Name : Ebrahim ElGaml
ID: 28-4498
Name: Raghda Khaled
ID: 28-3256

Machine Learning Report

1- The learning algorithm is the grid world game where we have a 4 by 4 grid consisting of player1,player2, wall,pit and a goal. The player moves either up,down,right or left. If a player faced a pit, reward becomes -10 and loses. If a player reached the goal, reward = 10 and wins. This learning algorithm is applied before the training algorithm being applied to the 4 interactive games.

2- In the 1000 epochs → When running the test algorithm and the game is played automatically, we lost
In the 700 epochs → When running the test algorithm and the game is played automatically, we lost
In the 400 epochs → When running the test algorithm and the game is played automatically, we lost
In the 100 epochs →

For this Grid :
```
array([[u' ', u'P', u' ', u' '],
       [u' ', u'-', u' ', u' '],
       [u' ', u' ', u'W', u' '],
       [u' ', u'P2', u' ', u'+']],
      dtype='<U2')
```
In 1000 epochs →
In 700 epochs → Game lost, too many moves (>10)
In 400 epochs → Computer Reward: -10, lost
In 100 epochs → Computer Reward: -10, lost

When playing with the computer:
In 400 epochs → Initial State:
```
[[u' ' u'P' u' ' u' ']
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u' ' u'P2' u' ' u'+']]
Enter your action 0, 1, 2, 3 for up, down , left, right
3
Move #: 0; Computer Taking action: 0
Move #: 0; Player Taking action: 3
```

```
[[u' ' u'P' u' ' u' ']
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u' ' u' ' u'P2' u'+']]
Enter your action 0, 1, 2, 3 for up, down , left, right
3
Move #: 1; Computer Taking action: 0
Move #: 1; Player Taking action: 3
[[u' ' u'P' u' ' u' ']
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u' ' u' ' u' ' u' ']]
Player Reward: 10
```
We won in this case

In 100 epochs →
```
Initial State:
[[u' ' u'P' u' ' u' ']
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u' ' u'P2' u' ' u'+']]
Enter your action 0, 1, 2, 3 for up, down , left, right
3
Move #: 0; Computer Taking action: 3
Move #: 0; Player Taking action: 3
[[u' ' u' ' u'P' u' ']
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u' ' u' ' u'P2' u'+']]
Enter your action 0, 1, 2, 3 for up, down , left, right
3
Move #: 1; Computer Taking action: 3
Move #: 1; Player Taking action: 3
[[u' ' u' ' u' ' u'P']
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u' ' u' ' u' ' u' ']]
Player Reward: 10
```
 We won in this case

In 700 epochs →
```
Initial State:
[[u' ' u'P' u' ' u' ']
```

```
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u' ' u'P2' u' ' u'+']]
Enter your action 0, 1, 2, 3 for up, down , left, right
3
Move #: 0; Computer Taking action: 2
Move #: 0; Player Taking action: 3
[[u'P' u' ' u' ' u' ']
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u' ' u' ' u'P2' u'+']]
Enter your action 0, 1, 2, 3 for up, down , left, right
3
Move #: 1; Computer Taking action: 2
Move #: 1; Player Taking action: 3
[[u'P' u' ' u' ' u' ']
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u' ' u' ' u' ' u' ']]
Player Reward: 10
```

For the 1000 epochs →

```
Initial State:
[[u' ' u'P' u' ' u' ']
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u' ' u'P2' u' ' u'+']]
Enter your action 0, 1, 2, 3 for up, down , left, right
2
Move #: 0; Computer Taking action: 2
Move #: 0; Player Taking action: 2
[[u'P' u' ' u' ' u' ']
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u'P2' u' ' u' ' u'+']]
Enter your action 0, 1, 2, 3 for up, down , left, right
3
Move #: 1; Computer Taking action: 2
Move #: 1; Player Taking action: 3
[[u'P' u' ' u' ' u' ']
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u' ' u'P2' u' ' u'+']]
```

```
Enter your action 0, 1, 2, 3 for up, down , left, right
3
Move #: 2; Computer Taking action: 3
Move #: 2; Player Taking action: 3
[[u' ' u'P' u' ' u' ']
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u' ' u' ' u'P2' u'+']]
Enter your action 0, 1, 2, 3 for up, down , left, right
3
Move #: 3; Computer Taking action: 3
Move #: 3; Player Taking action: 3
[[u' ' u' ' u'P' u' ']
 [u' ' u'-' u' ' u' ']
 [u' ' u' ' u'W' u' ']
 [u' ' u' ' u' ' u' ']]
Player Reward: 10
```

3) For the architecture of the neural network, we get the maximum of the Q values for every possible action in new state s’ where the function is maxQ(s’,a’)

```
model = Sequential()
model.add(Dense(164, init='lecun_uniform', input_shape=(80,)))
model.add(Activation('relu'))
model.add(Dense(150, init='lecun_uniform'))
model.add(Activation('relu'))
model.add(Dense(4, init='lecun_uniform'))
model.add(Activation('linear'))
rms = RMSprop()
model.compile(loss='mse', optimizer=rms)
model.predict(state.reshape(1,64), batch_size=1)
```

The input shape is 80, because the grid is initialized as as (4,4,5) 5 because we have 5 levels in this case for the 2 players and the wall and the pit and the goal.
As for the architecture, we added one input layer 80 units (4*4*5) and 2 hidden layers for 164 and 150 units and an output layer of 4 units for each of the possible actions (up,down,left,right)
The role of the neural network is to calculate the action for the current state and to select the action with the highest Q value. This is repeated until the game is either lost or won.
4) For the Learning of the neural network, we used the sequence as follows:
  1- We set up a for loop for the number of epochs, in our case we used 4 possibilities for the 4 interactive games (1000 epochs, 700 epochs, 400 epochs, 100 epochs) where every epoch is a full game played till it is complete.
  2- We start the game by setting status =1

3- We run the Q network on state (s) to get Q values for all possible actions → qval= model.predict(state.reshape(1,64), batch_size=1)
4- After time t, we choose the action associated with the highest Q value from our neural network → action = (np.argmax(qval))
5- Using the action obtained from step 4, we observe a new state → new_state = makeMove(state, action)
and a new reward(r(t+1)) → reward = getReward(new_state)
6- The target value to train the network is reward + (gamma * maxQ) where gamma is a value between 0 and 1 → update = (reward + (gamma * maxQ)).

How to play:
1- hardTrain(no of epochs)
2- testAlgoMulti(0 or 1 or 2) 0→ standard grid , 1 → random position for the player, 2 → random position for all
3- Enter your action 0, 1, 2, 3 for up, down , left, right respectively