Readme

1 Supplementary Material Directory Structure

The supplementary material accompanying this paper is organized as follows:

1. Inside the `supplementary` folder, you will find the following items:
   (a) One txt file:
      - `Config.txt`: Contains versions for all libraries and packages.

   (b) Two PDF files:
      - `ReadMe.pdf`: Contains general instructions and information about the supplementary material.
      - `Environment-Description.pdf`: Provides detailed descriptions of the environments used in the study, including definitions of state, action, reward, and the goal of the agent for each environment. Additionally, it includes descriptions of the best episodes obtained for each environment, as well as tabular results for each seed.

   (c) Three folders, each corresponding to one of the three environments:
      - `Mountain Car`: Contains subfolders for plots, code, results, and seeds used for all codes related to the Mountain Car environment.
      - `Grid World`: Contains subfolders for plots, code, results, and seeds used for all codes related to the Grid World environment. Additionally, there is a subfolder named `Ablation` specifically for ablation codes, plots, and results related to the Grid World environment.
      - `Lunar Lander`: Contains subfolders for plots, code, results, and seeds used for all codes related to the Lunar Lander environment.

2 Procedure to Run Code and Obtain Results and Plots

2.1 Running Environment

The code was executed in the Google Colab environment version 0.0.1a2 . Versions for all libraries and packages are provided in `Config.txt` file.

2.2 Running the Code

2.2.1 Running the code for Convergence Results and Plots for Returns

The procedure for running the code and obtaining convergence results is similar for each environment. We illustrate the process using the Mountain Car environment as an example:

1. Navigate to the Mountain Car folder.
2. Access the `code` subfolder, where you will find four Jupyter Notebook files: `DQN.ipynb`, `RYB-Exp.ipynb`, `Seed.ipynb`, and `PlotReturns.ipynb`.
3. In the `Seed.ipynb` notebook, `seeds` array consists of 10 randomly generated seeds.
4. For each selected seed:
   (a) Set the seed variable value in both `DQN.ipynb` and `RYB-Exp.ipynb` to the chosen seed from `Seed.ipynb`.
   (b) Execute both `DQN.ipynb` and `RYB-Exp.ipynb` notebooks.
   (c) After running for all seeds, 20 .txt files will be saved in the folder section of the Google Colab notebook: 10 in the `DQN.ipynb` folder section and 10 in the `RYB-Exp.ipynb` folder section. Each of these .txt files contains the total returns for each episode of DQN or RYB-Exp. The naming convention is as follows:
      - `219_NN.txt`: RYB-DQN returns for seed 219.
      - `219_PN.txt`: DQN returns for seed 219.
5. Upload all 20 .txt files to the `PlotReturns.ipynb` notebook. For your assistance all 20.txt files are given in `Results` folder of `Mountain Car`
6. Inside `PlotReturns.ipynb`, execute the code blocks to obtain convergence results and plots for returns:
   - The `results` array stores episode numbers at which DQN converged for each of the 10 seeds, while the `results2` array stores episode numbers at which RYB-Exp converged.
   - To calculate the performance gain, use the code block: `performance_gain = sum(results)/len(results) - sum(results2)/len(results2)`.
   - To plot returns for DQN and RYB-Exp for a given seed, use the following code block:

     ```
     plot_moving_averages(lists[index], lists2[index], window_size=10)
     ```

     and replace `index` with the corresponding index from the seed array in `Seed.ipynb`.

- For clarity, let's consider an example using the seeds array provided in `Seed.ipynb`:

  ```
  seeds = [219, 4065, 987, 434, 4218, 846, 2647, 4283, 1190, 4372]
  ```

  Suppose we want to plot returns for seed number 434. In this case, we look at the seeds array and find that the corresponding index is 3. We then use this index in the code block by setting the value to 3 for both the `lists` and `lists2` indices:

  ```
  plot_moving_averages(lists[3], lists2[3], window_size=10)
  ```

7. Ten plots corresponding to ten different seed numbers are provided in the `Plots` folder of the `Mountain Car` folder. Each plot is named after its respective seed number.

8. Repeat the same process for `Grid World` and `Lunar Lander` Environments.

2.2.2 Running the code for Ablation Analysis 1 (Percentage of Good Episodes)

1. Navigate to the Ablation folder inside the Grid World directory.

2. Within the Ablation folder, locate three subfolders: Code, Plots, and Results.

3. Access the Code subfolder, where you will find 5 .ipynb files: `DQN(Episode%).ipynb`, `Plot(Memory%).ipynb`, `RYB-Exp(Episode%).ipynb`, `RYB-End.ipynb` and `Plot(RYB-End).ipynb`

4. Run both `DQN(Episode%).ipynb` and `RYB-Exp(Episode%).ipynb` files.

5. Upon completion, a .txt file will be downloaded in the files folder of the Colab notebook for each of the two notebooks: `DQN(Episode%).ipynb` and `RYB-Exp(Episode%).ipynb`.

6. The .txt files corresponding to the two notebooks will be named `719_NN(per).txt` and `719_PN(per).txt`. These files are present in `Results` subfolder of `Ablation` folder.

7. These files contain the Percentage of Good Episodes in Replay Memory for each episode number.

8. Once these files are obtained, upload them to `Plot(Memory%).ipynb`.

9. Run the code in `Plot(Memory%).ipynb`, to obtain plot for Percentage of good episodes (Ablation Analysis 1). This plot is provided in `Plots` subfolder of `Ablation` folder as `Memory.png`.

2.2.3 Running the code for Ablation Analysis 2 (RYB Period Till End)

To generate the plot for Ablation Analysis 2 (RYB-Period Till End), follow these steps:

1. Navigate to the `Ablation` folder inside the `Grid World` Folder.
2. Enter the `Code` folder.
3. Open the `RYB-End.ipynb` notebook.
4. Execute the code in the notebook. At the end, a txt file named `719_NN(E).txt` will be downloaded inside the files section of the Colab Notebook.
5. File `719_NN(E).txt` contains returns for every episode when RYB period never ends.
6. Upload this `719_NN(E).txt` file along with two other txt files, `719_NN.txt` and `719_PN.txt`, to the `Plot(RYB-End).ipynb` notebook which is present in Code folder inside Ablation folder.
7. Files `719_NN(E).txt`,`719_NN.txt` and `719_PN.txt` are present in `Results` folder inside `Ablation` folder
8. Run the code in the `Plot(RYB-End).ipynb` notebook to obtain the plot for Ablation Analysis 2.
9. Plot obtained from step 8 is available in `Plots` folder inside `Ablation` Folder as `RYB-End.png`.

Note: The files `719_NN.txt` and `719_PN.txt` were obtained using the procedure outlined in section 2.2.1.

3 Brief Analysis of Ablation Codes

3.1 Explanation of Codes for Percentage of Good Episodes

1. **Initialization:** We begin by running the RYB-Exp algorithm twice. During the first run, we analyze each transition to determine its quality – whether it is considered good or bad. This analysis is based on a precalculated mapping that associates each state with the optimal action to be taken.
2. **Storage of Transition Quality:** After the first run, we have a record of quality of transitions corresponding to every time step of every episode.

3. **Second Run with Flagged Transitions:** In the second run of the algorithm, we utilize the information obtained from the first run , we repeat the process with the same initial setup as the first run. This includes setting the seed back to 719, ensuring that all states are reinitialized exactly as they were in the first run. Consequently, all transitions during the second run match those of the first run.

   During this second run, we enhance the replay memory buffer by adding an extra flag variable to each transition. This flag indicates whether the transition is good or bad based on the classification from the first run.

4. **Calculation of Good Episode Percentage:** After every episode during the second run, we traverse the entire replay buffer. For each transition, we check its flag to determine whether it is classified as good or bad. By tallying the number of good transitions and dividing it by the total number of transitions, we calculate the percentage of good transitions for that episode.

5. **Recording Results:** The calculated percentage of good transitions for each episode is then appended to a .txt file.