RLzoo

Release 1.0.3

May 21, 2020

User Guide

RLzoo is a collection of the most practical reinforcement learning algorithms, frameworks and applications, released on Github in November 2019. It is implemented with Tensorflow 2.0 and API of neural network layers in TensorLayer 2, to provide a hands-on fast-developing approach for reinforcement learning practices and benchmarks. It supports basic toy-test environments like OpenAI Gym and DeepMind Control Suite with very simple configurations. Moreover, RLzoo supports robot learning benchmark environment RLBench based on Vrep/Pyrep simulator. Other large-scale distributed training framework for more realistic scenarios with Unity 3D, Mujoco, Bullet Physics, etc, will be supported in the future.

We also provide novices friendly DRL Tutorials for algorithms implementation, where each algorithm is implemented in an individual script. The tutorials serve as code examples for our Springer textbook Deep Reinforcement Learning: Fundamentals, Research and Applications , you can get the free PDF if your institute has Springer license.

CHAPTER 1

Installation

RLzoo generally requires Python>=3.5. Also if you want to use DeepMind Control Suite environment, Python 3.6 will be required.

Direct installation:

```
pip3 install rlzoo --upgrade
```

Install from the source code on github:

```
git clone https://github.com/tensorlayer/RLzoo.git
cd RLzoo
pip3 install .
```

CHAPTER 2

Quick Start

2.1 Simple Usage

Open ./run_rlzoo.py:

```
from rlzoo.common.env_wrappers import build_env
from rlzoo.common.utils import call_default_params
from rlzoo.algorithms import TD3
# choose an algorithm
AlgName = 'TD3'
# select a corresponding environment type
EnvType = 'classic_control'
# chose an environment
EnvName = 'Pendulum-v0'
# build an environment with wrappers
env = build_env(EnvName, EnvType)
# call default parameters for the algorithm and learning process
alg_params, learn_params = call_default_params(env, EnvType, AlgName)
# instantiate the algorithm
alg = eval(AlgName+'(**alg_params)')
# start the training
alg.learn(env=env, mode='train', render=False, **learn_params)
# test after training
alg.learn(env=env, mode='test', render=True, **learn_params)
```

Run the example:

```
python run_rlzoo.py
```

Choices for `AlgName`: 'DQN', 'AC', 'A3C', 'DDPG', 'TD3', 'SAC', 'PG', 'TRPO', 'PPO', 'DPPO'

Choices for `EnvType`: 'atari', 'box2d', 'classic_control', 'mujoco', 'robotics', 'dm_control', 'rlbench'

Choices for `EnvName` refers to *List of Supported Environments in RLzoo*

2.2 Another Usage

For providing more flexibility, we provide another usage example of RLzoo with more explicit configurations as follows, where the users can pass in customized networks and otpimizers, etc.

```
import gym
from rlzoo.common.utils import make_env, set_seed
from rlzoo.algorithms import AC
from rlzoo.common.value_networks import ValueNetwork
from rlzoo.common.policy_networks import StochasticPolicyNetwork

''' load environment '''
env = gym.make('CartPole-v0').unwrapped
obs_space = env.observation_space
act_space = env.action_space
# reproducible
seed = 2
set_seed(seed, env)

''' build networks for the algorithm '''
num_hidden_layer = 4 #number of hidden layers for the networks
hidden_dim = 64 # dimension of hidden layers for the networks
with tf.name_scope('AC'):
        with tf.name_scope('Critic'):
                # choose the critic network, can be replaced with customized network
                critic = ValueNetwork(obs_space, hidden_dim_list=num_hidden_layer *
↪[hidden_dim])
        with tf.name_scope('Actor'):
                # choose the actor network, can be replaced with customized network
                actor = StochasticPolicyNetwork(obs_space, act_space, hidden_dim_
↪list=num_hidden_layer * [hidden_dim], output_activation=tf.nn.tanh)
net_list = [actor, critic] # list of the networks

''' choose optimizers '''
a_lr, c_lr = 1e-4, 1e-2  # a_lr: learning rate of the actor; c_lr: learning rate of
↪the critic
a_optimizer = tf.optimizers.Adam(a_lr)
c_optimizer = tf.optimizers.Adam(c_lr)
optimizers_list=[a_optimizer, c_optimizer]  # list of optimizers

# intialize the algorithm model, with algorithm parameters passed in
model = AC(net_list, optimizers_list)
'''
full list of arguments for the algorithm
----------------------------------------
net_list: a list of networks (value and policy) used in the algorithm, from common
↪functions or customization
optimizers_list: a list of optimizers for all networks and differentiable variables
gamma: discounted factor of reward
action_range: scale of action values
'''

# start the training process, with learning parameters passed in
model.learn(env, train_episodes=500,  max_steps=200,
            save_interval=50, mode='train', render=False)
'''
full list of parameters for training
```

(continues on next page)

(continued from previous page)

```
-------------------------------------
env: learning environment
train_episodes:  total number of episodes for training
test_episodes:  total number of episodes for testing
max_steps:  maximum number of steps for one episode
save_interval: time steps for saving the weights and plotting the results
mode: 'train' or 'test'
render:  if true, visualize the environment
'''

# test after training
model.learn(env, test_episodes=100, max_steps=200,  mode='test', render=True)
```

2.3 Interactive Configurations

We also provide an interactive learning configuration with Jupyter Notebook and *ipywidgets*, where you can select the algorithm, environment, and general learning settings with simple clicking on dropdown lists and sliders! A video demonstrating the usage is as following. The interactive mode can be used with rlzoo/interactive/main.ipynb by running `$ jupyter notebook` to open it.

CHAPTER 3

Configurations Overview

3.1 Supported DRL Agorithms

Generally RLzoo supports following DRL algorithms:

Value-based methods

- Deep Q-Networks (DQN)
- Double DQN
- Dueling DQN
- Prioritized Experience Replay (PER)
- Retrace
- Noisy DQN
- Distributed DQN

Policy-based methods

- Vanilla Policy Gradient (VPG)
- Trust Region Policy Optimization (TRPO)
- Proximal Policy Optimization (PPO)
- Distributed PPO (DPPO)

Actor-critic methods

- Actor-Critic (AC)
- Asychronous Advantage Actor-Critic (A3C)
- Deep Deterministic Policy Gradient (DDPG)
- Twin Delayed DDPG (TD3)
- Soft Actor-Critic (SAC)

3.2 Supported Environments

Generally RLzoo supports following environments for DRL:

- **OpenAI Gym**
  - Atari
  - Box2D
  - Classic Control
  - MuJoCo
  - Robotics
- DeepMind Control Suite
- RLBench

Full list of specific names of environments supported in RLzoo can be checked in *List of Supported Environments in RLzoo*.

3.3 Supported Configurations

Not all configurations (specific RL algorithm on specific environment) are supported in RLzoo, as in other libraries. The supported configurations for RL algorithms with corresponding environments in RLzoo are listed in the following table.

Algorithms	Action Space	Policy	Update	Envs
DQN (double, dueling, PER)	Discrete Only	NA	Off-policy	Atari, Classic Control
AC	Discrete/Continuous	Stochastic	On-policy	All
PG	Discrete/Continuous	Stochastic	On-policy	All
DDPG	Continuous	Deterministic	Off-policy	Classic Control, Box2D, MuJoCo, Robotics, DeepMind Control, RLBench
TD3	Continuous	Deterministic	Off-policy	Classic Control, Box2D, MuJoCo, Robotics, DeepMind Control, RLBench
SAC	Continuous	Stochastic	Off-policy	Classic Control, Box2D, MuJoCo, Robotics, DeepMind Control, RLBench
A3C	Discrete/Continuous	Stochastic	On-policy	Atari, Classic Control, Box2D, MuJoCo, Robotics, DeepMind Control
PPO	Discrete/Continuous	Stochastic	On-policy	All
DPPO	Discrete/Continuous	Stochastic	On-policy	Atari, Classic Control, Box2D, MuJoCo, Robotics, DeepMind Control
TRPO	Discrete/Continuous	Stochastic	On-policy	All

CHAPTER 4

API

4.1 make_env()

It can be used as:

```
env = build_env(EnvName, EnvType)
```

4.2 call_default_params()

It can be used as:

```
alg_params, learn_params = call_default_params(env, EnvType, AlgName)
```

The `call_default_params` returns the hyper-parameters stored in two dictionaries `alg_params` and `learn_params`, which can be printed to see what are contained inside. Hyper-parameters in these two dictionaries can also be changed by users before instantiating the agent and starting the learning process.

If you want to know exactly where the default hyper-parameters come from, they are stored in an individual Python script as `default.py` in each algorithm file in `./rlzoo/algorithms/`.

4.3 alg.learn()

It can be used as:

```
# start the training
alg.learn(env=env, mode='train', render=False, **learn_params)
# test after training
alg.learn(env=env, mode='test', render=True, **learn_params)
```

where the `alg` is an instantiation of DRL algorithm in RLzoo.

CHAPTER 5

DQN and Variants

5.1 Example

```
from rlzoo.common.env_wrappers import build_env
from rlzoo.common.utils import call_default_params
from rlzoo.algorithms import DQN

AlgName = 'DQN'
EnvName = 'PongNoFrameskip-v4'
EnvType = 'atari'

# EnvName = 'CartPole-v1'
# EnvType = 'classic_control'  # the name of env needs to match the type of env

env = build_env(EnvName, EnvType)
alg_params, learn_params = call_default_params(env, EnvType, AlgName)
alg = eval(AlgName+'(**alg_params)')
alg.learn(env=env, mode='train', **learn_params)
alg.learn(env=env, mode='test', render=True, **learn_params)
```

5.2 Deep Q-Networks

class rlzoo.algorithms.dqn.dqn.**DQN**(*net_list*, *optimizers_list*, *double_q*, *dueling*, *buffer_size*, *prioritized_replay*, *prioritized_alpha*, *prioritized_beta0*)

Papers:

Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540): 529.

Hessel M, Modayil J, Van Hasselt H, et al. Rainbow: Combining Improvements in Deep Reinforcement Learning[J]. 2017.

get_action(*obv*, *eps=0.2*)

get_action_greedy(*obv*)

learn(*env*, *mode='train'*, *render=False*, *train_episodes=1000*, *test_episodes=10*, *max_steps=200*, *save_interval=1000*, *gamma=0.99*, *exploration_rate=0.2*, *exploration_final_eps=0.01*, *target_network_update_freq=50*, *batch_size=32*, *train_freq=4*, *learning_starts=200*, *plot_func=None*)

Parameters

- **env** – learning environment
- **mode** – train or test
- **render** – render each step
- **train_episodes** – total number of episodes for training
- **test_episodes** – total number of episodes for testing
- **max_steps** – maximum number of steps for one episode
- **save_interval** – time steps for saving
- **gamma** – reward decay factor
- **(float)** (*exploration_final_eps*) – fraction of entire training period over which the exploration rate is annealed
- **(float)** – final value of random action probability
- **(int)** (*learning_starts*) – update the target network every *target_network_update_freq* steps
- **(int)** – size of a batched sampled from replay buffer for training
- **(int)** – update the model every *train_freq* steps
- **(int)** – how many steps of the model to collect transitions for before learning starts
- **plot_func** – additional function for interactive module

load_ckpt(*env_name*)
load trained weights :return: None

save_ckpt(*env_name*)
save trained weights :return: None

store_transition(*s*, *a*, *r*, *s_*, *d*)

sync()
Copy q network to target q network

update(*batch_size*, *gamma*)

5.3 Default Hyper-parameters

rlzoo.algorithms.dqn.default.**atari**(*env*, *default_seed=False*, ***kwargs*)

rlzoo.algorithms.dqn.default.**classic_control**(*env*, *default_seed=False*, ***kwargs*)

CHAPTER 6

VPG

6.1 Example

```python
from rlzoo.common.env_wrappers import build_env
from rlzoo.common.utils import call_default_params
from rlzoo.algorithms import PG

AlgName = 'PG'
EnvName = 'PongNoFrameskip-v4'
EnvType = 'atari'

# EnvName = 'CartPole-v0'
# EnvType = 'classic_control'

# EnvName = 'BipedalWalker-v2'
# EnvType = 'box2d'

# EnvName = 'Ant-v2'
# EnvType = 'mujoco'

# EnvName = 'FetchPush-v1'
# EnvType = 'robotics'

# EnvName = 'FishSwim-v0'
# EnvType = 'dm_control'

# EnvName = 'ReachTarget'
# EnvType = 'rlbench'

env = build_env(EnvName, EnvType)
alg_params, learn_params = call_default_params(env, EnvType, AlgName)
alg = eval(AlgName+'(**alg_params)')
alg.learn(env=env, mode='train', render=False, **learn_params)
alg.learn(env=env, mode='test', render=True, **learn_params)
```

6.2 Vanilla Policy Gradient

class `rlzoo.algorithms.pg.pg.`**PG**(*net_list*, *optimizers_list*)

PG class

get_action(*s*)

choose action with probabilities.

Parameters **s** – state

Returns act

get_action_greedy(*s*)

choose action with greedy policy

Parameters **s** – state

Returns act

learn(*env*, *train_episodes=200*, *test_episodes=100*, *max_steps=200*, *save_interval=100*, *mode='train'*, *render=False*, *gamma=0.95*, *plot_func=None*)

Parameters

- **env** – learning environment
- **train_episodes** – total number of episodes for training
- **test_episodes** – total number of episodes for testing
- **max_steps** – maximum number of steps for one episode
- **save_interval** – time steps for saving
- **mode** – train or test
- **render** – render each step
- **gamma** – reward decay
- **plot_func** – additional function for interactive module

Returns None

load_ckpt(*env_name*)

load trained weights

Returns None

save_ckpt(*env_name*)

save trained weights

Returns None

store_transition(*s*, *a*, *r*)

store data in memory buffer

Parameters

- **s** – state
- **a** – act
- **r** – reward

Returns

update(*gamma*)
update policy parameters via stochastic gradient ascent

Returns None

6.3 Default Hyper-parameters

rlzoo.algorithms.pg.default.**atari**(*env*, *default_seed=True*)

rlzoo.algorithms.pg.default.**box2d**(*env*, *default_seed=True*)

rlzoo.algorithms.pg.default.**classic_control**(*env*, *default_seed=True*)

rlzoo.algorithms.pg.default.**dm_control**(*env*, *default_seed=True*)

rlzoo.algorithms.pg.default.**mujoco**(*env*, *default_seed=True*)

rlzoo.algorithms.pg.default.**rlbench**(*env*, *default_seed=True*)

rlzoo.algorithms.pg.default.**robotics**(*env*, *default_seed=True*)

CHAPTER 7

AC

7.1 Example

```python
from rlzoo.common.env_wrappers import build_env
from rlzoo.common.utils import call_default_params
from rlzoo.algorithms import AC

AlgName = 'AC'
EnvName = 'PongNoFrameskip-v4'
EnvType = 'atari'

# EnvName = 'Pendulum-v0'
# EnvType = 'classic_control'

# EnvName = 'BipedalWalker-v2'
# EnvType = 'box2d'

# EnvName = 'Ant-v2'
# EnvType = 'mujoco'

# EnvName = 'FetchPush-v1'
# EnvType = 'robotics'

# EnvName = 'FishSwim-v0'
# EnvType = 'dm_control'

# EnvName = 'ReachTarget'
# EnvType = 'rlbench'

env = build_env(EnvName, EnvType)
alg_params, learn_params = call_default_params(env, EnvType, AlgName)
alg = eval(AlgName+'(**alg_params)')
alg.learn(env=env, mode='train', render=False, **learn_params)
alg.learn(env=env, mode='test', render=True, **learn_params)
```

7.2 Actor-Critic

class rlzoo.algorithms.ac.ac.**AC**(*net_list*, *optimizers_list*, *gamma=0.9*)

get_action(*s*)

get_action_greedy(*s*)

learn(*env*, *train_episodes=1000*, *test_episodes=500*, *max_steps=200*, *save_interval=100*, *mode='train'*, *render=False*, *plot_func=None*)

Parameters

- **env** – learning environment
- **train_episodes** – total number of episodes for training
- **test_episodes** – total number of episodes for testing
- **max_steps** – maximum number of steps for one episode
- **save_interval** – time steps for saving the weights and plotting the results
- **mode** – 'train' or 'test'
- **render** – if true, visualize the environment
- **plot_func** – additional function for interactive module

load_ckpt(*env_name*)

save_ckpt(*env_name*)

update(*s*, *a*, *r*, *s_*)

7.3 Default Hyper-parameters

rlzoo.algorithms.ac.default.**atari**(*env*, *default_seed=True*)

rlzoo.algorithms.ac.default.**box2d**(*env*, *default_seed=True*)

rlzoo.algorithms.ac.default.**classic_control**(*env*, *default_seed=True*)

rlzoo.algorithms.ac.default.**dm_control**(*env*, *default_seed=True*)

rlzoo.algorithms.ac.default.**mujoco**(*env*, *default_seed=True*)

rlzoo.algorithms.ac.default.**rlbench**(*env*, *default_seed=True*)

rlzoo.algorithms.ac.default.**robotics**(*env*, *default_seed=True*)

CHAPTER 8

A3C

8.1 Example

```python
from rlzoo.common.env_wrappers import build_env
from rlzoo.common.utils import call_default_params
from rlzoo.algorithms import A3C

AlgName = 'A3C'
EnvName = 'PongNoFrameskip-v4'
EnvType = 'atari'

# EnvName = 'Pendulum-v0'  # only continuous action
# EnvType = 'classic_control'

# EnvName = 'BipedalWalker-v2'
# EnvType = 'box2d'

# EnvName = 'Ant-v2'
# EnvType = 'mujoco'

# EnvName = 'FetchPush-v1'
# EnvType = 'robotics'

# EnvName = 'FishSwim-v0'
# EnvType = 'dm_control'

number_workers = 2  # need to specify number of parallel workers
env = build_env(EnvName, EnvType, nenv=number_workers)
alg_params, learn_params = call_default_params(env, EnvType, AlgName)
alg = eval(AlgName+'(**alg_params)')
alg.learn(env=env,  mode='train', render=False, **learn_params)
alg.learn(env=env,  mode='test', render=True, **learn_params)
```

8.2 Asychronous Advantage Actor-Critic

class rlzoo.algorithms.a3c.a3c.**A3C**(*net_list*, *optimizers_list*, *entropy_beta=0.005*)

learn(*env*, *train_episodes=1000*, *test_episodes=10*, *max_steps=150*, *render=False*, *n_workers=1*, *update_itr=10*, *gamma=0.99*, *save_interval=500*, *mode='train'*, *plot_func=None*)

Parameters

- **env** – a list of same learning environments
- **train_episodes** – total number of episodes for training
- **test_episodes** – total number of episodes for testing
- **max_steps** – maximum number of steps for one episode
- **render** – render or not
- **n_workers** – manually set number of workers
- **update_itr** – update global policy after several episodes
- **gamma** – reward discount factor
- **save_interval** – timesteps for saving the weights and plotting the results
- **mode** – train or test
- **plot_func** – additional function for interactive module

8.3 Default Hyper-parameters

rlzoo.algorithms.a3c.default.**atari**(*env*, *default_seed=True*)

rlzoo.algorithms.a3c.default.**box2d**(*env*, *default_seed=True*)

rlzoo.algorithms.a3c.default.**classic_control**(*env*, *default_seed=True*)

rlzoo.algorithms.a3c.default.**dm_control**(*env*, *default_seed=True*)

rlzoo.algorithms.a3c.default.**mujoco**(*env*, *default_seed=True*)

rlzoo.algorithms.a3c.default.**rlbench**(*env*, *default_seed=True*)

rlzoo.algorithms.a3c.default.**robotics**(*env*, *default_seed=True*)

CHAPTER 9

DDPG

9.1 Example

```python
from rlzoo.common.env_wrappers import build_env
from rlzoo.common.utils import call_default_params
from rlzoo.algorithms import DDPG

AlgName = 'DDPG'
EnvName = 'Pendulum-v0'  # only continuous action
EnvType = 'classic_control'

# EnvName = 'BipedalWalker-v2'
# EnvType = 'box2d'

# EnvName = 'Ant-v2'
# EnvType = 'mujoco'

# EnvName = 'FetchPush-v1'
# EnvType = 'robotics'

# EnvName = 'FishSwim-v0'
# EnvType = 'dm_control'

# EnvName = 'ReachTarget'
# EnvType = 'rlbench'

env = build_env(EnvName, EnvType)
alg_params, learn_params = call_default_params(env, EnvType, AlgName)
alg = eval(AlgName+'(**alg_params)')
alg.learn(env=env, mode='train', render=False, **learn_params)
alg.learn(env=env, mode='test', render=True, **learn_params)
```

9.2 Deep Deterministic Policy Gradient

class `rlzoo.algorithms.ddpg.ddpg.`**`DDPG`**(*net_list*, *optimizers_list*, *replay_buffer_size*, *action_range=1.0*, *tau=0.01*)

DDPG class

`ema_update`()

Soft updating by exponential smoothing

Returns None

`get_action`(*s*, *noise_scale*)

Choose action with exploration

Parameters **s** – state

Returns action

`get_action_greedy`(*s*)

Choose action

Parameters **s** – state

Returns action

`learn`(*env*, *train_episodes=200*, *test_episodes=100*, *max_steps=200*, *save_interval=10*, *explore_steps=500*, *mode='train'*, *render=False*, *batch_size=32*, *gamma=0.9*, *noise_scale=1.0*, *noise_scale_decay=0.995*, *plot_func=None*)

learn function

Parameters

- **env** – learning environment
- **train_episodes** – total number of episodes for training
- **test_episodes** – total number of episodes for testing
- **max_steps** – maximum number of steps for one episode
- **save_interval** – time steps for saving
- **explore_steps** – for random action sampling in the beginning of training
- **mode** – train or test mode
- **render** – render each step
- **batch_size** – update batch size
- **gamma** – reward decay factor
- **noise_scale** – range of action noise for exploration
- **noise_scale_decay** – noise scale decay factor
- **plot_func** – additional function for interactive module

Returns None

`load_ckpt`(*env_name*)

load trained weights

Returns None

`sample_action`()

generate random actions for exploration

save_ckpt(*env_name*)
save trained weights

Returns None

store_transition(*s*, *a*, *r*, *s_*, *d*)
Store data in data buffer

Parameters

- **s** – state
- **a** – act
- **r** – reward
- **s** – next state

Returns None

update(*batch_size*, *gamma*)
Update parameters

Parameters

- **batch_size** – update batch size
- **gamma** – reward decay factor

Returns

9.3 Default Hyper-parameters

rlzoo.algorithms.ddpg.default.**box2d**(*env*, *default_seed=True*)

rlzoo.algorithms.ddpg.default.**classic_control**(*env*, *default_seed=True*)

rlzoo.algorithms.ddpg.default.**dm_control**(*env*, *default_seed=True*)

rlzoo.algorithms.ddpg.default.**mujoco**(*env*, *default_seed=True*)

rlzoo.algorithms.ddpg.default.**rlbench**(*env*, *default_seed=True*)

rlzoo.algorithms.ddpg.default.**robotics**(*env*, *default_seed=True*)

CHAPTER 10

TD3

10.1 Example

```
from rlzoo.common.env_wrappers import build_env
from rlzoo.common.utils import call_default_params
from rlzoo.algorithms import TD3

AlgName = 'TD3'
EnvName = 'Pendulum-v0'  # only continuous action
EnvType = 'classic_control'

# EnvName = 'BipedalWalker-v2'
# EnvType = 'box2d'

# EnvName = 'Ant-v2'
# EnvType = 'mujoco'

# EnvName = 'FetchPush-v1'
# EnvType = 'robotics'

# EnvName = 'FishSwim-v0'
# EnvType = 'dm_control'

# EnvName = 'ReachTarget'
# EnvType = 'rlbench'

env = build_env(EnvName, EnvType)
alg_params, learn_params = call_default_params(env, EnvType, AlgName)
alg = eval(AlgName+'(**alg_params)')
alg.learn(env=env, mode='train', render=False, **learn_params)
alg.learn(env=env, mode='test', render=True, **learn_params)
```

10.2 Twin Delayed DDPG

class rlzoo.algorithms.td3.td3.**TD3**(*net_list*, *optimizers_list*, *replay_buffer_capacity=500000.0*, *policy_target_update_interval=5*)

twin-delayed ddpg

evaluate(*state*, *eval_noise_scale*, *target=False*)

generate action with state for calculating gradients;

Parameters eval_noise_scale – as the trick of target policy smoothing, for generating noisy actions.

get_action(*state*, *explore_noise_scale*)

generate action with state for interaction with envronment

get_action_greedy(*state*)

generate action with state for interaction with envronment

learn(*env*, *train_episodes=1000*, *test_episodes=1000*, *max_steps=150*, *batch_size=64*, *explore_steps=500*, *update_itr=3*, *reward_scale=1.0*, *save_interval=10*, *explore_noise_scale=1.0*, *eval_noise_scale=0.5*, *mode='train'*, *render=False*, *plot_func=None*)

Parameters

- **env** – learning environment
- **train_episodes** – total number of episodes for training
- **test_episodes** – total number of episodes for testing
- **max_steps** – maximum number of steps for one episode
- **batch_size** – udpate batchsize
- **explore_steps** – for random action sampling in the beginning of training
- **update_itr** – repeated updates for single step
- **reward_scale** – value range of reward
- **save_interval** – timesteps for saving the weights and plotting the results
- **explore_noise_scale** – range of action noise for exploration
- **eval_noise_scale** – range of action noise for evaluation of action value
- **mode** – 'train' or 'test'
- **render** – if true, visualize the environment
- **plot_func** – additional function for interactive module

load_ckpt(*env_name*)

sample_action()

generate random actions for exploration

save_ckpt(*env_name*)

target_ini(*net*, *target_net*)

hard-copy update for initializing target networks

target_soft_update(*net*, *target_net*, *soft_tau*)

soft update the target net with Polyak averaging

update(*batch_size*, *eval_noise_scale*, *reward_scale=1.0*, *gamma=0.9*, *soft_tau=0.01*)
update all networks in TD3

10.3 Default Hyper-parameters

rlzoo.algorithms.td3.default.**box2d**(*env*, *default_seed=True*)

rlzoo.algorithms.td3.default.**classic_control**(*env*, *default_seed=True*)

rlzoo.algorithms.td3.default.**dm_control**(*env*, *default_seed=True*)

rlzoo.algorithms.td3.default.**mujoco**(*env*, *default_seed=True*)

rlzoo.algorithms.td3.default.**rlbench**(*env*, *default_seed=True*)

rlzoo.algorithms.td3.default.**robotics**(*env*, *default_seed=True*)

CHAPTER 11

SAC

11.1 Example

```
from rlzoo.common.env_wrappers import build_env
from rlzoo.common.utils import call_default_params
from rlzoo.algorithms import SAC

AlgName = 'SAC'
EnvName = 'Pendulum-v0'  # only continuous action
EnvType = 'classic_control'

# EnvName = 'BipedalWalker-v2'
# EnvType = 'box2d'

# EnvName = 'Ant-v2'
# EnvType = 'mujoco'

# EnvName = 'FetchPush-v1'
# EnvType = 'robotics'

# EnvName = 'FishSwim-v0'
# EnvType = 'dm_control'

# EnvName = 'ReachTarget'
# EnvType = 'rlbench'

env = build_env(EnvName, EnvType)
alg_params, learn_params = call_default_params(env, EnvType, AlgName)
alg = eval(AlgName+'(**alg_params)')
alg.learn(env=env, mode='train', render=False, **learn_params)
alg.learn(env=env, mode='test', render=True, **learn_params)
```

11.2 Soft Actor-Critic

class rlzoo.algorithms.sac.sac.**SAC**(*net_list*, *optimizers_list*, *replay_buffer_capacity=500000.0*)

Soft Actor-Critic

evaluate(*state*, *epsilon=1e-06*)
generate action with state for calculating gradients

get_action(*state*)
generate action with state for interaction with envronment

get_action_greedy(*state*)
generate action with state for interaction with envronment

learn(*env*, *train_episodes=1000*, *test_episodes=1000*, *max_steps=150*, *batch_size=64*, *explore_steps=500*, *update_itr=3*, *policy_target_update_interval=3*, *reward_scale=1.0*, *save_interval=20*, *mode='train'*, *AUTO_ENTROPY=True*, *render=False*, *plot_func=None*)

Parameters

- **env** – learning environment
- **train_episodes** – total number of episodes for training
- **test_episodes** – total number of episodes for testing
- **max_steps** – maximum number of steps for one episode
- **batch_size** – udpate batchsize
- **explore_steps** – for random action sampling in the beginning of training
- **update_itr** – repeated updates for single step
- **policy_target_update_interval** – delayed update for the policy network and target networks
- **reward_scale** – value range of reward
- **save_interval** – timesteps for saving the weights and plotting the results
- **mode** – 'train' or 'test'
- **AUTO_ENTROPY** – automatically updating variable alpha for entropy
- **render** – if true, visualize the environment
- **plot_func** – additional function for interactive module

load_ckpt(*env_name*)
load trained weights

sample_action()
generate random actions for exploration

save_ckpt(*env_name*)
save trained weights

target_ini(*net*, *target_net*)
hard-copy update for initializing target networks

target_soft_update(*net*, *target_net*, *soft_tau*)
soft update the target net with Polyak averaging

update(*batch_size*, *reward_scale=10.0*, *auto_entropy=True*, *target_entropy=-2*, *gamma=0.99*, *soft_tau=0.01*)
update all networks in SAC

11.3 Default Hyper-parameters

`rlzoo.algorithms.sac.default.`**box2d**(*env*, *default_seed=True*)

`rlzoo.algorithms.sac.default.`**classic_control**(*env*, *default_seed=True*)

`rlzoo.algorithms.sac.default.`**dm_control**(*env*, *default_seed=True*)

`rlzoo.algorithms.sac.default.`**mujoco**(*env*, *default_seed=True*)

`rlzoo.algorithms.sac.default.`**rlbench**(*env*, *default_seed=True*)

`rlzoo.algorithms.sac.default.`**robotics**(*env*, *default_seed=True*)

CHAPTER 12

TRPO

12.1 Example

```
from rlzoo.common.env_wrappers import build_env
from rlzoo.common.utils import call_default_params
from rlzoo.algorithms import TD3

AlgName = 'TRPO'
EnvName = 'PongNoFrameskip-v4'
EnvType = 'atari'

# EnvName = 'CartPole-v0'
# EnvType = 'classic_control'

# EnvName = 'BipedalWalker-v2'
# EnvType = 'box2d'

# EnvName = 'Ant-v2'
# EnvType = 'mujoco'

# EnvName = 'FetchPush-v1'
# EnvType = 'robotics'

# EnvName = 'FishSwim-v0'
# EnvType = 'dm_control'

# EnvName = 'ReachTarget'
# EnvType = 'rlbench'

env = build_env(EnvName, EnvType)
alg_params, learn_params = call_default_params(env, EnvType, AlgName)
alg = eval(AlgName+'(**alg_params)')
alg.learn(env=env, mode='train', render=False, **learn_params)
alg.learn(env=env, mode='test', render=True, **learn_params)
```

12.2 Trust Region Policy Optimization

class rlzoo.algorithms.trpo.trpo.**TRPO**(*net_list*, *optimizers_list*, *damping_coeff=0.1*, *cg_iters=10*, *delta=0.01*)

trpo class

a_train(*s*, *a*, *adv*, *oldpi_prob*, *backtrack_iters*, *backtrack_coeff*)

static assign_params_from_flat(*x*, *params*)

assign params from flat input

Parameters

- **x** –
- **params** –

Returns group

c_train(*tfdc_r*, *s*)

Update actor network

Parameters

- **tfdc_r** – cumulative reward
- **s** – state

Returns None

cal_adv(*tfs*, *tfdc_r*)

Calculate advantage

Parameters

- **tfs** – state
- **tfdc_r** – cumulative reward

Returns advantage

cg(*Ax*, *b*)

Conjugate gradient algorithm (see https://en.wikipedia.org/wiki/Conjugate_gradient_method)

eval(*bs*, *ba*, *badv*, *oldpi_prob*)

static flat_concat(*xs*)

flat concat input

Parameters **xs** – a list of tensor

Returns flat tensor

get_action(*s*)

Choose action

Parameters **s** – state

Returns clipped act

get_action_greedy(*s*)

Choose action

Parameters **s** – state

Returns clipped act

get_pi_params()
get actor trainable parameters

Returns flat actor trainable parameters

get_v(*s*)
Compute value

Parameters **s** – state

Returns value

gradient(*inputs*)
pi gradients

Parameters **inputs** – a list of x_ph, a_ph, adv_ph, ret_ph, logp_old_ph and other inputs

Returns gradient

hessian_vector_product(*s, a, adv, oldpi_prob, v_ph*)

learn(*env, train_episodes=200, test_episodes=100, max_steps=200, save_interval=10, gamma=0.9, mode='train', render=False, batch_size=32, backtrack_iters=10, backtrack_coeff=0.8, train_critic_iters=80, plot_func=None*)
learn function

Parameters

- **env** – learning environment
- **train_episodes** – total number of episodes for training
- **test_episodes** – total number of episodes for testing
- **max_steps** – maximum number of steps for one episode
- **save_interval** – time steps for saving
- **gamma** – reward discount factor
- **mode** – train or test
- **render** – render each step
- **batch_size** – update batch size
- **backtrack_iters** – Maximum number of steps allowed in the backtracking line search
- **backtrack_coeff** – How far back to step during backtracking line search
- **train_critic_iters** – critic update iteration steps

Returns None

load_ckpt(*env_name*)
load trained weights

Returns None

pi_loss(*inputs*)
calculate pi loss

Parameters **inputs** – a list of x_ph, a_ph, adv_ph, ret_ph, logp_old_ph and other inputs

Returns pi loss

save_ckpt(*env_name*)
save trained weights

Returns None

set_pi_params (*v_ph*)
set actor trainable parameters

Parameters **v_ph** – inputs

Returns None

update (*bs*, *ba*, *br*, *train_critic_iters*, *backtrack_iters*, *backtrack_coeff*)
update trpo

Returns None

12.3 Default Hyper-parameters

rlzoo.algorithms.trpo.default.**atari** (*env*, *default_seed=True*)

rlzoo.algorithms.trpo.default.**box2d** (*env*, *default_seed=True*)

rlzoo.algorithms.trpo.default.**classic_control** (*env*, *default_seed=True*)

rlzoo.algorithms.trpo.default.**dm_control** (*env*, *default_seed=True*)

rlzoo.algorithms.trpo.default.**mujoco** (*env*, *default_seed=True*)

rlzoo.algorithms.trpo.default.**rlbench** (*env*, *default_seed=True*)

rlzoo.algorithms.trpo.default.**robotics** (*env*, *default_seed=True*)

CHAPTER 13

PPO

13.1 Example

```
from rlzoo.common.env_wrappers import build_env
from rlzoo.common.utils import call_default_params
from rlzoo.algorithms import PPO

EnvName = 'PongNoFrameskip-v4'
EnvType = 'atari'

# EnvName = 'Pendulum-v0'
# EnvType = 'classic_control'

# EnvName = 'BipedalWalker-v2'
# EnvType = 'box2d'

# EnvName = 'Ant-v2'
# EnvType = 'mujoco'

# EnvName = 'FetchPush-v1'
# EnvType = 'robotics'

# EnvName = 'FishSwim-v0'
# EnvType = 'dm_control'

# EnvName = 'ReachTarget'
# EnvType = 'rlbench'

env = build_env(EnvName, EnvType)
alg_params, learn_params = call_default_params(env, EnvType, 'PPO')
alg = PPO(method='clip', **alg_params) # specify 'clip' or 'penalty' method for PPO
alg.learn(env=env,  mode='train', render=False, **learn_params)
alg.learn(env=env,  mode='test', render=False, **learn_params)
```

13.2 Proximal Policy Optimization (Penalty)

class rlzoo.algorithms.ppo_penalty.ppo_penalty.**PPO_PENALTY**(*net_list*, *optimizers_list*, *kl_target=0.01*, *lam=0.5*)

PPO class

a_train(*tfs*, *tfa*, *tfadv*, *oldpi_prob*)
Update policy network

Parameters

- **tfs** – state
- **tfa** – act
- **tfadv** – advantage

Returns

c_train(*tfdc_r*, *s*)
Update actor network

Parameters

- **tfdc_r** – cumulative reward
- **s** – state

Returns None

cal_adv(*tfs*, *tfdc_r*)
Calculate advantage

Parameters

- **tfs** – state
- **tfdc_r** – cumulative reward

Returns advantage

get_action(*s*)
Choose action

Parameters **s** – state

Returns clipped act

get_action_greedy(*s*)
Choose action

Parameters **s** – state

Returns clipped act

get_v(*s*)
Compute value

Parameters **s** – state

Returns value

learn(*env*, *train_episodes=1000*, *test_episodes=10*, *max_steps=200*, *save_interval=10*, *gamma=0.9*, *mode='train'*, *render=False*, *batch_size=32*, *a_update_steps=10*, *c_update_steps=10*, *plot_func=None*)
learn function

Parameters

- **env** – learning environment
- **train_episodes** – total number of episodes for training
- **test_episodes** – total number of episodes for testing
- **max_steps** – maximum number of steps for one episode
- **save_interval** – time steps for saving
- **gamma** – reward discount factor
- **mode** – train or test
- **render** – render each step
- **batch_size** – update batch size
- **a_update_steps** – actor update iteration steps
- **c_update_steps** – critic update iteration steps
- **plot_func** – additional function for interactive module

Returns None

load_ckpt(*env_name*)
load trained weights

Returns None

save_ckpt(*env_name*)
save trained weights

Returns None

update(*s*, *a*, *r*, *a_update_steps*, *c_update_steps*)
Update parameter with the constraint of KL divergent

Parameters

- **s** – state
- **a** – act
- **r** – reward

Returns None

13.3 Proximal Policy Optimization (Clip)

class rlzoo.algorithms.ppo_clip.ppo_clip.**PPO_CLIP**(*net_list*, *optimizers_list*, *epsilon=0.2*)
PPO class

a_train(*tfs*, *tfa*, *tfadv*, *oldpi_prob*)
Update policy network

Parameters

- **tfs** – state
- **tfa** – act
- **tfadv** – advantage
- **oldpi_prob** – old policy distribution

Returns None

c_train (*tfdc_r*, *s*)
Update actor network

Parameters

- **tfdc_r** – cumulative reward
- **s** – state

Returns None

cal_adv (*tfs*, *tfdc_r*)
Calculate advantage

Parameters

- **tfs** – state
- **tfdc_r** – cumulative reward

Returns advantage

get_action (*s*)
Choose action

Parameters **s** – state

Returns clipped act

get_action_greedy (*s*)
Choose action :param s: state :return: clipped act

get_v (*s*)
Compute value

Parameters **s** – state

Returns value

learn (*env*, *train_episodes=200*, *test_episodes=100*, *max_steps=200*, *save_interval=10*, *gamma=0.9*, *mode='train'*, *render=False*, *batch_size=32*, *a_update_steps=10*, *c_update_steps=10*, *plot_func=None*)
learn function

Parameters

- **env** – learning environment
- **train_episodes** – total number of episodes for training
- **test_episodes** – total number of episodes for testing
- **max_steps** – maximum number of steps for one episode
- **save_interval** – timesteps for saving
- **gamma** – reward discount factor
- **mode** – train or test

- **render** – render each step
- **batch_size** – udpate batchsize
- **a_update_steps** – actor update iteration steps
- **c_update_steps** – critic update iteration steps
- **plot_func** – additional function for interactive module

Returns None

load_ckpt (*env_name*)
load trained weights

Returns None

save_ckpt (*env_name*)
save trained weights

Returns None

update (*s*, *a*, *r*, *a_update_steps*, *c_update_steps*)
Update parameter with the constraint of KL divergent

Parameters

- **s** – state
- **a** – act
- **r** – reward

Returns None

13.4 Default Hyper-parameters

rlzoo.algorithms.ppo.default.**atari** (*env*, *default_seed=True*)

rlzoo.algorithms.ppo.default.**box2d** (*env*, *default_seed=True*)

rlzoo.algorithms.ppo.default.**classic_control** (*env*, *default_seed=True*)

rlzoo.algorithms.ppo.default.**dm_control** (*env*, *default_seed=True*)

rlzoo.algorithms.ppo.default.**mujoco** (*env*, *default_seed=True*)

rlzoo.algorithms.ppo.default.**rlbench** (*env*, *default_seed=True*)

rlzoo.algorithms.ppo.default.**robotics** (*env*, *default_seed=True*)

CHAPTER 14

DPPO

14.1 Example

```
from rlzoo.common.env_wrappers import build_env
from rlzoo.common.utils import call_default_params
from rlzoo.algorithms import DPPO

EnvName = 'PongNoFrameskip-v4'
EnvType = 'atari'

# EnvName = 'Pendulum-v0'
# EnvType = 'classic_control'

# EnvName = 'BipedalWalker-v2'
# EnvType = 'box2d'

# EnvName = 'Ant-v2'
# EnvType = 'mujoco'

# EnvName = 'FetchPush-v1'
# EnvType = 'robotics'

# EnvName = 'FishSwim-v0'
# EnvType = 'dm_control'

# EnvName = 'ReachTarget'
# EnvType = 'rlbench'

number_workers = 2  # need to specify number of parallel workers
env = build_env(EnvName, EnvType, nenv=number_workers)
alg_params, learn_params = call_default_params(env, EnvType, 'DPPO')
alg = DPPO(method='penalty', **alg_params) # specify 'clip' or 'penalty' method for PPO
alg.learn(env=env,  mode='train', render=False, **learn_params)
```

(continues on next page)

(continued from previous page)

```
alg.learn(env=env,  mode='test', render=True, **learn_params)
```

14.2 Distributed Proximal Policy Optimization (Penalty)

class rlzoo.algorithms.dppo_penalty.dppo_penalty.**DPPO_PENALTY**(*net_list*, *optimizers_list*, *kl_target=0.01*, *lam=0.5*)

PPO class

a_train(*tfs*, *tfa*, *tfadv*, *oldpi_prob*)
Update policy network

Parameters

- **tfs** – state
- **tfa** – act
- **tfadv** – advantage

Returns

c_train(*tfdc_r*, *s*)
Update actor network

Parameters

- **tfdc_r** – cumulative reward
- **s** – state

Returns None

cal_adv(*tfs*, *tfdc_r*)
Calculate advantage

Parameters

- **tfs** – state
- **tfdc_r** – cumulative reward

Returns advantage

get_action(*s*)
Choose action

Parameters **s** – state

Returns clipped act

get_action_greedy(*s*)
Choose action

Parameters **s** – state

Returns clipped act

get_v(*s*)
Compute value

Parameters **s** – state

Returns value

learn (*env*, *train_episodes=200*, *test_episodes=100*, *max_steps=200*, *save_interval=10*, *gamma=0.9*, *mode='train'*, *render=False*, *batch_size=32*, *a_update_steps=10*, *c_update_steps=10*, *n_workers=4*, *plot_func=None*)
learn function

Parameters

- **env** – learning environment
- **train_episodes** – total number of episodes for training
- **test_episodes** – total number of episodes for testing
- **max_steps** – maximum number of steps for one episode
- **save_interval** – time steps for saving
- **gamma** – reward discount factor
- **mode** – train or test
- **render** – render each step
- **batch_size** – update batch size
- **a_update_steps** – actor update iteration steps
- **c_update_steps** – critic update iteration steps
- **n_workers** – number of workers
- **plot_func** – additional function for interactive module

Returns None

load_ckpt (*env_name*)
load trained weights

Returns None

save_ckpt (*env_name*)
save trained weights

Returns None

update (*a_update_steps*, *c_update_steps*, *save_interval*, *env*)
Update

Parameters

- **a_update_steps** – actor update steps
- **c_update_steps** – critic update steps
- **save_interval** – save interval
- **env** – environment

Returns None

14.3 Distributed Proximal Policy Optimization (Clip)

class rlzoo.algorithms.dppo_clip.dppo_clip.**DPPO_CLIP**(*net_list*, *optimizers_list*, *epsilon=0.2*)

PPO class

a_train(*tfs*, *tfa*, *tfadv*, *oldpi_prob*)

Update policy network

Parameters

- **tfs** – state
- **tfa** – act
- **tfadv** – advantage
- **oldpi_prob** – old pi probability of a in s

Returns

c_train(*tfdc_r*, *s*)

Update actor network

Parameters

- **tfdc_r** – cumulative reward
- **s** – state

Returns None

cal_adv(*tfs*, *tfdc_r*)

Calculate advantage

Parameters

- **tfs** – state
- **tfdc_r** – cumulative reward

Returns advantage

get_action(*s*)

Choose action

Parameters **s** – state

Returns clipped act

get_action_greedy(*s*)

Choose action

Parameters **s** – state

Returns clipped act

get_v(*s*)

Compute value

Parameters **s** – state

Returns value

learn(*env*, *train_episodes=200*, *test_episodes=100*, *max_steps=200*, *save_interval=10*, *gamma=0.9*, *mode='train'*, *render=False*, *batch_size=32*, *a_update_steps=10*, *c_update_steps=10*, *n_workers=4*, *plot_func=None*)

learn function

Parameters

- **env** – learning environment
- **train_episodes** – total number of episodes for training
- **test_episodes** – total number of episodes for testing
- **max_steps** – maximum number of steps for one episode
- **save_interval** – time steps for saving
- **gamma** – reward discount factor
- **mode** – train or test
- **render** – render each step
- **batch_size** – update batch size
- **a_update_steps** – actor update iteration steps
- **c_update_steps** – critic update iteration steps
- **n_workers** – number of workers
- **plot_func** – additional function for interactive module

Returns None

load_ckpt(*env_name*)

load trained weights

Returns None

save_ckpt(*env_name*)

save trained weights

Returns None

update(*a_update_steps*, *c_update_steps*, *save_interval*, *env*)

Update

Parameters

- **a_update_steps** – actor update steps
- **c_update_steps** – critic update steps
- **save_interval** – save interval
- **env** – environment

Returns None

14.4 Default Hyper-parameters

rlzoo.algorithms.dppo.default.**atari**(*env*, *default_seed=True*)

rlzoo.algorithms.dppo.default.**box2d**(*env*, *default_seed=True*) `rlzoo.algorithms.dppo.default.`**`classic_control`**(*env*, *default_seed=True*)

`rlzoo.algorithms.dppo.default.`**`dm_control`**(*env*, *default_seed=True*)

`rlzoo.algorithms.dppo.default.`**`mujoco`**(*env*, *default_seed=True*)

`rlzoo.algorithms.dppo.default.`**`robotics`**(*env*, *default_seed=True*)

CHAPTER 15

Basic Networks

15.1 Basic Networks in RLzoo

Basic neural networks

`rlzoo.common.basic_nets.`**`CNN`**`(`*input_shape*, *conv_kwargs=None*`)`
Multiple convolutional layers for approximation Default setting is equal to architecture used in DQN

Parameters

- **`input_shape`** – (tuple[int]) (H, W, C)
- **`conv_kwargs`** – (list[param]) list of conv parameters for tl.layers.Conv2d

Return: input tensor, output tensor

`rlzoo.common.basic_nets.`**`CNNModel`**`(`*input_shape*, *conv_kwargs=None*`)`
Multiple convolutional layers for approximation Default setting is equal to architecture used in DQN

Parameters

- **`input_shape`** – (tuple[int]) (H, W, C)
- **`conv_kwargs`** – (list[param]) list of conv parameters for tl.layers.Conv2d

Return: tl.model.Model

`rlzoo.common.basic_nets.`**`CreateInputLayer`**`(`*state_space*, *conv_kwargs=None*`)`

`rlzoo.common.basic_nets.`**`MLP`**`(`*input_dim*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.Orthogonal object>*, *activation=<function relu>*, **args*, ***kwargs*`)`
Multiple fully-connected layers for approximation

Parameters

- **`input_dim`** – (int) size of input tensor
- **`hidden_dim_list`** – (list[int]) a list of dimensions of hidden layers

- **w_init** – (callable) initialization method for weights
- **activation** – (callable) activation function of hidden layers

Return: input tensor, output tensor

rlzoo.common.basic_nets.**MLPModel**(*input_dim*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.Orthogonal object>*, *activation=<function relu>*, **args*, ***kwargs*)

Multiple fully-connected layers for approximation

Parameters

- **input_dim** – (int) size of input tensor
- **hidden_dim_list** – (list[int]) a list of dimensions of hidden layers
- **w_init** – (callable) initialization method for weights
- **activation** – (callable) activation function of hidden layers

Return: input tensor, output tensor

CHAPTER 16

Policy Networks

16.1 Policy Networks in RLzoo

class rlzoo.common.policy_networks.**StochasticContinuousPolicyNetwork**(*state_shape*, *action_shape*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.* *object>*, *activation=<function relu>*, *output_activation=None*, *log_std_min=-20*, *log_std_max=2*, *trainable=True*)

Bases: tensorlayer.models.core.Model

__init__(*state_shape*, *action_shape*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.GlorotNormal object>*, *activation=<function relu>*, *output_activation=None*, *log_std_min=-20*, *log_std_max=2*, *trainable=True*)

Stochastic continuous policy network with multiple fully-connected layers or convolutional layers (according to state shape)

Parameters

- **state_shape** – (tuple[int]) shape of the state, for example, (state_dim,) for single-dimensional state

- **action_shape** – (tuple[int]) shape of the action, for example, (action_dim,) for single-dimensional action
- **hidden_dim_list** – (list[int]) a list of dimensions of hidden layers
- **w_init** – (callable) weights initialization
- **activation** – (callable) activation function
- **output_activation** – (callable or None) output activation function
- **log_std_min** – (float) lower bound of standard deviation of action
- **log_std_max** – (float) upper bound of standard deviation of action
- **trainable** – (bool) set training and evaluation mode

__module__ = 'rlzoo.common.policy_networks'

class rlzoo.common.policy_networks.**DeterministicContinuousPolicyNetwork**(*state_shape*, *action_shape*, *hidden_dim_list*, *w_init=<tensorflow.python.ob- ject>*, *activation=<function relu>*, *output_activation=<function tanh>*, *trainable=True*)

Bases: tensorlayer.models.core.Model

__init__(*state_shape*, *action_shape*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.GlorotNormal object>*, *activation=<function relu>*, *output_activation=<function tanh>*, *trainable=True*)

Deterministic continuous policy network with multiple fully-connected layers or convolutional layers (according to state shape)

Parameters

- **state_shape** – (tuple[int]) shape of the state, for example, (state_dim,) for single-dimensional state
- **action_shape** – (tuple[int]) shape of the action, for example, (action_dim,) for single-dimensional action
- **hidden_dim_list** – (list[int]) a list of dimensions of hidden layers
- **w_init** – (callable) weights initialization
- **activation** – (callable) activation function
- **output_activation** – (callable or None) output activation function
- **trainable** – (bool) set training and evaluation mode

__module__ = 'rlzoo.common.policy_networks'

class rlzoo.common.policy_networks.**DeterministicPolicyNetwork**(*state_space*, *action_space*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.GlorotNormal object>*, *activation=<function relu>*, *output_activation=<function tanh>*, *trainable=True*, *name=None*)

Bases: tensorlayer.models.core.Model

__call__(*states*, **args*, ***kwargs*)

Forward input tensors through this network by calling.

inputs [Tensor or list of Tensors, numpy.ndarray of list of numpy.ndarray] Inputs for network forwarding

is_train [boolean] Network's mode for this time forwarding. If 'is_train' == True, this network is set as training mode. If 'is_train' == False, this network is set as evaluation mode

kwargs : For other keyword-only arguments.

__init__(*state_space*, *action_space*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.GlorotNormal object>*, *activation=<function relu>*, *output_activation=<function tanh>*, *trainable=True*, *name=None*)

Deterministic continuous/discrete policy network with multiple fully-connected layers

Parameters

- **state_space** – (gym.spaces) space of the state from gym environments
- **action_space** – (gym.spaces) space of the action from gym environments
- **hidden_dim_list** – (list[int]) a list of dimensions of hidden layers
- **w_init** – (callable) weights initialization
- **activation** – (callable) activation function
- **output_activation** – (callable or None) output activation function
- **trainable** – (bool) set training and evaluation mode

__module__ = 'rlzoo.common.policy_networks'

action_shape

action_space

random_sample()

generate random actions for exploration

state_shape

state_space

class rlzoo.common.policy_networks.**StochasticPolicyNetwork**(*state_space*, *action_space*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.GlorotNormal object>*, *activation=<function relu>*, *output_activation=<function tanh>*, *log_std_min=-20*, *log_std_max=2*, *trainable=True*, *name=None*, *state_conditioned=False*)

Bases: tensorlayer.models.core.Model

__call__(*states*, **args*, *greedy=False*, ***kwargs*)

Forward input tensors through this network by calling.

inputs [Tensor or list of Tensors, numpy.ndarray of list of numpy.ndarray] Inputs for network forwarding

is_train [boolean] Network's mode for this time forwarding. If 'is_train' == True, this network is set as training mode. If 'is_train' == False, this network is set as evaluation mode

kwargs : For other keyword-only arguments.

__init__(*state_space*, *action_space*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.GlorotNormal object>*, *activation=<function relu>*, *output_activation=<function tanh>*, *log_std_min=-20*, *log_std_max=2*, *trainable=True*, *name=None*, *state_conditioned=False*)

Stochastic continuous/discrete policy network with multiple fully-connected layers

Parameters

- **state_space** – (gym.spaces) space of the state from gym environments
- **action_space** – (gym.spaces) space of the action from gym environments
- **hidden_dim_list** – (list[int]) a list of dimensions of hidden layers
- **w_init** – (callable) weights initialization
- **activation** – (callable) activation function
- **output_activation** – (callable or None) output activation function
- **log_std_min** – (float) lower bound of standard deviation of action
- **log_std_max** – (float) upper bound of standard deviation of action
- **trainable** – (bool) set training and evaluation mode

Tips: We recommend to use tf.nn.tanh for output_activation, especially for continuous action space, to ensure the final action range is exactly the same as declared in action space after action normalization.

__module__ = 'rlzoo.common.policy_networks'

action_shape

action_space

random_sample()

generate random actions for exploration

state_shape

state_space

CHAPTER 17

Value Networks

17.1 Value Networks in RLzoo

class rlzoo.common.value_networks.**ValueNetwork**(*state_space*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.GlorotNormal object>*, *activation=<function relu>*, *output_activation=None*, *trainable=True*, *name=None*)

__call__(*states*, **args*, ***kwargs*)

Forward input tensors through this network by calling.

inputs [Tensor or list of Tensors, numpy.ndarray of list of numpy.ndarray] Inputs for network forwarding

is_train [boolean] Network's mode for this time forwarding. If 'is_train' == True, this network is set as training mode. If 'is_train' == False, this network is set as evaluation mode

kwargs : For other keyword-only arguments.

__init__(*state_space*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.GlorotNormal object>*, *activation=<function relu>*, *output_activation=None*, *trainable=True*, *name=None*)

Value network with multiple fully-connected layers or convolutional layers (according to state shape)

Parameters

- **state_space** – (gym.spaces) space of the state from gym environments
- **hidden_dim_list** – (list[int]) a list of dimensions of hidden layers
- **w_init** – (callable) weights initialization
- **activation** – (callable) activation function
- **output_activation** – (callable or None) output activation function
- **trainable** – (bool) set training and evaluation mode

__module__ = 'rlzoo.common.value_networks'

state_shape

state_space

class rlzoo.common.value_networks.**MlpQNetwork**(*state_shape*, *action_shape*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.GlorotNormal object>*, *activation=<function relu>*, *output_activation=None*, *trainable=True*)

__init__(*state_shape*, *action_shape*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.GlorotNormal object>*, *activation=<function relu>*, *output_activation=None*, *trainable=True*)

Q-value network with multiple fully-connected layers

Inputs: (state tensor, action tensor)

Parameters

- **state_shape** – (tuple[int]) shape of the state, for example, (state_dim,) for single-dimensional state
- **action_shape** – (tuple[int]) shape of the action, for example, (action_dim,) for single-dimensional action
- **hidden_dim_list** – (list[int]) a list of dimensions of hidden layers
- **w_init** – (callable) weights initialization
- **activation** – (callable) activation function
- **output_activation** – (callable or None) output activation function
- **trainable** – (bool) set training and evaluation mode

__module__ = 'rlzoo.common.value_networks'

class rlzoo.common.value_networks.**QNetwork**(*state_space*, *action_space*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.GlorotNormal object>*, *activation=<function relu>*, *output_activation=None*, *trainable=True*, *name=None*, *state_only=False*, *dueling=False*)

__call__(*inputs*, **args*, ***kwargs*)

Forward input tensors through this network by calling.

inputs [Tensor or list of Tensors, numpy.ndarray of list of numpy.ndarray] Inputs for network forwarding

is_train [boolean] Network's mode for this time forwarding. If 'is_train' == True, this network is set as training mode. If 'is_train' == False, this network is set as evaluation mode

kwargs : For other keyword-only arguments.

__init__(*state_space*, *action_space*, *hidden_dim_list*, *w_init=<tensorflow.python.ops.init_ops_v2.GlorotNormal object>*, *activation=<function relu>*, *output_activation=None*, *trainable=True*, *name=None*, *state_only=False*, *dueling=False*)

Q-value network with multiple fully-connected layers or convolutional layers (according to state shape)

Parameters

- **state_space** – (gym.spaces) space of the state from gym environments
- **action_space** – (gym.spaces) space of the action from gym environments
- **hidden_dim_list** – (list[int]) a list of dimensions of hidden layers

- **w_init** – (callable) weights initialization
- **activation** – (callable) activation function
- **output_activation** – (callable or None) output activation function
- **trainable** – (bool) set training and evaluation mode
- **name** – (str) name the model
- **state_only** – (bool) only input state or not, available in discrete action space
- **dueling** – (bool) whether use the dueling output or not, available in discrete action space

__module__ = 'rlzoo.common.value_networks'

action_shape

action_space

state_shape

state_space

CHAPTER 18

Replay Buffer

18.1 Replay Buffer in RLzoo

Functions for utilization.

Requirements tensorflow==2.0.0a0 tensorlayer==2.0.1

class rlzoo.common.buffer.**HindsightReplayBuffer**(*capacity*, *hindsight_freq*, *goal_type*, *reward_func*, *done_func*)

Bases: *rlzoo.common.buffer.ReplayBuffer*

Hindsight Experience Replay In this buffer, state is a tuple consists of (observation, goal)

GOAL_EPISODE = 'episode'

GOAL_FUTURE = 'future'

GOAL_RANDOM = 'random'

__init__(*capacity*, *hindsight_freq*, *goal_type*, *reward_func*, *done_func*)

Parameters

- **(int)** (*hindsight_freq*) – How many hindsight transitions will be generated for each real transition
- **(str)** (*goal_type*) – The generatation method of hindsight goals. Should be HER_GOAL_*
- **(callable)** (*done_func*) – goal (np.array) X next_state (np.array) -> reward (float)
- **(callable)** – goal (np.array) X next_state (np.array) -> done_flag (bool)

__module__ = 'rlzoo.common.buffer'

push(**args*, ***kwargs*)

push_episode(*states*, *actions*, *rewards*, *next_states*, *dones*)

class rlzoo.common.buffer.**MinSegmentTree**(*capacity*)
Bases: *rlzoo.common.buffer.SegmentTree*

__init__(*capacity*)
Build a Segment Tree data structure.

https://en.wikipedia.org/wiki/Segment_tree

Can be used as regular array, but with two important differences:

a) setting item's value is slightly slower. It is O(lg capacity) instead of O(1).

b) user has access to an efficient (O(log segment size)) *reduce* operation which reduces *operation* over a contiguous subsequence of items in the array.

Parameters

- **apacity** – (int) Total size of the array - must be a power of two.
- **operation** – (lambda obj, obj -> obj) and operation for combining elements (eg. sum, max) must form a mathematical group together with the set of possible values for array elements (i.e. be associative)
- **neutral_element** – (obj) neutral element for the operation above. eg. float('-inf') for max and 0 for sum.

__module__ = 'rlzoo.common.buffer'

min(*start=0*, *end=None*)
Returns min(arr[start], . . . , arr[end])

class rlzoo.common.buffer.**PrioritizedReplayBuffer**(*capacity*, *alpha*, *beta*)
Bases: *rlzoo.common.buffer.ReplayBuffer*

__init__(*capacity*, *alpha*, *beta*)
Create Prioritized Replay buffer.

Parameters

- **capacity** – (int) Max number of transitions to store in the buffer. When the buffer overflows the old memories are dropped.
- **alpha** – (float) how much prioritization is used (0 - no prioritization, 1 - full prioritization)

See Also: ReplayBuffer.__init__

__module__ = 'rlzoo.common.buffer'

push(**args*)
See ReplayBuffer.store_effect

sample(*batch_size*)
Sample a batch of experiences

update_priorities(*idxes*, *priorities*)
Update priorities of sampled transitions

class rlzoo.common.buffer.**ReplayBuffer**(*capacity*)
Bases: object

A standard ring buffer for storing transitions and sampling for training

__dict__ = mappingproxy({'__module__': 'rlzoo.common.buffer', '__doc__': 'A standard

__init__(*capacity*)
Initialize self. See help(type(self)) for accurate signature.

__len__()

__module__ = 'rlzoo.common.buffer'

__weakref__
list of weak references to the object (if defined)

push(*state*, *action*, *reward*, *next_state*, *done*)

sample(*batch_size*)

class rlzoo.common.buffer.**SegmentTree**(*capacity*, *operation*, *neutral_element*)
Bases: `object`

__dict__ = mappingproxy({'__module__': 'rlzoo.common.buffer', '__init__': <function

__getitem__(*idx*)

__init__(*capacity*, *operation*, *neutral_element*)
Build a Segment Tree data structure.

https://en.wikipedia.org/wiki/Segment_tree

Can be used as regular array, but with two important differences:

a) setting item's value is slightly slower. It is O(lg capacity) instead of O(1).

b) user has access to an efficient (O(log segment size)) *reduce* operation which reduces *operation* over a contiguous subsequence of items in the array.

Parameters

- **apacity** – (int) Total size of the array - must be a power of two.
- **operation** – (lambda obj, obj -> obj) and operation for combining elements (eg. sum, max) must form a mathematical group together with the set of possible values for array elements (i.e. be associative)
- **neutral_element** – (obj) neutral element for the operation above. eg. float('-inf') for max and 0 for sum.

__module__ = 'rlzoo.common.buffer'

__setitem__(*idx*, *val*)

__weakref__
list of weak references to the object (if defined)

reduce(*start=0*, *end=None*)
Returns result of applying *self.operation* to a contiguous subsequence of the array.

Parameters

- **start** – (int) beginning of the subsequence
- **end** – (int) end of the subsequences

Returns: reduced: (obj) result of reducing self.operation over the specified range of array.

class rlzoo.common.buffer.**SumSegmentTree**(*capacity*)
Bases: *rlzoo.common.buffer.SegmentTree*

`__init__`(*capacity*)

Build a Segment Tree data structure.

https://en.wikipedia.org/wiki/Segment_tree

Can be used as regular array, but with two important differences:

a) setting item's value is slightly slower. It is O(lg capacity) instead of O(1).

b) user has access to an efficient (O(log segment size)) *reduce* operation which reduces *operation* over a contiguous subsequence of items in the array.

Parameters

- **`apacity`** – (int) Total size of the array - must be a power of two.
- **`operation`** – (lambda obj, obj -> obj) and operation for combining elements (eg. sum, max) must form a mathematical group together with the set of possible values for array elements (i.e. be associative)
- **`neutral_element`** – (obj) neutral element for the operation above. eg. float('-inf') for max and 0 for sum.

`__module__ = 'rlzoo.common.buffer'`

`find_prefixsum_idx`(*prefixsum*)

Find the highest index *i* in the array such that sum(arr[0] + arr[1] + … + arr[i - i]) <= prefixsum

if array values are probabilities, this function allows to sample indexes according to the discrete probability efficiently.

Parameters **`perfixsum`** – (float) upperbound on the sum of array prefix

Returns:

idx: (int) highest index satisfying the prefixsum constraint

`sum`(*start=0*, *end=None*)

Returns arr[start] + … + arr[end]

CHAPTER 19

Distributions

19.1 Distributions for Stochastic Policy in RLzoo

Definition of parametrized distributions. Adapted from openai/baselines

class `rlzoo.common.distributions.`**`Categorical`**(*ndim*, *logits=None*)
    Bases: *`rlzoo.common.distributions.Distribution`*

    Creates a categorical distribution

    `entropy`()
        Calculate the entropy of distribution.

    `get_param`()

    `greedy_sample`()
        Get actions greedily

    `kl`(*logits*)

        Args: logits (tensor): logits variables of another distribution

    `logp`(*x*)
        Calculate log probability of a sample.

    `ndim`

    `neglogp`(*x*)
        Calculate negative log probability of a sample.

    `sample`()
        Sample actions from distribution, using the Gumbel-Softmax trick

    `set_param`(*logits*)

        Args: logits (tensor): logits variables to set

class `rlzoo.common.distributions.`**`DiagGaussian`**(*ndim*, *mean_logstd=None*)
    Bases: *`rlzoo.common.distributions.Distribution`*

Creates a diagonal Gaussian distribution

entropy()
Calculate the entropy of distribution.

get_param()
Get parameters

greedy_sample()
Get actions greedily/deterministically

kl(*mean_logstd*)

Args: mean_logstd (tensor): mean and logstd of another distribution

logp(*x*)
Calculate log probability of a sample.

ndim

neglogp(*x*)
Calculate negative log probability of a sample.

sample()
Get actions in deterministic or stochastic manner

set_param(*mean_logstd*)

Args: mean_logstd (tensor): mean and log std

class rlzoo.common.distributions.**Distribution**
Bases: `object`

A particular probability distribution

entropy()
Calculate the entropy of distribution.

kl(**parameters*)
Calculate Kullback–Leibler divergence

logp(*x*)
Calculate log probability of a sample.

neglogp(*x*)
Calculate negative log probability of a sample.

sample(**args*, ***kwargs*)
Sampling from distribution. Allow explore parameters.

set_param(**args*, ***kwargs*)

rlzoo.common.distributions.**expand_dims**(*func*)

rlzoo.common.distributions.**make_dist**(*ac_space*)
Get distribution based on action space

Parameters **ac_space** – gym.spaces.Space

CHAPTER 20

Environment Wrappers

20.1 Environment Wrappers in RLzoo

Env wrappers Most common wrappers can be checked from following links for usage:

https://pypi.org/project/gym-vec-env

https://github.com/openai/baselines/blob/master/baselines/common/wrappers.py

`rlzoo.common.env_wrappers.`**`build_env`**(*env_id*, *env_type*, *vectorized=False*, *seed=0*, *reward_shaping=None*, *nenv=1*, ***kwargs*)

Build env based on options

Parameters

- **`env_id`** – (str) environment id
- **`env_type`** – (str) atari, classic_control, box2d
- **`vectorized`** – (bool) whether sampling parrallel
- **`seed`** – (int) random seed for env
- **`reward_shaping`** – (callable) callable function for reward shaping
- **`nenv`** – (int) how many processes will be used in sampling
- **`kwargs`** – (dict)
- **`max_episode_steps`** – (int) the maximum episode steps

`class` `rlzoo.common.env_wrappers.`**`TimeLimit`**(*env*, *max_episode_steps=None*)

Bases: `gym.core.Wrapper`

`reset`(***kwargs*)

Resets the state of the environment and returns an initial observation.

Returns: observation (object): the initial observation.

step (*ac*)
Run one timestep of the environment's dynamics. When end of episode is reached, you are responsible for calling *reset()* to reset this environment's state.

Accepts an action and returns a tuple (observation, reward, done, info).

Args: action (object): an action provided by the agent

Returns: observation (object): agent's observation of the current environment reward (float) : amount of reward returned after previous action done (bool): whether the episode has ended, in which case further step() calls will return undefined results info (dict): contains auxiliary diagnostic information (helpful for debugging, and sometimes learning)

class rlzoo.common.env_wrappers.**NoopResetEnv** (*env*, *noop_max=30*)
Bases: gym.core.Wrapper

reset (***kwargs*)
Do no-op action for a number of steps in [1, noop_max].

step (*ac*)
Run one timestep of the environment's dynamics. When end of episode is reached, you are responsible for calling *reset()* to reset this environment's state.

Accepts an action and returns a tuple (observation, reward, done, info).

Args: action (object): an action provided by the agent

Returns: observation (object): agent's observation of the current environment reward (float) : amount of reward returned after previous action done (bool): whether the episode has ended, in which case further step() calls will return undefined results info (dict): contains auxiliary diagnostic information (helpful for debugging, and sometimes learning)

class rlzoo.common.env_wrappers.**FireResetEnv** (*env*)
Bases: gym.core.Wrapper

reset (***kwargs*)
Resets the state of the environment and returns an initial observation.

Returns: observation (object): the initial observation.

step (*ac*)
Run one timestep of the environment's dynamics. When end of episode is reached, you are responsible for calling *reset()* to reset this environment's state.

Accepts an action and returns a tuple (observation, reward, done, info).

Args: action (object): an action provided by the agent

Returns: observation (object): agent's observation of the current environment reward (float) : amount of reward returned after previous action done (bool): whether the episode has ended, in which case further step() calls will return undefined results info (dict): contains auxiliary diagnostic information (helpful for debugging, and sometimes learning)

class rlzoo.common.env_wrappers.**EpisodicLifeEnv** (*env*)
Bases: gym.core.Wrapper

reset (***kwargs*)
Reset only when lives are exhausted. This way all states are still reachable even though lives are episodic, and the learner need not know about any of this behind-the-scenes.

step (*action*)
Run one timestep of the environment's dynamics. When end of episode is reached, you are responsible for calling *reset()* to reset this environment's state.

Accepts an action and returns a tuple (observation, reward, done, info).

Args: action (object): an action provided by the agent

Returns: observation (object): agent's observation of the current environment reward (float) : amount of reward returned after previous action done (bool): whether the episode has ended, in which case further step() calls will return undefined results info (dict): contains auxiliary diagnostic information (helpful for debugging, and sometimes learning)

class `rlzoo.common.env_wrappers.`**`MaxAndSkipEnv`**(*env*, *skip=4*)
Bases: `gym.core.Wrapper`

`reset`(***kwargs*)
Resets the state of the environment and returns an initial observation.

Returns: observation (object): the initial observation.

`step`(*action*)
Repeat action, sum reward, and max over last observations.

class `rlzoo.common.env_wrappers.`**`ClipRewardEnv`**(*env*)
Bases: `gym.core.RewardWrapper`

`reward`(*reward*)
Bin reward to {+1, 0, -1} by its sign.

class `rlzoo.common.env_wrappers.`**`WarpFrame`**(*env*, *width=84*, *height=84*, *grayscale=True*)
Bases: `gym.core.ObservationWrapper`

`observation`(*frame*)

class `rlzoo.common.env_wrappers.`**`FrameStack`**(*env*, *k*)
Bases: `gym.core.Wrapper`

`reset`()
Resets the state of the environment and returns an initial observation.

Returns: observation (object): the initial observation.

`step`(*action*)
Run one timestep of the environment's dynamics. When end of episode is reached, you are responsible for calling *reset()* to reset this environment's state.

Accepts an action and returns a tuple (observation, reward, done, info).

Args: action (object): an action provided by the agent

Returns: observation (object): agent's observation of the current environment reward (float) : amount of reward returned after previous action done (bool): whether the episode has ended, in which case further step() calls will return undefined results info (dict): contains auxiliary diagnostic information (helpful for debugging, and sometimes learning)

class `rlzoo.common.env_wrappers.`**`LazyFrames`**(*frames*)
Bases: `object`

class `rlzoo.common.env_wrappers.`**`RewardShaping`**(*env*, *func*)
Bases: `gym.core.RewardWrapper`

Shaping the reward For reward scale, func can be *lambda r: r * scale*

`reward`(*reward*)

class `rlzoo.common.env_wrappers.`**`SubprocVecEnv`**(*env_fns*)
Bases: `object`

close()

reset()

Reset all the environments and return an array of observations, or a tuple of observation arrays. If step_async is still doing work, that work will be cancelled and step_wait() should not be called until step_async() is invoked again.

step(*actions*)

class `rlzoo.common.env_wrappers.`**VecFrameStack**(*env*, *k*)

Bases: `object`

reset()

step(*action*)

class `rlzoo.common.env_wrappers.`**Monitor**(*env*, *info_keywords=None*)

Bases: `gym.core.Wrapper`

reset(***kwargs*)

Resets the state of the environment and returns an initial observation.

Returns: observation (object): the initial observation.

step(*action*)

Run one timestep of the environment's dynamics. When end of episode is reached, you are responsible for calling *reset()* to reset this environment's state.

Accepts an action and returns a tuple (observation, reward, done, info).

Args: action (object): an action provided by the agent

Returns: observation (object): agent's observation of the current environment reward (float) : amount of reward returned after previous action done (bool): whether the episode has ended, in which case further step() calls will return undefined results info (dict): contains auxiliary diagnostic information (helpful for debugging, and sometimes learning)

class `rlzoo.common.env_wrappers.`**NormalizedActions**(*env*)

Bases: `gym.core.ActionWrapper`

class `rlzoo.common.env_wrappers.`**DmObsTrans**(*env*)

Bases: `gym.core.Wrapper`

Observation process for DeepMind Control Suite environments

reset(***kwargs*)

Resets the state of the environment and returns an initial observation.

Returns: observation (object): the initial observation.

step(*ac*)

Run one timestep of the environment's dynamics. When end of episode is reached, you are responsible for calling *reset()* to reset this environment's state.

Accepts an action and returns a tuple (observation, reward, done, info).

Args: action (object): an action provided by the agent

Returns: observation (object): agent's observation of the current environment reward (float) : amount of reward returned after previous action done (bool): whether the episode has ended, in which case further step() calls will return undefined results info (dict): contains auxiliary diagnostic information (helpful for debugging, and sometimes learning)

CHAPTER 21

Environment List

21.1 List of Supported Environments in RLzoo

`rlzoo.common.env_list.get_envlist(env_type)`

get list of env names wrt the type of env

```
all_env_list = {
    ## Gym
    # Atari
    'atari': ['AirRaid-v0',
              'AirRaid-v4',
              'AirRaidDeterministic-v0',
              'AirRaidDeterministic-v4',
              'AirRaidNoFrameskip-v0',
              'AirRaidNoFrameskip-v4',
              'AirRaid-ram-v0',
              'AirRaid-ram-v4',
              'AirRaid-ramDeterministic-v0',
              'AirRaid-ramDeterministic-v4',
              'AirRaid-ramNoFrameskip-v0',
              'AirRaid-ramNoFrameskip-v4',
              'Alien-v0',
              'Alien-v4',
              'AlienDeterministic-v0',
              'AlienDeterministic-v4',
              'AlienNoFrameskip-v0',
              'AlienNoFrameskip-v4',
              'Alien-ram-v0',
              'Alien-ram-v4',
              'Alien-ramDeterministic-v0',
              'Alien-ramDeterministic-v4',
              'Alien-ramNoFrameskip-v0',
              'Alien-ramNoFrameskip-v4',
              'Amidar-v0',
```

(continues on next page)

(continued from previous page)

```
                'Amidar-v4',
                'AmidarDeterministic-v0',
                'AmidarDeterministic-v4',
                'AmidarNoFrameskip-v0',
                'AmidarNoFrameskip-v4',
                'Amidar-ram-v0',
                'Amidar-ram-v4',
                'Amidar-ramDeterministic-v0',
                'Amidar-ramDeterministic-v4',
                'Amidar-ramNoFrameskip-v0',
                'Amidar-ramNoFrameskip-v4',
                'Assault-v0',
                'Assault-v4',
                'AssaultDeterministic-v0',
                'AssaultDeterministic-v4',
                'AssaultNoFrameskip-v0',
                'AssaultNoFrameskip-v4',
                'Assault-ram-v0',
                'Assault-ram-v4',
                'Assault-ramDeterministic-v0',
                'Assault-ramDeterministic-v4',
                'Assault-ramNoFrameskip-v0',
                'Assault-ramNoFrameskip-v4',
                'Asterix-v0',
                'Asterix-v4',
                'AsterixDeterministic-v0',
                'AsterixDeterministic-v4',
                'AsterixNoFrameskip-v0',
                'AsterixNoFrameskip-v4',
                'Asterix-ram-v0',
                'Asterix-ram-v4',
                'Asterix-ramDeterministic-v0',
                'Asterix-ramDeterministic-v4',
                'Asterix-ramNoFrameskip-v0',
                'Asterix-ramNoFrameskip-v4',
                'Asteroids-v0',
                'Asteroids-v4',
                'AsteroidsDeterministic-v0',
                'AsteroidsDeterministic-v4',
                'AsteroidsNoFrameskip-v0',
                'AsteroidsNoFrameskip-v4',
                'Asteroids-ram-v0',
                'Asteroids-ram-v4',
                'Asteroids-ramDeterministic-v0',
                'Asteroids-ramDeterministic-v4',
                'Asteroids-ramNoFrameskip-v0',
                'Asteroids-ramNoFrameskip-v4',
                'Atlantis-v0',
                'Atlantis-v4',
                'AtlantisDeterministic-v0',
                'AtlantisDeterministic-v4',
                'AtlantisNoFrameskip-v0',
                'AtlantisNoFrameskip-v4',
                'Atlantis-ram-v0',
                'Atlantis-ram-v4',
                'Atlantis-ramDeterministic-v0',
                'Atlantis-ramDeterministic-v4',
```

(continues on next page)

(continued from previous page)

```
                'Atlantis-ramNoFrameskip-v0',
                'Atlantis-ramNoFrameskip-v4',
                'BankHeist-v0',
                'BankHeist-v4',
                'BankHeistDeterministic-v0',
                'BankHeistDeterministic-v4',
                'BankHeistNoFrameskip-v0',
                'BankHeistNoFrameskip-v4',
                'BankHeist-ram-v0',
                'BankHeist-ram-v4',
                'BankHeist-ramDeterministic-v0',
                'BankHeist-ramDeterministic-v4',
                'BankHeist-ramNoFrameskip-v0',
                'BankHeist-ramNoFrameskip-v4',
                'BattleZone-v0',
                'BattleZone-v4',
                'BattleZoneDeterministic-v0',
                'BattleZoneDeterministic-v4',
                'BattleZoneNoFrameskip-v0',
                'BattleZoneNoFrameskip-v4',
                'BattleZone-ram-v0',
                'BattleZone-ram-v4',
                'BattleZone-ramDeterministic-v0',
                'BattleZone-ramDeterministic-v4',
                'BattleZone-ramNoFrameskip-v0',
                'BattleZone-ramNoFrameskip-v4',
                'BeamRider-v0',
                'BeamRider-v4',
                'BeamRiderDeterministic-v0',
                'BeamRiderDeterministic-v4',
                'BeamRiderNoFrameskip-v0',
                'BeamRiderNoFrameskip-v4',
                'BeamRider-ram-v0',
                'BeamRider-ram-v4',
                'BeamRider-ramDeterministic-v0',
                'BeamRider-ramDeterministic-v4',
                'BeamRider-ramNoFrameskip-v0',
                'BeamRider-ramNoFrameskip-v4',
                'Berzerk-v0',
                'Berzerk-v4',
                'BerzerkDeterministic-v0',
                'BerzerkDeterministic-v4',
                'BerzerkNoFrameskip-v0',
                'BerzerkNoFrameskip-v4',
                'Berzerk-ram-v0',
                'Berzerk-ram-v4',
                'Berzerk-ramDeterministic-v0',
                'Berzerk-ramDeterministic-v4',
                'Berzerk-ramNoFrameskip-v0',
                'Berzerk-ramNoFrameskip-v4',
                'Bowling-v0',
                'Bowling-v4',
                'BowlingDeterministic-v0',
                'BowlingDeterministic-v4',
                'BowlingNoFrameskip-v0',
                'BowlingNoFrameskip-v4',
                'Bowling-ram-v0',
```

(continues on next page)

(continued from previous page)

```
                'Bowling-ram-v4',
                'Bowling-ramDeterministic-v0',
                'Bowling-ramDeterministic-v4',
                'Bowling-ramNoFrameskip-v0',
                'Bowling-ramNoFrameskip-v4',
                'Boxing-v0',
                'Boxing-v4',
                'BoxingDeterministic-v0',
                'BoxingDeterministic-v4',
                'BoxingNoFrameskip-v0',
                'BoxingNoFrameskip-v4',
                'Boxing-ram-v0',
                'Boxing-ram-v4',
                'Boxing-ramDeterministic-v0',
                'Boxing-ramDeterministic-v4',
                'Boxing-ramNoFrameskip-v0',
                'Boxing-ramNoFrameskip-v4',
                'Breakout-v0',
                'Breakout-v4',
                'BreakoutDeterministic-v0',
                'BreakoutDeterministic-v4',
                'BreakoutNoFrameskip-v0',
                'BreakoutNoFrameskip-v4',
                'Breakout-ram-v0',
                'Breakout-ram-v4',
                'Breakout-ramDeterministic-v0',
                'Breakout-ramDeterministic-v4',
                'Breakout-ramNoFrameskip-v0',
                'Breakout-ramNoFrameskip-v4',
                'Carnival-v0',
                'Carnival-v4',
                'CarnivalDeterministic-v0',
                'CarnivalDeterministic-v4',
                'CarnivalNoFrameskip-v0',
                'CarnivalNoFrameskip-v4',
                'Carnival-ram-v0',
                'Carnival-ram-v4',
                'Carnival-ramDeterministic-v0',
                'Carnival-ramDeterministic-v4',
                'Carnival-ramNoFrameskip-v0',
                'Carnival-ramNoFrameskip-v4',
                'Centipede-v0',
                'Centipede-v4',
                'CentipedeDeterministic-v0',
                'CentipedeDeterministic-v4',
                'CentipedeNoFrameskip-v0',
                'CentipedeNoFrameskip-v4',
                'Centipede-ram-v0',
                'Centipede-ram-v4',
                'Centipede-ramDeterministic-v0',
                'Centipede-ramDeterministic-v4',
                'Centipede-ramNoFrameskip-v0',
                'Centipede-ramNoFrameskip-v4',
                'ChopperCommand-v0',
                'ChopperCommand-v4',
                'ChopperCommandDeterministic-v0',
                'ChopperCommandDeterministic-v4',
```

(continues on next page)

(continued from previous page)

```
                'ChopperCommandNoFrameskip-v0',
                'ChopperCommandNoFrameskip-v4',
                'ChopperCommand-ram-v0',
                'ChopperCommand-ram-v4',
                'ChopperCommand-ramDeterministic-v0',
                'ChopperCommand-ramDeterministic-v4',
                'ChopperCommand-ramNoFrameskip-v0',
                'ChopperCommand-ramNoFrameskip-v4',
                'CrazyClimber-v0',
                'CrazyClimber-v4',
                'CrazyClimberDeterministic-v0',
                'CrazyClimberDeterministic-v4',
                'CrazyClimberNoFrameskip-v0',
                'CrazyClimberNoFrameskip-v4',
                'CrazyClimber-ram-v0',
                'CrazyClimber-ram-v4',
                'CrazyClimber-ramDeterministic-v0',
                'CrazyClimber-ramDeterministic-v4',
                'CrazyClimber-ramNoFrameskip-v0',
                'CrazyClimber-ramNoFrameskip-v4',
                'DemonAttack-v0',
                'DemonAttack-v4',
                'DemonAttackDeterministic-v0',
                'DemonAttackDeterministic-v4',
                'DemonAttackNoFrameskip-v0',
                'DemonAttackNoFrameskip-v4',
                'DemonAttack-ram-v0',
                'DemonAttack-ram-v4',
                'DemonAttack-ramDeterministic-v0',
                'DemonAttack-ramDeterministic-v4',
                'DemonAttack-ramNoFrameskip-v0',
                'DemonAttack-ramNoFrameskip-v4',
                'DoubleDunk-v0',
                'DoubleDunk-v4',
                'DoubleDunkDeterministic-v0',
                'DoubleDunkDeterministic-v4',
                'DoubleDunkNoFrameskip-v0',
                'DoubleDunkNoFrameskip-v4',
                'DoubleDunk-ram-v0',
                'DoubleDunk-ram-v4',
                'DoubleDunk-ramDeterministic-v0',
                'DoubleDunk-ramDeterministic-v4',
                'DoubleDunk-ramNoFrameskip-v0',
                'DoubleDunk-ramNoFrameskip-v4',
                'ElevatorAction-v0',
                'ElevatorAction-v4',
                'ElevatorActionDeterministic-v0',
                'ElevatorActionDeterministic-v4',
                'ElevatorActionNoFrameskip-v0',
                'ElevatorActionNoFrameskip-v4',
                'ElevatorAction-ram-v0',
                'ElevatorAction-ram-v4',
                'ElevatorAction-ramDeterministic-v0',
                'ElevatorAction-ramDeterministic-v4',
                'ElevatorAction-ramNoFrameskip-v0',
                'ElevatorAction-ramNoFrameskip-v4',
                'Enduro-v0',
```

(continues on next page)

(continued from previous page)

```
                'Enduro-v4',
                'EnduroDeterministic-v0',
                'EnduroDeterministic-v4',
                'EnduroNoFrameskip-v0',
                'EnduroNoFrameskip-v4',
                'Enduro-ram-v0',
                'Enduro-ram-v4',
                'Enduro-ramDeterministic-v0',
                'Enduro-ramDeterministic-v4',
                'Enduro-ramNoFrameskip-v0',
                'Enduro-ramNoFrameskip-v4',
                'FishingDerby-v0',
                'FishingDerby-v4',
                'FishingDerbyDeterministic-v0',
                'FishingDerbyDeterministic-v4',
                'FishingDerbyNoFrameskip-v0',
                'FishingDerbyNoFrameskip-v4',
                'FishingDerby-ram-v0',
                'FishingDerby-ram-v4',
                'FishingDerby-ramDeterministic-v0',
                'FishingDerby-ramDeterministic-v4',
                'FishingDerby-ramNoFrameskip-v0',
                'FishingDerby-ramNoFrameskip-v4',
                'Freeway-v0',
                'Freeway-v4',
                'FreewayDeterministic-v0',
                'FreewayDeterministic-v4',
                'FreewayNoFrameskip-v0',
                'FreewayNoFrameskip-v4',
                'Freeway-ram-v0',
                'Freeway-ram-v4',
                'Freeway-ramDeterministic-v0',
                'Freeway-ramDeterministic-v4',
                'Freeway-ramNoFrameskip-v0',
                'Freeway-ramNoFrameskip-v4',
                'Frostbite-v0',
                'Frostbite-v4',
                'FrostbiteDeterministic-v0',
                'FrostbiteDeterministic-v4',
                'FrostbiteNoFrameskip-v0',
                'FrostbiteNoFrameskip-v4',
                'Frostbite-ram-v0',
                'Frostbite-ram-v4',
                'Frostbite-ramDeterministic-v0',
                'Frostbite-ramDeterministic-v4',
                'Frostbite-ramNoFrameskip-v0',
                'Frostbite-ramNoFrameskip-v4',
                'Gopher-v0',
                'Gopher-v4',
                'GopherDeterministic-v0',
                'GopherDeterministic-v4',
                'GopherNoFrameskip-v0',
                'GopherNoFrameskip-v4',
                'Gopher-ram-v0',
                'Gopher-ram-v4',
                'Gopher-ramDeterministic-v0',
                'Gopher-ramDeterministic-v4',
```

(continues on next page)

(continued from previous page)

```
                'Gopher-ramNoFrameskip-v0',
                'Gopher-ramNoFrameskip-v4',
                'Gravitar-v0',
                'Gravitar-v4',
                'GravitarDeterministic-v0',
                'GravitarDeterministic-v4',
                'GravitarNoFrameskip-v0',
                'GravitarNoFrameskip-v4',
                'Gravitar-ram-v0',
                'Gravitar-ram-v4',
                'Gravitar-ramDeterministic-v0',
                'Gravitar-ramDeterministic-v4',
                'Gravitar-ramNoFrameskip-v0',
                'Gravitar-ramNoFrameskip-v4',
                'Hero-v0',
                'Hero-v4',
                'HeroDeterministic-v0',
                'HeroDeterministic-v4',
                'HeroNoFrameskip-v0',
                'HeroNoFrameskip-v4',
                'Hero-ram-v0',
                'Hero-ram-v4',
                'Hero-ramDeterministic-v0',
                'Hero-ramDeterministic-v4',
                'Hero-ramNoFrameskip-v0',
                'Hero-ramNoFrameskip-v4',
                'IceHockey-v0',
                'IceHockey-v4',
                'IceHockeyDeterministic-v0',
                'IceHockeyDeterministic-v4',
                'IceHockeyNoFrameskip-v0',
                'IceHockeyNoFrameskip-v4',
                'IceHockey-ram-v0',
                'IceHockey-ram-v4',
                'IceHockey-ramDeterministic-v0',
                'IceHockey-ramDeterministic-v4',
                'IceHockey-ramNoFrameskip-v0',
                'IceHockey-ramNoFrameskip-v4',
                'Jamesbond-v0',
                'Jamesbond-v4',
                'JamesbondDeterministic-v0',
                'JamesbondDeterministic-v4',
                'JamesbondNoFrameskip-v0',
                'JamesbondNoFrameskip-v4',
                'Jamesbond-ram-v0',
                'Jamesbond-ram-v4',
                'Jamesbond-ramDeterministic-v0',
                'Jamesbond-ramDeterministic-v4',
                'Jamesbond-ramNoFrameskip-v0',
                'Jamesbond-ramNoFrameskip-v4',
                'JourneyEscape-v0',
                'JourneyEscape-v4',
                'JourneyEscapeDeterministic-v0',
                'JourneyEscapeDeterministic-v4',
                'JourneyEscapeNoFrameskip-v0',
                'JourneyEscapeNoFrameskip-v4',
                'JourneyEscape-ram-v0',
```

(continues on next page)

(continued from previous page)

```
                  'JourneyEscape-ram-v4',
                  'JourneyEscape-ramDeterministic-v0',
                  'JourneyEscape-ramDeterministic-v4',
                  'JourneyEscape-ramNoFrameskip-v0',
                  'JourneyEscape-ramNoFrameskip-v4',
                  'Kangaroo-v0',
                  'Kangaroo-v4',
                  'KangarooDeterministic-v0',
                  'KangarooDeterministic-v4',
                  'KangarooNoFrameskip-v0',
                  'KangarooNoFrameskip-v4',
                  'Kangaroo-ram-v0',
                  'Kangaroo-ram-v4',
                  'Kangaroo-ramDeterministic-v0',
                  'Kangaroo-ramDeterministic-v4',
                  'Kangaroo-ramNoFrameskip-v0',
                  'Kangaroo-ramNoFrameskip-v4',
                  'Krull-v0',
                  'Krull-v4',
                  'KrullDeterministic-v0',
                  'KrullDeterministic-v4',
                  'KrullNoFrameskip-v0',
                  'KrullNoFrameskip-v4',
                  'Krull-ram-v0',
                  'Krull-ram-v4',
                  'Krull-ramDeterministic-v0',
                  'Krull-ramDeterministic-v4',
                  'Krull-ramNoFrameskip-v0',
                  'Krull-ramNoFrameskip-v4',
                  'KungFuMaster-v0',
                  'KungFuMaster-v4',
                  'KungFuMasterDeterministic-v0',
                  'KungFuMasterDeterministic-v4',
                  'KungFuMasterNoFrameskip-v0',
                  'KungFuMasterNoFrameskip-v4',
                  'KungFuMaster-ram-v0',
                  'KungFuMaster-ram-v4',
                  'KungFuMaster-ramDeterministic-v0',
                  'KungFuMaster-ramDeterministic-v4',
                  'KungFuMaster-ramNoFrameskip-v0',
                  'KungFuMaster-ramNoFrameskip-v4',
                  'MontezumaRevenge-v0',
                  'MontezumaRevenge-v4',
                  'MontezumaRevengeDeterministic-v0',
                  'MontezumaRevengeDeterministic-v4',
                  'MontezumaRevengeNoFrameskip-v0',
                  'MontezumaRevengeNoFrameskip-v4',
                  'MontezumaRevenge-ram-v0',
                  'MontezumaRevenge-ram-v4',
                  'MontezumaRevenge-ramDeterministic-v0',
                  'MontezumaRevenge-ramDeterministic-v4',
                  'MontezumaRevenge-ramNoFrameskip-v0',
                  'MontezumaRevenge-ramNoFrameskip-v4',
                  'MsPacman-v0',
                  'MsPacman-v4',
                  'MsPacmanDeterministic-v0',
                  'MsPacmanDeterministic-v4',
```

(continues on next page)

(continued from previous page)

```
                'MsPacmanNoFrameskip-v0',
                'MsPacmanNoFrameskip-v4',
                'MsPacman-ram-v0',
                'MsPacman-ram-v4',
                'MsPacman-ramDeterministic-v0',
                'MsPacman-ramDeterministic-v4',
                'MsPacman-ramNoFrameskip-v0',
                'MsPacman-ramNoFrameskip-v4',
                'NameThisGame-v0',
                'NameThisGame-v4',
                'NameThisGameDeterministic-v0',
                'NameThisGameDeterministic-v4',
                'NameThisGameNoFrameskip-v0',
                'NameThisGameNoFrameskip-v4',
                'NameThisGame-ram-v0',
                'NameThisGame-ram-v4',
                'NameThisGame-ramDeterministic-v0',
                'NameThisGame-ramDeterministic-v4',
                'NameThisGame-ramNoFrameskip-v0',
                'NameThisGame-ramNoFrameskip-v4',
                'Phoenix-v0',
                'Phoenix-v4',
                'PhoenixDeterministic-v0',
                'PhoenixDeterministic-v4',
                'PhoenixNoFrameskip-v0',
                'PhoenixNoFrameskip-v4',
                'Phoenix-ram-v0',
                'Phoenix-ram-v4',
                'Phoenix-ramDeterministic-v0',
                'Phoenix-ramDeterministic-v4',
                'Phoenix-ramNoFrameskip-v0',
                'Phoenix-ramNoFrameskip-v4',
                'Pitfall-v0',
                'Pitfall-v4',
                'PitfallDeterministic-v0',
                'PitfallDeterministic-v4',
                'PitfallNoFrameskip-v0',
                'PitfallNoFrameskip-v4',
                'Pitfall-ram-v0',
                'Pitfall-ram-v4',
                'Pitfall-ramDeterministic-v0',
                'Pitfall-ramDeterministic-v4',
                'Pitfall-ramNoFrameskip-v0',
                'Pitfall-ramNoFrameskip-v4',
                'Pong-v0',
                'Pong-v4',
                'PongDeterministic-v0',
                'PongDeterministic-v4',
                'PongNoFrameskip-v0',
                'PongNoFrameskip-v4',
                'Pong-ram-v0',
                'Pong-ram-v4',
                'Pong-ramDeterministic-v0',
                'Pong-ramDeterministic-v4',
                'Pong-ramNoFrameskip-v0',
                'Pong-ramNoFrameskip-v4',
                'Pooyan-v0',
```

(continues on next page)

(continued from previous page)

```
                    'Pooyan-v4',
                    'PooyanDeterministic-v0',
                    'PooyanDeterministic-v4',
                    'PooyanNoFrameskip-v0',
                    'PooyanNoFrameskip-v4',
                    'Pooyan-ram-v0',
                    'Pooyan-ram-v4',
                    'Pooyan-ramDeterministic-v0',
                    'Pooyan-ramDeterministic-v4',
                    'Pooyan-ramNoFrameskip-v0',
                    'Pooyan-ramNoFrameskip-v4',
                    'PrivateEye-v0',
                    'PrivateEye-v4',
                    'PrivateEyeDeterministic-v0',
                    'PrivateEyeDeterministic-v4',
                    'PrivateEyeNoFrameskip-v0',
                    'PrivateEyeNoFrameskip-v4',
                    'PrivateEye-ram-v0',
                    'PrivateEye-ram-v4',
                    'PrivateEye-ramDeterministic-v0',
                    'PrivateEye-ramDeterministic-v4',
                    'PrivateEye-ramNoFrameskip-v0',
                    'PrivateEye-ramNoFrameskip-v4',
                    'Qbert-v0',
                    'Qbert-v4',
                    'QbertDeterministic-v0',
                    'QbertDeterministic-v4',
                    'QbertNoFrameskip-v0',
                    'QbertNoFrameskip-v4',
                    'Qbert-ram-v0',
                    'Qbert-ram-v4',
                    'Qbert-ramDeterministic-v0',
                    'Qbert-ramDeterministic-v4',
                    'Qbert-ramNoFrameskip-v0',
                    'Qbert-ramNoFrameskip-v4',
                    'Riverraid-v0',
                    'Riverraid-v4',
                    'RiverraidDeterministic-v0',
                    'RiverraidDeterministic-v4',
                    'RiverraidNoFrameskip-v0',
                    'RiverraidNoFrameskip-v4',
                    'Riverraid-ram-v0',
                    'Riverraid-ram-v4',
                    'Riverraid-ramDeterministic-v0',
                    'Riverraid-ramDeterministic-v4',
                    'Riverraid-ramNoFrameskip-v0',
                    'Riverraid-ramNoFrameskip-v4',
                    'RoadRunner-v0',
                    'RoadRunner-v4',
                    'RoadRunnerDeterministic-v0',
                    'RoadRunnerDeterministic-v4',
                    'RoadRunnerNoFrameskip-v0',
                    'RoadRunnerNoFrameskip-v4',
                    'RoadRunner-ram-v0',
                    'RoadRunner-ram-v4',
                    'RoadRunner-ramDeterministic-v0',
                    'RoadRunner-ramDeterministic-v4',
```

(continues on next page)

(continued from previous page)

```
                    'RoadRunner-ramNoFrameskip-v0',
                    'RoadRunner-ramNoFrameskip-v4',
                    'Robotank-v0',
                    'Robotank-v4',
                    'RobotankDeterministic-v0',
                    'RobotankDeterministic-v4',
                    'RobotankNoFrameskip-v0',
                    'RobotankNoFrameskip-v4',
                    'Robotank-ram-v0',
                    'Robotank-ram-v4',
                    'Robotank-ramDeterministic-v0',
                    'Robotank-ramDeterministic-v4',
                    'Robotank-ramNoFrameskip-v0',
                    'Robotank-ramNoFrameskip-v4',
                    'Seaquest-v0',
                    'Seaquest-v4',
                    'SeaquestDeterministic-v0',
                    'SeaquestDeterministic-v4',
                    'SeaquestNoFrameskip-v0',
                    'SeaquestNoFrameskip-v4',
                    'Seaquest-ram-v0',
                    'Seaquest-ram-v4',
                    'Seaquest-ramDeterministic-v0',
                    'Seaquest-ramDeterministic-v4',
                    'Seaquest-ramNoFrameskip-v0',
                    'Seaquest-ramNoFrameskip-v4',
                    'Skiing-v0',
                    'Skiing-v4',
                    'SkiingDeterministic-v0',
                    'SkiingDeterministic-v4',
                    'SkiingNoFrameskip-v0',
                    'SkiingNoFrameskip-v4',
                    'Skiing-ram-v0',
                    'Skiing-ram-v4',
                    'Skiing-ramDeterministic-v0',
                    'Skiing-ramDeterministic-v4',
                    'Skiing-ramNoFrameskip-v0',
                    'Skiing-ramNoFrameskip-v4',
                    'Solaris-v0',
                    'Solaris-v4',
                    'SolarisDeterministic-v0',
                    'SolarisDeterministic-v4',
                    'SolarisNoFrameskip-v0',
                    'SolarisNoFrameskip-v4',
                    'Solaris-ram-v0',
                    'Solaris-ram-v4',
                    'Solaris-ramDeterministic-v0',
                    'Solaris-ramDeterministic-v4',
                    'Solaris-ramNoFrameskip-v0',
                    'Solaris-ramNoFrameskip-v4',
                    'SpaceInvaders-v0',
                    'SpaceInvaders-v4',
                    'SpaceInvadersDeterministic-v0',
                    'SpaceInvadersDeterministic-v4',
                    'SpaceInvadersNoFrameskip-v0',
                    'SpaceInvadersNoFrameskip-v4',
                    'SpaceInvaders-ram-v0',
```

(continues on next page)

(continued from previous page)

```
                    'SpaceInvaders-ram-v4',
                    'SpaceInvaders-ramDeterministic-v0',
                    'SpaceInvaders-ramDeterministic-v4',
                    'SpaceInvaders-ramNoFrameskip-v0',
                    'SpaceInvaders-ramNoFrameskip-v4',
                    'StarGunner-v0',
                    'StarGunner-v4',
                    'StarGunnerDeterministic-v0',
                    'StarGunnerDeterministic-v4',
                    'StarGunnerNoFrameskip-v0',
                    'StarGunnerNoFrameskip-v4',
                    'StarGunner-ram-v0',
                    'StarGunner-ram-v4',
                    'StarGunner-ramDeterministic-v0',
                    'StarGunner-ramDeterministic-v4',
                    'StarGunner-ramNoFrameskip-v0',
                    'StarGunner-ramNoFrameskip-v4',
                    'Tennis-v0',
                    'Tennis-v4',
                    'TennisDeterministic-v0',
                    'TennisDeterministic-v4',
                    'TennisNoFrameskip-v0',
                    'TennisNoFrameskip-v4',
                    'Tennis-ram-v0',
                    'Tennis-ram-v4',
                    'Tennis-ramDeterministic-v0',
                    'Tennis-ramDeterministic-v4',
                    'Tennis-ramNoFrameskip-v0',
                    'Tennis-ramNoFrameskip-v4',
                    'TimePilot-v0',
                    'TimePilot-v4',
                    'TimePilotDeterministic-v0',
                    'TimePilotDeterministic-v4',
                    'TimePilotNoFrameskip-v0',
                    'TimePilotNoFrameskip-v4',
                    'TimePilot-ram-v0',
                    'TimePilot-ram-v4',
                    'TimePilot-ramDeterministic-v0',
                    'TimePilot-ramDeterministic-v4',
                    'TimePilot-ramNoFrameskip-v0',
                    'TimePilot-ramNoFrameskip-v4',
                    'Tutankham-v0',
                    'Tutankham-v4',
                    'TutankhamDeterministic-v0',
                    'TutankhamDeterministic-v4',
                    'TutankhamNoFrameskip-v0',
                    'TutankhamNoFrameskip-v4',
                    'Tutankham-ram-v0',
                    'Tutankham-ram-v4',
                    'Tutankham-ramDeterministic-v0',
                    'Tutankham-ramDeterministic-v4',
                    'Tutankham-ramNoFrameskip-v0',
                    'Tutankham-ramNoFrameskip-v4',
                    'UpNDown-v0',
                    'UpNDown-v4',
                    'UpNDownDeterministic-v0',
                    'UpNDownDeterministic-v4',
```

(continues on next page)

(continued from previous page)

```
                    'UpNDownNoFrameskip-v0',
                    'UpNDownNoFrameskip-v4',
                    'UpNDown-ram-v0',
                    'UpNDown-ram-v4',
                    'UpNDown-ramDeterministic-v0',
                    'UpNDown-ramDeterministic-v4',
                    'UpNDown-ramNoFrameskip-v0',
                    'UpNDown-ramNoFrameskip-v4',
                    'Venture-v0',
                    'Venture-v4',
                    'VentureDeterministic-v0',
                    'VentureDeterministic-v4',
                    'VentureNoFrameskip-v0',
                    'VentureNoFrameskip-v4',
                    'Venture-ram-v0',
                    'Venture-ram-v4',
                    'Venture-ramDeterministic-v0',
                    'Venture-ramDeterministic-v4',
                    'Venture-ramNoFrameskip-v0',
                    'Venture-ramNoFrameskip-v4',
                    'VideoPinball-v0',
                    'VideoPinball-v4',
                    'VideoPinballDeterministic-v0',
                    'VideoPinballDeterministic-v4',
                    'VideoPinballNoFrameskip-v0',
                    'VideoPinballNoFrameskip-v4',
                    'VideoPinball-ram-v0',
                    'VideoPinball-ram-v4',
                    'VideoPinball-ramDeterministic-v0',
                    'VideoPinball-ramDeterministic-v4',
                    'VideoPinball-ramNoFrameskip-v0',
                    'VideoPinball-ramNoFrameskip-v4',
                    'WizardOfWor-v0',
                    'WizardOfWor-v4',
                    'WizardOfWorDeterministic-v0',
                    'WizardOfWorDeterministic-v4',
                    'WizardOfWorNoFrameskip-v0',
                    'WizardOfWorNoFrameskip-v4',
                    'WizardOfWor-ram-v0',
                    'WizardOfWor-ram-v4',
                    'WizardOfWor-ramDeterministic-v0',
                    'WizardOfWor-ramDeterministic-v4',
                    'WizardOfWor-ramNoFrameskip-v0',
                    'WizardOfWor-ramNoFrameskip-v4',
                    'YarsRevenge-v0',
                    'YarsRevenge-v4',
                    'YarsRevengeDeterministic-v0',
                    'YarsRevengeDeterministic-v4',
                    'YarsRevengeNoFrameskip-v0',
                    'YarsRevengeNoFrameskip-v4',
                    'YarsRevenge-ram-v0',
                    'YarsRevenge-ram-v4',
                    'YarsRevenge-ramDeterministic-v0',
                    'YarsRevenge-ramDeterministic-v4',
                    'YarsRevenge-ramNoFrameskip-v0',
                    'YarsRevenge-ramNoFrameskip-v4',
                    'Zaxxon-v0',
```

(continues on next page)

(continued from previous page)

```
                'Zaxxon-v4',
                'ZaxxonDeterministic-v0',
                'ZaxxonDeterministic-v4',
                'ZaxxonNoFrameskip-v0',
                'ZaxxonNoFrameskip-v4',
                'Zaxxon-ram-v0',
                'Zaxxon-ram-v4',
                'Zaxxon-ramDeterministic-v0',
                'Zaxxon-ramDeterministic-v4',
                'Zaxxon-ramNoFrameskip-v0',
                'Zaxxon-ramNoFrameskip-v4'],

    # Classic control
    'classic_control': [
        'Acrobot-v1',
        'CartPole-v1',
        'CartPole-v0',
        'MountainCar-v0',
        'MountainCarContinuous-v0',
        'Pendulum-v0'
    ],

    # Box2D
    'box2d': [
        'BipedalWalker-v2',
        'BipedalWalkerHardcore-v2',
        'CarRacing-v0',
        'LunarLander-v2',
        'LunarLanderContinuous-v2'
    ],

    # MuJoCo
    'mujoco': [
        'Ant-v2',
        'HalfCheetah-v2',
        'Hopper-v2',
        'Humanoid-v2',
        'HumanoidStandup-v2',
        'InvertedDoublePendulum-v2',
        'InvertedPendulum-v2',
        'Reacher-v2',
        'Swimmer-v2',
        'Walker2d-v2'
    ],

    # Robotics
    'robotics': [
        'FetchPickAndPlace-v1',
        'FetchPush-v1',
        'FetchReach-v1',
        'FetchSlide-v1',
        'HandManipulateBlock-v0',
        'HandManipulateEgg-v0',
        'HandManipulatePen-v0',
        'HandReach-v0'
    ],

```

(continues on next page)

(continued from previous page)

```
    ## Deepmind Control Suite  (need check!)
    'dm_control': [
        'AcrobotSparse-v0',
        'BallincupCatch-v0',
        'CartpoleSwingup-v0',
        'FingerTurn-v0',
        'FishSwim-v0',
        'CheetahRun-v0',
        'HopperHop-v0',
        'HumanoidStand-v0',
        'HumanoidWalk-v0',
        'HumanoidRun-v0',
        'ManipulatorBringball-v0',
        'PendulumSwingup-v0',
        'Pointmass-v0',
        'ReacherHard-v0',
        'Swimmer-v0',
        'WalkerRun-v0'
    ],

    ## RLBench
    'rlbench': [
        'BeatTheBuzz',
        'BlockPyramid',
        'ChangeChannel',
        'ChangeClock',
        'CloseBox',
        'CloseDoor',
        'CloseDrawer',
        'CloseFridge',
        'CloseGrill',
        'CloseJar',
        'CloseLaptopLid',
        'CloseMicrowave',
        'EmptyContainer',
        'EmptyDishwasher',
        'GetIceFromFridge',
        'HangFrameOnHanger',
        'HannoiSquare',
        'HitBallWithQueue',
        'Hockey',
        'InsertUsbInComputer',
        'LampOff',
        'LampOn',
        'LightBulbIn',
        'LightBulbOut',
        'MeatOffGrill',
        'MeatOnGrill',
        'MoveHanger',
        'OpenBox',
        'OpenDoor',
        'OpenDrawer',
        'OpenFridge',
        'OpenGrill',
        'OpenJar',
        'OpenMicrowave',
        'OpenOven',
```

(continues on next page)

(continued from previous page)

```
        'OpenWindow',
        'OpenWineBottle',
        'PhoneOnBase',
        'PickAndLift',
        'PickUpCup',
        'PlaceCups',
        'PlaceHangerOnRack',
        'PlaceShapeInShapeSorter',
        'PlayJenga',
        'PlugChargerInPowerSupply',
        'PourFromCupToCup',
        'PressSwitch',
        'PushButton',
        'PushButtons',
        'PutBooksOnBookshelf',
        'PutBottleInFridge',
        'PutGroceriesInCupboard',
        'PutItemInDrawer',
        'PutKnifeInKnifeBlock',
        'PutKnifeOnChoppingBoard',
        'PutMoneyInSafe',
        'PutPlateInColoredDishRack',
        'PutRubbishInBin',
        'PutShoesInBox',
        'PutToiletRollOnStand',
        'PutTrayInOven',
        'PutUmbrellaInUmbrellaStand',
        'ReachAndDrag',
        'ReachTarget',
        'RemoveCups',
        'ScoopWithSpatula',
        'ScrewNail',
        'SetTheTable',
        'SetupCheckers',
        'SlideBlockToTarget',
        'SlideCabinetOpen',
        'SlideCabinetOpenAndPlaceCups',
        'SolvePuzzle',
        'StackBlocks',
        'StackCups',
        'StackWine',
        'StraightenRope',
        'SweepToDustpan',
        'TakeCupOutFromCabinet',
        'TakeFrameOffHanger',
        'TakeItemOutOfDrawer',
        'TakeLidOffSaucepan',
        'TakeMoneyOutSafe',
        'TakeOffWeighingScales',
        'TakePlateOffColoredDishRack',
        'TakeShoesOutOfBox',
        'TakeToiletRollOffStand',
        'TakeTrayOutOfOven',
        'TakeUmbrellaOutOfUmbrellaStand',
        'TakeUsbOutOfComputer',
        'ToiletSeatDown',
        'ToiletSeatUp',
```

(continues on next page)

(continued from previous page)

```
        'TurnOvenOn',
        'TurnTap',
        'TvOff',
        'TvOn',
        'UnplugCharger',
        'WaterPlants',
        'WeighingScales',
        'WipeDesk'
    ]
}
```

CHAPTER 22

Math Utilities

22.1 Math Utilities in RLzoo

Functions for mathematics utilization.

Requirements tensorflow==2.0.0a0 tensorlayer==2.0.1

`rlzoo.common.math_utils.`**`flatten_dims`**(*shapes*)

CHAPTER 23

Common Utilities

23.1 Common Utilities in RLzoo

Functions for utilization.

Requirements tensorflow==2.0.0a0 tensorlayer==2.0.1

`rlzoo.common.utils.`**`call_default_params`**(*env*, *envtype*, *alg*, *default_seed=True*)
Get the default parameters for training from the default script

`rlzoo.common.utils.`**`get_algorithm_module`**(*algorithm*, *submodule*)
Get algorithm module in the corresponding folder

`rlzoo.common.utils.`**`load_model`**(*model*, *model_name*, *algorithm_name*, *env_name*)
load saved neural network model

Parameters

- **`model`** – tensorlayer.models.Model
- **`model_name`** – string, e.g. 'model_sac_q1'
- **`algorithm_name`** – string, e.g. 'SAC'

`rlzoo.common.utils.`**`make_env`**(*env_id*)

`rlzoo.common.utils.`**`parse_all_args`**(*parser*)
Parse known and unknown args

`rlzoo.common.utils.`**`plot`**(*episode_rewards*, *algorithm_name*, *env_name*)
plot the learning curve, saved as ./img/algorithm_name-env_name.png

Parameters

- **`episode_rewards`** – array of floats
- **`algorithm_name`** – string
- **`env_name`** – string

rlzoo.common.utils.**plot_save_log**(*episode_rewards*, *algorithm_name*, *env_name*)
plot the learning curve, saved as ./img/algorithm_name-env_name.png, and save the rewards log as ./log/algorithm_name-env_name.npy

Parameters

- **episode_rewards** – array of floats
- **algorithm_name** – string
- **env_name** – string

rlzoo.common.utils.**save_model**(*model*, *model_name*, *algorithm_name*, *env_name*)
save trained neural network model

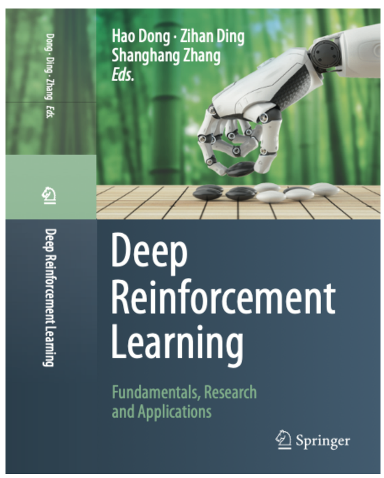
Parameters

- **model** – tensorlayer.models.Model
- **model_name** – string, e.g. 'model_sac_q1'
- **algorithm_name** – string, e.g. 'SAC'

rlzoo.common.utils.**set_seed**(*seed*, *env=None*)
set random seed for reproduciblity

CHAPTER 24

DRL Book



- You can get the free PDF if your institute has Springer license.

Deep reinforcement learning (DRL) relies on the intersection of reinforcement learning (RL) and deep learning (DL). It has been able to solve a wide range of complex decision-making tasks that were previously out of reach for a machine and famously contributed to the success of AlphaGo. Furthermore, it opens up numerous new applications in domains such as healthcare, robotics, smart grids, and finance.

Divided into three main parts, this book provides a comprehensive and self-contained introduction to DRL. The first part introduces the foundations of DL, RL and widely used DRL methods and discusses their implementation. The second part covers selected DRL research topics, which are useful for those wanting to specialize in DRL research. To help readers gain a deep understanding of DRL and quickly apply the techniques in practice, the third part presents mass applications, such as the intelligent transportation system and learning to run, with detailed explanations.

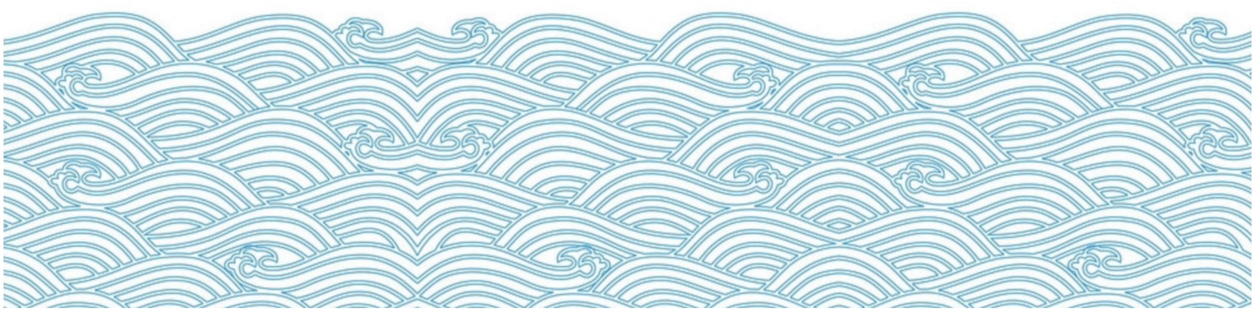
The book is intended for computer science students, both undergraduate and postgraduate, who would like to learn DRL from scratch, practice its implementation, and explore the research topics. This book also appeals to engineers and practitioners who do not have strong machine learning background, but want to quickly understand how DRL works and use the techniques in their applications.

24.1 Editors

- Hao Dong - Peking University
- Zihan Ding - Princeton University
- Shanghang Zhang - University of California, Berkeley

24.2 Authors

- Hao Dong - Peking University
- Zihan Ding - Princeton University
- Shanghang Zhang - University of California, Berkeley
- Hang Yuan - Oxford University
- Hongming Zhang - Peking University
- Jingqing Zhang - Imperial College London
- Yanhua Huang - Xiaohongshu Technology Co.
- Tianyang Yu - Nanchang University
- Huaqing Zhang - Google
- Ruitong Huang - Borealis AI



CHAPTER 25

DRL Tutorial

Different from RLzoo for simple usage with **high-level APIs**, the RL tutorial aims to make the reinforcement learning tutorial simple, transparent and straight-forward with **low-level APIs**, as this would not only benefits new learners of reinforcement learning, but also provide convenience for senior researchers to testify their new ideas quickly.

CHAPTER 26

Contributing

This project is under active development, if you want to join the core team, feel free to contact Zihan Ding at zhding[at]mail.ustc.edu.cn

CHAPTER 27

Citation

- genindex
- modindex
- search

Python Module Index

r

Index

Symbols

A

B

C

D

H

K

L

M

S

T

U

V

W