AI Final Project

Train an agent to play『Text World』

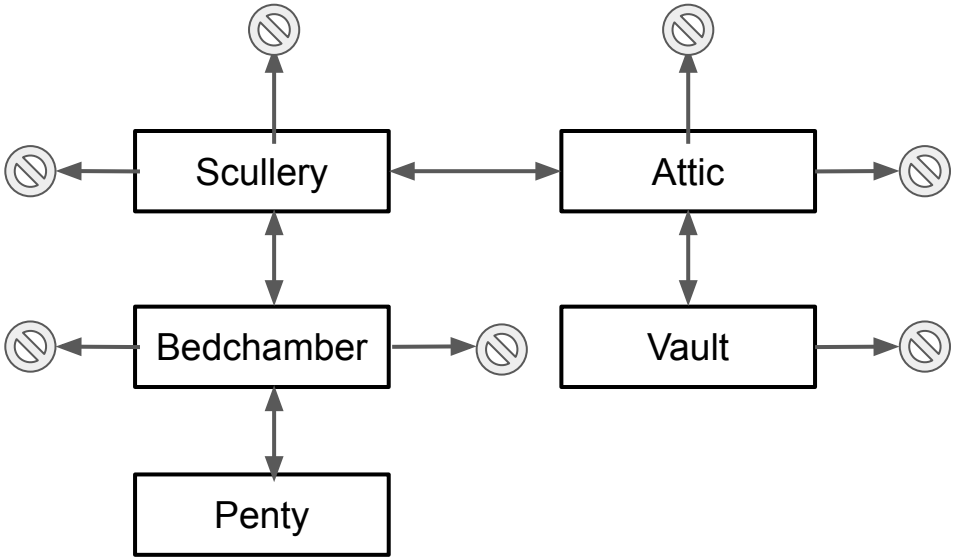
Alexis Lin
2019/1/2

環境說明

- Text World 是一個以Python開發的文字冒險類遊戲沙盒(開發環境)，包括遊戲創建和遊玩兩大功能。
- 以 Text World 創建一個遊戲
  - 參數：5個房間 / 10個物件 / 5個完成遊戲所需最少指令 (一個指令即為一個步數)
  - 指令：tw-make custom --world-size 5 --nb-objects 10 --quest-length 5 --seed 1234 --output gen_games/
- 撰寫一個Agent與Text World遊戲環境互動
  - game_state, reward, done = env.step(command)
- 遊戲環境提供函式讓agent取得環境資訊
  - env.compute_intermediate_reward()　　取得立即回饋分數
  - game_state.description　　取得目前所在房間的狀態的描述文字
  - game_state.admissible_commands　　取得目前狀態下可下的指令
  - game_state.state　　取得目前所在房間的位置資訊 (JSON格式)

互動觀察

- 互動方式
  - 以人工方式與遊戲互動
- 資訊取得
  - 遊戲環境僅會顯示部分資訊，沒有明確的遊戲目標指示。
- 房間配置
  - (如右圖)
- 完成遊戲
  - 在 Attic 中執行 insert the keycard into the toolbox 可完成遊戲



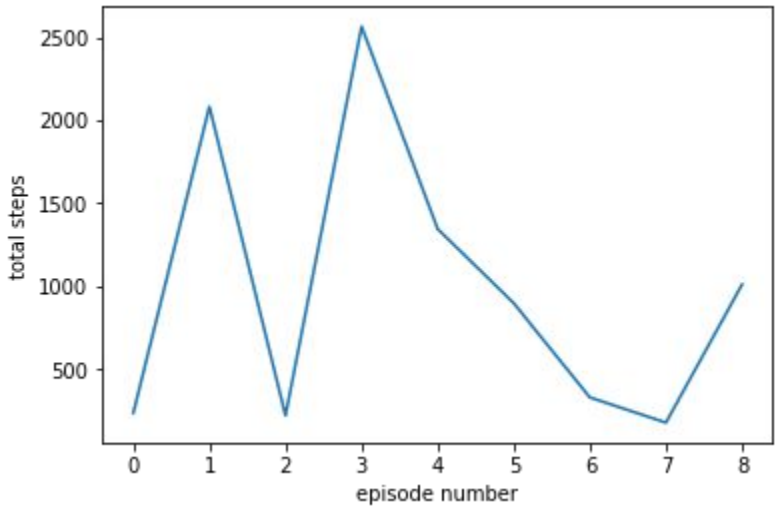
互動觀察

- 互動方式
  - 以程式與遊戲互動
- 資訊取得
  - 可取得明確的遊戲目標 (game_state.objective)
  - 可取得明確的遊戲環境的描述 (game_state.description)
  - 可查詢已取得的物品 (game_state.inventory)
- 立即回饋
  - 進入正確的房間(go east / go west / go south / go north)會得 +1 分
  - 進入不正確的房間(go east / go west / go south / go north)會得 -1 分
  - 其他都是 0分

Implement a Basic Agent

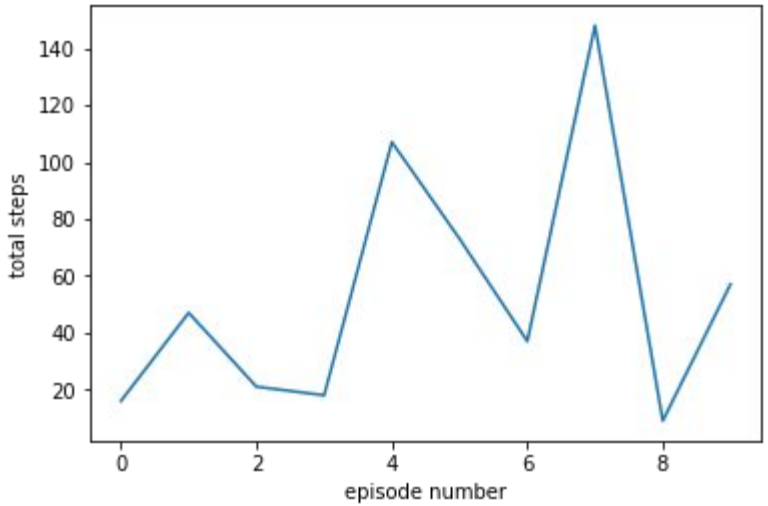
Basic Agent

- 實驗方式
  - 以所在房間作為state，共有5個值。
  - 以 game_state.admissible_commands 取得可執行的 command list
  - 從command list中**隨機取出**一個command代入env.step(command)函式執行
  - 10 episodes / 5000 steps in a episode
- 實驗結果
  - avg. steps: **1386.1**;
  - avg. score: 0.9 / 1.
  - 探索過110個actions



Basic Agent

- 實驗方式
  - 以所在房間作為state，共有5個值。
  - 以 game_state.admissible_commands 取得可執行的 command list
  - 篩選出**以 go, insert, take 開頭的動作**存放在 good_command_list
  - 從good_command_list 中**隨機取出**一個command代入 env.step(command)函式執行
  - 10 episodes / 1000 steps in a episode
- 實驗結果
  - avg. steps: **53.3**;
  - avg. score: 1.0 / 1.
  - 探索過28個actions



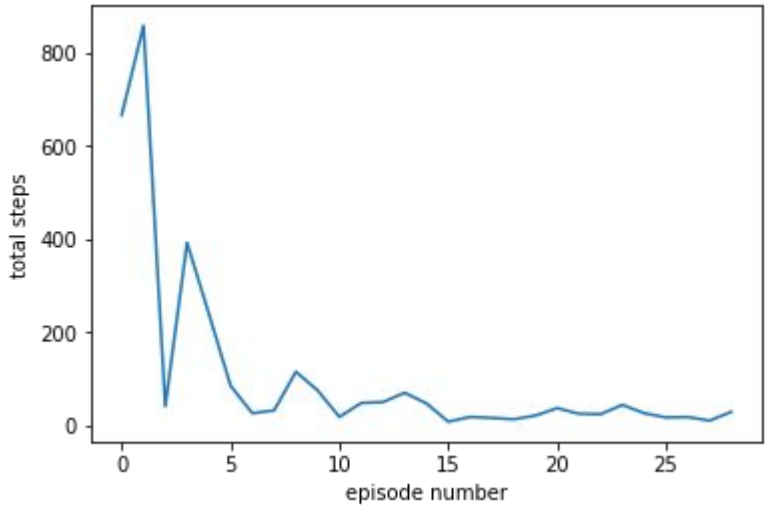
Implement a Q-Learning Agent

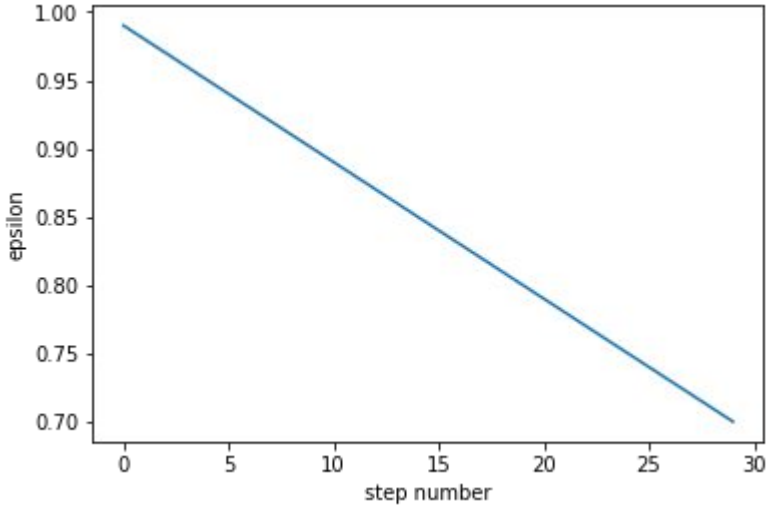
Q-Learning Agent

- 實驗方式
  - 以所在房間作為state，共有5個值。
  - 以 game_state.admissible_commands 取得可執行的 command list
  - 在每個state 可執行的command list內容不同，長度也不同
  - 以env.compute_intermediate_reward()取得立即回饋分數，將 (state,command,intermediate_reward) 存入Q table，並持續更新
  - 50%機率從command list 中隨機取出一個command，50%機率從 Q-table中取出最高分的動作代，代入 env.step(command)函式執行
  - 30 episodes / 1000 steps in a episode
  - **嘗試不同 epsilon**

Q-Learning Agent

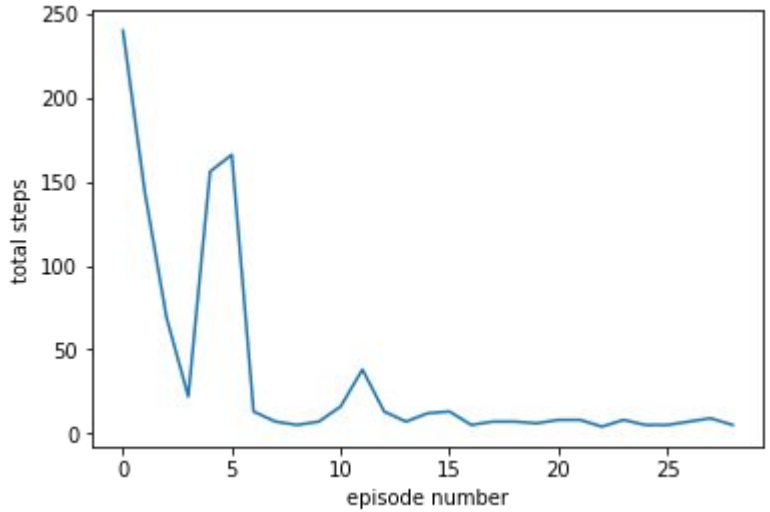
- 實驗結果1
  - epsilon: episode - 0.01
  - avg. steps: 137.6;
  - avg. score: 1.0 / 1.

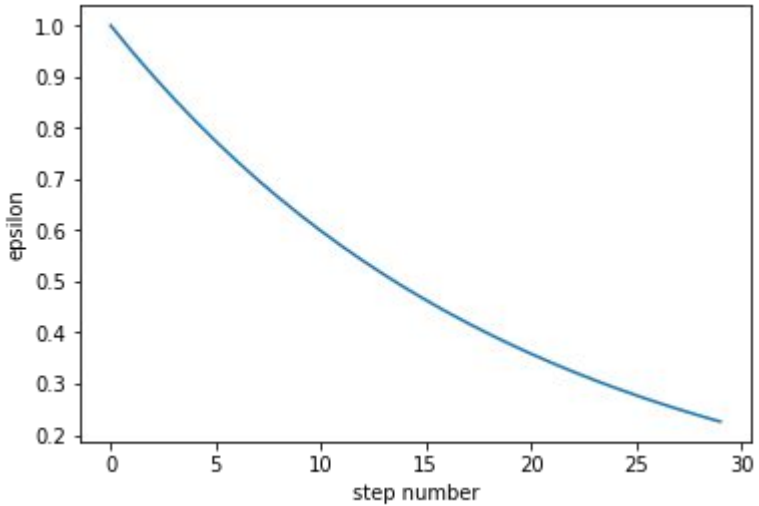




Q-Learning Agent

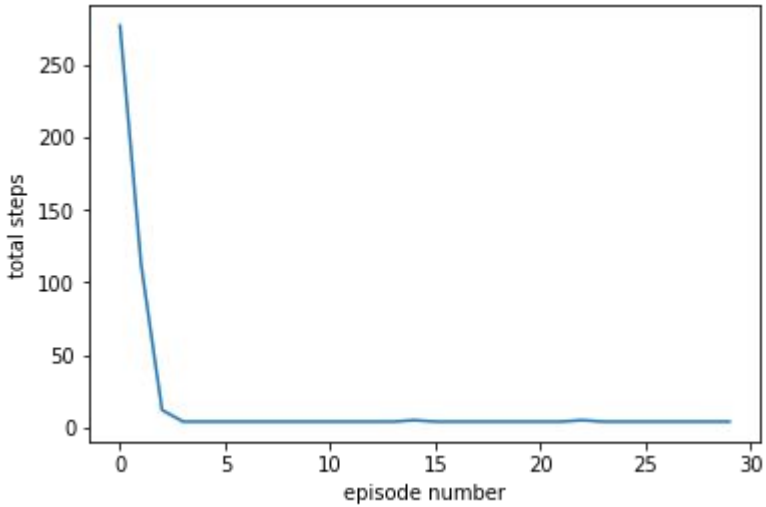
- 實驗結果2
  - epsilon: math.pow(0.95,no_episode)
  - avg. steps: 68.0;
  - avg. score: 1.0 / 1.

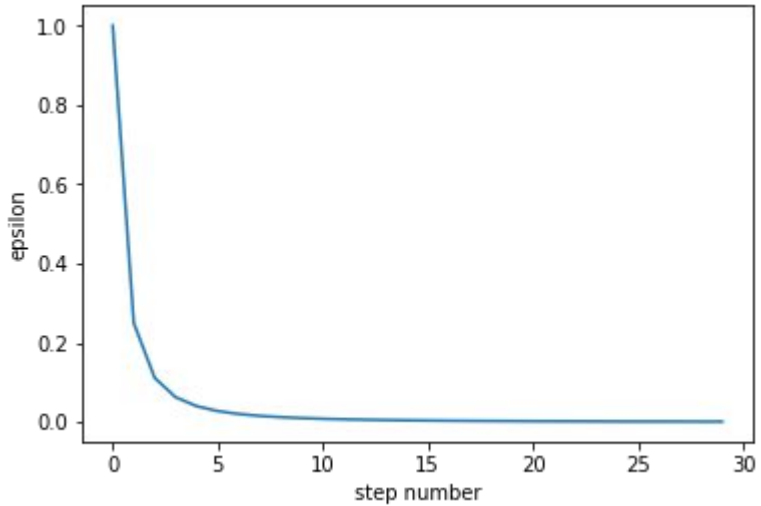




Q-Learning Agent

- 實驗結果3
  - epsilon: 1/math.pow(no_episode+1,2)
  - avg. steps: 18.1; **收斂於 5 steps**
  - avg. score: 1.0 / 1.





Implement a DQN Agent

DQN Agent

- 實驗方式
  - 以所在房間作為state，共有5個值，並以one hot encoding方式編碼。
  - 以 game_state.admissible_commands 取得可執行的 command list
  - 篩選出以go,insert,take開頭的動作(共28個動作)，並以one hot encoding 方式編碼
  - 以env.compute_intermediate_reward()取得立即回饋分數，並將 (state,command,intermediate_reward) 資訊記錄起來，作為訓練模型的資料集。
  - 以DQN訓練模型
  - 10 episodes / 1000 steps in a episode

DQN Agent

- Parameters
  - gamma = 0.95
  - epsilon = 1.0
  - epsilon_min = 0.001
  - **epsilon_decay = 0.995**
  - **learning_rate = 0.001**

DQN Agent

- Model
  - model.add(Dense(24, input_dim=self.state_size, activation='relu'))
  - model.add(Dense(24, activation='relu'))
  - model.add(Dense(self.action_size, activation='linear'))
  - model.compile(loss='mse',optimizer=Adam(lr=self.learning_rate))
- Parameters
  - gamma = 0.95
  - epsilon = 1.0
  - epsilon_min = 0.001
  - **epsilon_decay = 0.9**
  - **learning_rate = 0.5**

```
======COMPLETED======
episode: 1/10, score: 245, e: 0.00096
======COMPLETED======
episode: 2/10, score: 972, e: 0.00096
======COMPLETED======
episode: 3/10, score: 449, e: 0.00096
======COMPLETED======
episode: 4/10, score: 548, e: 0.00096
======COMPLETED======
episode: 5/10, score: 213, e: 0.00096
======COMPLETED======
episode: 6/10, score: 761, e: 0.00096
======COMPLETED======
episode: 7/10, score: 360, e: 0.00096
======COMPLETED======
episode: 8/10, score: 654, e: 0.00096
======COMPLETED======
episode: 9/10, score: 148, e: 0.00096
avg. steps: 535.9; avg. score:  0.9 / 1.
```

DQN效果不佳的可能問題

- state只有所在room, feature太少, learning時容易卡住
- intermediate_reward 只有在go開頭的command才有正負值, 其他都是0, 容易卡住

參考資料

https://github.com/microsoft/textworld

https://textworld.readthedocs.io/en/latest/