TextWorld

A text-based game generator and extensible sandbox learning environment for training and testing reinforcement learning (RL) agents. Also check out aka.ms/textworld for more info about TextWorld and its creators. Have questions or feedback about TextWorld? Send them to textworld@microsoft.com or use the Gitter channel listed above.

Installation

TextWorld requires **Python 3** and only supports **Linux** and **macOS** systems at the moment. For **Windows** users, docker can be used as a workaround (see Docker section below).

Requirements

TextWorld requires some system libraries for its native components.
On a Debian/Ubuntu-based system, these can be installed with

```
sudo apt update && sudo apt install build-essential libffi-dev python3-dev curl
git
```

And on macOS, with

```
brew install libffi curl git
```

> **Note:** We advise our users to use virtual environments to avoid Python packages from different projects to interfere with each other. Popular choices are Conda Environments and Virtualenv

Installing TextWorld

The easiest way to install TextWorld is via `pip`:

```
pip install textworld
```

Or, after cloning the repo, go inside the root folder of the project (i.e. alongside `setup.py`) and run

```
pip install .
```

Visualization

TextWorld comes with some tools to visualize game states. Make sure all dependencies are installed by running

```
pip install textworld[vis]
```

Then, you will need to install either the Chrome or Firefox webdriver (depending on which browser you have currently installed).
If you have Chrome already installed you can use the following command to install chromedriver

```
pip install chromedriver_installer
```

Current visualization tools include: `take_screenshot`, `visualize` and `show_graph` from `textworld.render`.

Docker

A docker container with the latest TextWorld release is available on DockerHub.

```
docker pull marccote19/textworld
docker run -p 8888:8888 -it --rm marccote19/textworld
```

Then, in your browser, navigate to the Jupyter notebook's link displayed in your terminal. The link should look like this

```
http://127.0.0.1:8888/?token=8d7aaa...e95
```

> **Note:** See README.md in the docker folder for troubleshooting information.

Usage

Generating a game

TextWorld provides an easy way of generating simple text-based games via the `tw-make` script. For instance,

```
tw-make custom --world-size 5 --nb-objects 10 --quest-length 5 --seed 1234 --
output tw_games/custom_game.z8
```

where `custom` indicates we want to customize the game using the following options: `--world-size` controls the number of rooms in the world, `--nb-objects` controls the number of objects that can be interacted with (excluding doors) and `--quest-length` controls the minimum number of commands that is required to type in order to win the game. Once done, the game `custom_game.z8` will be saved in the `tw_games/` folder.

Playing a game (terminal)

To play a game, one can use the `tw-play` script. For instance, the command to play the game generated in the previous section would be

```
tw-play tw_games/custom_game.z8
```

> **Note:** Only Z-machine's games (*.z1 through *.z8) and Glulx's games (*.ulx) are supported.

To visualize the game state while playing, use the `--viewer [port]` option.

```
tw-play tw_games/custom_game.z8 --viewer
```

A new browser tab should open and track your progress in the game.

Playing a game (Python + Gym)

Here's how you can interact with a text-based game from within Python using OpenAI's Gym framework.

```
import gym
import textworld.gym

# Register a text-based game as a new Gym's environment.
env_id = textworld.gym.register_game("tw_games/custom_game.z8",
                                     max_episode_steps=50)

env = gym.make(env_id)  # Start the environment.

obs, infos = env.reset()  # Start new episode.
env.render()

score, moves, done = 0, 0, False
while not done:
    command = input("> ")
    obs, score, done, infos = env.step(command)
    env.render()
    moves += 1

env.close()
print("moves: {}; score: {}".format(moves, score))
```

> **Note:** To play text-based games without Gym, see Playing text-based games with TextWorld.ipynb

Documentation

For more information about TextWorld, check the documentation.

Visual Studio Code

You can install the textworld-vscode extension that enables syntax highlighting for editing `.twl` and `.twg` TextWorld files.

Notebooks

Check the notebooks provided with the framework to see what you can do with it. You will need the Jupyter Notebook to run them. You can install it with

```
pip install jupyter
```

Citing TextWorld

If you use TextWorld, please cite the following BibTex:

```
@Article{cote18textworld,
  author = {Marc-Alexandre C\^ot\'e and
            \'Akos K\'ad\'ar and
```

```
            Xingdi Yuan and
            Ben Kybartas and
            Tavian Barnes and
            Emery Fine and
            James Moore and
            Ruo Yu Tao and
            Matthew Hausknecht and
            Layla El Asri and
            Mahmoud Adada and
            Wendy Tay and
            Adam Trischler},
  title = {TextWorld: A Learning Environment for Text-based Games},
  journal = {CoRR},
  volume = {abs/1806.11532},
  year = {2018}
}
```

Contributing

This project welcomes contributions and suggestions. Most contributions require you to agree to a
Contributor License Agreement (CLA) declaring that you have the right to, and actually do, grant us
the rights to use your contribution. For details, visit https://cla.microsoft.com.

When you submit a pull request, a CLA-bot will automatically determine whether you need to provide
a CLA and decorate the PR appropriately (e.g., label, comment). Simply follow the instructions
provided by the bot. You will only need to do this once across all repos using our CLA.

This project has adopted the Microsoft Open Source Code of Conduct.
For more information see the Code of Conduct FAQ or
contact opencode@microsoft.com with any additional questions or comments.