Master Thesis

Deep Reinforcement Learning for Autonomous Train Operation

Abstract

In this thesis, an investigation into the application of deep reinforcement learning to optimize train operation with a focus on achieving an intelligent train driver that adheres to speed limits for safety, ensures passenger comfort, maintains punctuality, exhibits accurate parking, and promotes energy efficiency is done. Two prominent DRL algorithms, Double Deep Q-Network and proximal Policy Optimization are explored, to develop an intelligent agent capable of addressing these objectives. This research includes an analysis of the performance of both algorithms in the context of the train operation optimization problem. While the Double Deep Q-Network algorithm demonstrates certain merits in other fields, it fails to converge to an optimal solution that fulfills all the objectives outlined in the thesis. In contrast, the Proximal Policy Optimization algorithm successfully learns an optimal policy that comprehensively addresses the stated goals, showcasing its potential for efficient train operation optimization. The findings provide valuable insights for further research in the development of intelligent and sustainable train operation systems that prioritize safety, comfort, punctuality, parking accuracy, and energy efficiency.

Installing Program

Step 1: Get Repository Clone this repository to your local repository.

```
git clone https://github.com/joelmwaka/Master-Thesis.git
```

You can also just download the compressed .zip folder and extract it in your own local repository.

In the folder, you will find an Implementation folder that contains the code of the program, and a Thesis folder that contains the written thesis in Latex.

Step 2: Create Virtual Environment and Install Libraries Dependencies from requirements.txt

1. Install `Python 3.10`. Program may work with any other python 3 version, but this isn't guaranteed.
2. Enter into `.\Master-Thesis\` folder: `cd .\Master-Thesis\`
3. Create a virtual environment: `python.exe -m venv .\venv`
4. Activate virtual environment: `.\venv\Scripts\activate`
5. Enter into `.\Implementation\` folder: `cd .\Implementation\`

6. If not yet done, install/update pip. Version used: 21.2.3: `python.exe -m pip install pip==21.2.3`
7. Install necessary dependencies from the `requirements.text` file: `pip install -r .\requirements.txt`
8. Install custom train environment: `cd .\gym-train\` `pip install -e .`
9. Return to `.\Implementation\` folder: `cd ..`

On some machines, the PyTorch libraries (torch, torchaudio and torchvision) may throw an error during installation of the dependencies from `requirements.txt` and need a different installation command. On the PyTorch website, you can select the desired config from the installation matrix, and run the created command in your terminal.

Running Program from Terminal

Navigate to `..\Master-Thesis\Implementation\`

Run `python.exe .\main.py [START UP OPTIONS]`

The following start up options are available.

Argument, Short	Description	Expected Values
–help, -h	Provides some help by showing all possible arguments one can call the `main.py` script with.	-
–algo, -a	Which algorithm to use, either in training or evaluation. Should always be provided, even during evaluation.	`DDQN`, `PPO`
–train, -t	Activate agent training mode. If not added as an option, Evaluation mode is activated.	-
–training_path, -tp	Path to a location where training results will be saved.	`\path\to\training\folder\`
–evaluation_path, -ep	Path to location with agent that should be evaluated.	`\path\to\training\folder\`

Argument, Short	Description	Expected Values
–agent_name, -an	Name of the agent to be evaluated. Do not include `.zip` or `.pth`. The agent should be available in the evaluation folder provided.	Agent Name: `str`
–journey_mode, -jm	Define how the journey should be chosen for evaluation.	`0`: Random Journey, `1`: Journey from `.\files\journey.xml`, `2`: Journey by ID
–journey_id, -id	If journey mode is `2`, then the journey ID must be selected. The ID corresponds to the entry of the journey in the Excel File `.\files\journeys.xlsx`	`int`
–render, -r	Render train environment while evaluating performance.	-
–update_journeys, -uj	Recalculate journey data using journey time distribution algorithm and save to `.\files\journeys.xlsx`.	-

Examples

Train an agent using PPO algorithm

1. Adjust the Hyperparameters of the PPO algorithm in the `main.py` file.
2. Create a new folder in which you want to save your training information: `mkdir .\models\PPO_Agents\'folder_name'`
3. Start training the agent: `python.exe .\main.py -a PPO -t -tp .\models\PPO_Agents\'folder_name'`
4. During training, the learning progress of the PPO algorithm (with Stable Baselines 3) can be monitored with Tensorboard. In a new terminal, run: `tensorboard.exe --log_dir .\models\PPO_Agents`
5. To view the progress, open your Tensorboard in your browser at `http://localhost:'port/`. The `'port'` is given in the terminal where you called it.

Train an agent using DDQN algorithm

1. Adjust the Hyperparameters of the DDQN algorithm in the `main.py` file.
2. Create a new folder in which you want to save your training information: `mkdir .\models\DDQN_Agents\'folder_name'`
3. Start training the agent: `python.exe .\main.py -a DDQN -t -tp .\models\DDQN_Agents\'folder_name'`
4. Unlike the PPO algorithm implemented with Stable Baselines 3, the manually implemented DDQN algorithm doesn't offer Tensorboard progress tracking. Instead, the learning metrics are saved to a .csv file `.\models\DDQN_Agents\'folder_name'\logging.csv` from which learning progress can be plotted.

The hyperparameters of the training algorithm are saved in the training folder as a .txt file.

Evaluating a PPO Agent Run this command for evaluation on a random journey: `python.exe .\main.py -a PPO -ep .\models\PPO_Agents\'folder_name' -an 'agent_name' -jm 0'`

Evaluating a DDQN Agent Run this command for evaluation on a random journey: `python.exe .\main.py -a DDQN -ep .\models\DDQN_Agents\'folder_name' -an 'agent_name' -jm 0'`

Contact

In case of any questions or issues regarding this project, please feel free to contact me via e-mail: `joel.mwaka@outlook.com`