Augmented Random Search (ARS) with BBRL

Joe CHAMOUN
Jawher Jriby

1 Project Structure

- **ARS_BBRL/**:
  - **ARS_BBRL.ipynb**: Jupyter notebook containing an example implementation and demonstration of the ARS algorithm using BBRL.
  - **ARSAgent.py**: Base class definition for the ARS agent.
  - **ARSAgent_v1.py**: Implementation of ARS agent version 1, inheriting from `ARSAgent`.
  - **ARSAgent_v2.py**: Implementation of ARS agent version 2, inheriting from `ARSAgent`.
  - **config.yaml**: YAML configuration file specifying hyperparameters and settings for the ARS algorithm.
  - **Logger.py**: Logger class for recording and logging experiment data.
  - **run_ars.py**: Main script for running the ARS algorithm and evaluating its performance.
  - **Videos/**: Contains rendered videos from the ARS algorithm executions.
- **Cahier des charges.pdf**: Project requirements document (in PDF format).
- **Documentation.pdf**: Detailed project documentation covering implementation details, usage instructions, and additional resources.
- **OLD_Notebooks/**:
  - **CHAMOUN_JRIBY_ARS_BBRL.ipynb**: Implementation of the ARS algorithm using BBRL.
  - **CHAMOUN_JRIBY_ARS.ipynb**: Implementation of the ARS algorithm without using BBRL.
  - **CHAMOUN_JRIBY_BRS.ipynb**: Implementation of a related algorithm (BRS: Basic Random Search) without using BBRL.
- **Rapport.pdf**: Analytical report comparing the performance and results of ARS with Cross-Entropy Method (CEM).
- **README.md**: Documentation file providing an overview of the project and instructions for usage.

2 Usage

2.1 Running Locally

To run the ARS algorithm locally:

1. Ensure you have Python 3.x installed on your system.
2. Install the required dependencies and register the necessary configurations using the provided code snippet below:

```
# Install easypip and required Python packages
pip install easypip>=1.2.0
easypip install "bbrl>=0.2.2" "gymnasium" "tensorboard"
"bbrl_gymnasium>=0.2.0" "bbrl_gymnasium[box2d]" "mazemdp"

# Install additional system-level dependencies
sudo apt-get install -y python3-dev swig
pip install box2d-py
```

3. Modify the `config.yaml` file to adjust hyperparameters and settings as needed.
4. Execute the `run_ars.py` script in your terminal or preferred Python environment to start the ARS algorithm with the specified configuration.

2.2 Running in Google Colab

To run the ARS algorithm in Google Colab:

1. Open Google Colab in your web browser.
2. Create a new notebook or open an existing one.
3. Upload the `ARS_BBRL.ipynb` notebook file to your Google Colab session:
   - Click on the "Files" tab in the left sidebar of the Colab interface.
   - Click on the "Upload" button and select `ARS_BBRL.ipynb` from your local machine.
4. Upload the remaining files (`ARSAgent.py`, `ARSAgent_v1.py`, `ARSAgent_v2.py`, `config.yaml`, `Logger.py`, `run_ars.py`) directly within the Colab notebook environment.
5. Execute the code cells within `ARS_BBRL.ipynb` to run the ARS algorithm. Ensure that you have the required dependencies installed within your Colab environment.

The `ARS_BBRL.ipynb` notebook provides an interactive environment to experiment with and visualize the ARS algorithm. Users can modify hyperparameters, settings, or experiment configurations directly within the notebook.

3 Requirements

- Python 3.x
- Jupyter Notebook (for local execution)
- Google Colab (for cloud-based execution)
- Required dependencies are imported and managed within the `ARS_BBRL.ipynb` notebook or can be installed as needed.

4 Additional Resources

For further information and resources related to ARS and reinforcement learning:

- BBRL Documentation
- OpenAI Gym Documentation
- TensorFlow Documentation
- Matplotlib Documentation
- NumPy Documentation
- MoviePy Documentation
- MazeMDP Documentation

For detailed information on specific functions and classes used within the ARS algorithm, refer to the source code files (`ARSAgent.py`, `ARSAgent_v1.py`, `ARSAgent_v2.py`, `run_ars.py`, etc.) in the repository.

5 Authors

Joe CHAMOUN
Jawher JRIBY