lv_state,lv_Act);//Reward and learn
            }
            else{
                paint();
                lv_Reset = '';
            }
            if(started == true){
               lv_state   = environment.getState();
               lv_Act = environment.getAction(lv_state);
               environment.implementAction(lv_Act);
               paint();
               score_Update();
               //checkGame();
            }
            else{
                score_Update();
                game_Check();
                lv_Reset='X';
            }
        }

    }
```

In these piece code, we first create instance of Snake. Then we get 6 states(goAhead, goLeft, goRight, foodAhead, foodLeft, foodRight) and each function 0 or 1 values. Then we get the action by Q-table. Q-table is action – state pair. Paint function is repaint the canvas every given milliseconds.

```
var rl_Snake = function() {
     this.reset();
    }
    rl_Snake.prototype = {
        reset: function(){
          this.alpha = 0.1;//Learning rate 0 -1
          this.gamma = 1;//Discount factor 0-1
          this.aLoop = 0;
          this.food = {};
        },
        getState: function(){

            var s=[];
            var px = snake_array[0].x;
            var py = snake_array[0].y;
            var p1=0,p2=0,p3=0;

            if(started == false){//Game over state
               return [0,0,0,0,0,0];
            }

            //Is it clear left 0 - No 1 - Yes
            if(d=="right"){//ahead:x++;left=y--;right=y++
                p1=px + 1;
                p2=py - 1;
                p3=py + 1;
                s.push(checkIfClear(p1,py));//check clear ahead
                s.push(checkIfClear(px,p2));//check clear left
                s.push(checkIfClear(px,p3));//check clear right

                s = s.concat(checkIfFood(px,py,d));//check food
            }
```

From here, inside of prototype we put some functions like getState, getAction, implementAction, and etc. In getState we mentioned 6 states is ahead clear?, is right clear?, is left clear?, is food ahead?, is food left?, is food right?.

```
reward: function(s, a){
            var rewardForState=0;
            var futureState = this.getState();

            var lv_string_c = JSON.stringify(s);
```

```
            var lv_string_f = JSON.stringify(futureState);
            if(lv_string_c != lv_string_f){
                    //If snake collides, reward = -1
                    //If snake gets closer to food, reward=1
                     if((s[0]==0 && a==0) || (s[1]==0 && a==1) || (s[2]==0 &&
a==2)){
                       rewardForState=-1;
                     }
                     if((s[0]==1 && a==0 && s[3]==1) || (s[1]==1 && a==1 &&
s[4]==1) || (s[2]==1 && a==2 && s[5]==1)){
                       rewardForState=1;
                     }
```

In reward function, if snake get closer to food we gave 1, if not -1 values.

In addition, in code file “script.js” belongs to us. We used the codes which in “jq.js” file for user interface part.