간단한 예제

[출처] : https://github.com/ray-project/ray/blob/master/rllib/examples/custom_env.py

라이브러리 코드

```
"""
Example of a custom gym environment and model. Run this for a demo.
This example shows:
  - using a custom environment
  - using a custom model
  - using Tune for grid search to try different learning rates
You can visualize experiment results in ~/ray_results using TensorBoard.
Run example with defaults:
$ python custom_env.py
For CLI options:
$ python custom_env.py --help
"""
import argparse
import gym
from gym.spaces import Discrete, Box
import numpy as np
import os
import random

import ray
from ray import tune
from ray.rllib.agents import ppo
from ray.rllib.env.env_context import EnvContext
from ray.rllib.models import ModelCatalog
from ray.rllib.models.tf.tf_modelv2 import TFModelV2
from ray.rllib.models.tf.fcnet import FullyConnectedNetwork
from ray.rllib.models.torch.torch_modelv2 import TorchModelV2
from ray.rllib.models.torch.fcnet import FullyConnectedNetwork as TorchFC
from ray.rllib.utils.framework import try_import_tf, try_import_torch
from ray.rllib.utils.test_utils import check_learning_achieved
from ray.tune.logger import pretty_print
```

기타 설정

```
tf1, tf, tfv = try_import_tf()
torch, nn = try_import_torch()
```

```
parser = argparse.ArgumentParser()
parser.add_argument(
    "--run", type=str, default="PPO", help="The RLlib-registered algorithm to use."
)
parser.add_argument(
    "--framework",
    choices=["tf", "tf2", "tfe", "torch"],
    default="tf",
    help="The DL framework specifier.",
)
parser.add_argument(
    "--as-test",
    action="store_true",
    help="Whether this script should be run as a test: --stop-reward must "
    "be achieved within --stop-timesteps AND --stop-iters.",
)
parser.add_argument(
    "--stop-iters", type=int, default=50, help="Number of iterations to train."
)
parser.add_argument(
    "--stop-timesteps", type=int, default=100000, help="Number of timesteps to train."
)
parser.add_argument(
    "--stop-reward", type=float, default=0.1, help="Reward at which we stop training."
)
parser.add_argument(
    "--no-tune",
    action="store_true",
    help="Run without Tune using a manual train loop instead. In this case,"
    "use PPO without grid search and no TensorBoard.",
)
parser.add_argument(
    "--local-mode",
    action="store_true",
    help="Init Ray in local mode for easier debugging.",
)
```

Environment

```
class SimpleCorridor(gym.Env):
    """Example of a custom env in which you have to walk down a corridor.
    You can configure the length of the corridor via the env config."""

    def __init__(self, config: EnvContext):
        self.end_pos = config["corridor_length"]
        self.cur_pos = 0
        self.action_space = Discrete(2)
        self.observation_space = Box(0.0, self.end_pos, shape=(1,), dtype=np.float32)
        # Set the seed. This is only used for the final (reach goal) reward.
        self.seed(config.worker_index * config.num_workers)

    def reset(self):
        self.cur_pos = 0
        return [self.cur_pos]

    def step(self, action):
        assert action in [0, 1], action
        if action == 0 and self.cur_pos > 0:
            self.cur_pos -= 1
        elif action == 1:
            self.cur_pos += 1
        done = self.cur_pos >= self.end_pos
        # Produce a random reward when we reach the goal.
        return [self.cur_pos], random.random() * 2 if done else -0.1, done, {}

    def seed(self, seed=None):
        random.seed(seed)
```

TF & Torch Model

```
class CustomModel(TFModelV2):
    """Example of a keras custom model that just delegates to an fc-net."""

    def __init__(self, obs_space, action_space, num_outputs, model_config, name):
        super(CustomModel, self).__init__(
            obs_space, action_space, num_outputs, model_config, name
        )
        self.model = FullyConnectedNetwork(
            obs_space, action_space, num_outputs, model_config, name
        )

    def forward(self, input_dict, state, seq_lens):
        return self.model.forward(input_dict, state, seq_lens)

    def value_function(self):
        return self.model.value_function()


class TorchCustomModel(TorchModelV2, nn.Module):
    """Example of a PyTorch custom model that just delegates to a fc-net."""

    def __init__(self, obs_space, action_space, num_outputs, model_config, name):
        TorchModelV2.__init__(
            self, obs_space, action_space, num_outputs, model_config, name
        )
        nn.Module.__init__(self)

        self.torch_sub_model = TorchFC(
            obs_space, action_space, num_outputs, model_config, name
        )
    def forward(self, input_dict, state, seq_lens):
        input_dict["obs"] = input_dict["obs"].float()
        fc_out, _ = self.torch_sub_model(input_dict, state, seq_lens)
        return fc_out, []

    def value_function(self):
        return torch.reshape(self.torch_sub_model.value_function(), [-1])
```

Main Function

```
if __name__ == "__main__":
    args = parser.parse_args()
    print(f"Running with following CLI options: {args}")

    ray.init(local_mode=args.local_mode)

    # Can also register the env creator function explicitly with:
    # register_env("corridor", lambda config: SimpleCorridor(config))
    ModelCatalog.register_custom_model(
        "my_model", TorchCustomModel if args.framework == "torch" else CustomModel
    )

    config = {
        "env": SimpleCorridor,  # or "corridor" if registered above
        "env_config": {
            "corridor_length": 5,
        },
        # Use GPUs iff `RLLIB_NUM_GPUS` env var set to > 0.
        "num_gpus": int(os.environ.get("RLLIB_NUM_GPUS", "0")),
        "model": {
            "custom_model": "my_model",
            "vf_share_layers": True,
        },
        "num_workers": 1,  # parallelism
        "framework": args.framework,
    }

    stop = {
        "training_iteration": args.stop_iters,
        "timesteps_total": args.stop_timesteps,
        "episode_reward_mean": args.stop_reward,
    }

    if args.no_tune:
        # manual training with train loop using PPO and fixed learning rate
        if args.run != "PPO":
            raise ValueError("Only support --run PPO with --no-tune.")
        print("Running manual train loop without Ray Tune.")
        ppo_config = ppo.DEFAULT_CONFIG.copy()
        ppo_config.update(config)
        # use fixed learning rate instead of grid search (needs tune)
        ppo_config["lr"] = 1e-3
        trainer = ppo.PPOTrainer(config=ppo_config, env=SimpleCorridor)
        # run manual training loop and print results after each iteration
        for _ in range(args.stop_iters):
            result = trainer.train()
            print(pretty_print(result))
            # stop training of the target train steps or reward are reached
            if (
                result["timesteps_total"] >= args.stop_timesteps
                or result["episode_reward_mean"] >= args.stop_reward
            ):
                break
    else:
        # automated run with Tune and grid search and TensorBoard
        print("Training automatically with Ray Tune")
        results = tune.run(args.run, config=config, stop=stop)

        if args.as_test:
            print("Checking if learning goals were achieved")
            check_learning_achieved(results, args.stop_reward)

    ray.shutdown()
```

간단한 예제 2

[출처] : https://github.com/DerwenAI/gym_example/blob/main/gym-example/gym_example/envs/example_env.py

1. 코드 나열

코드 - 라이브러리

```
import gym,ray
import numpy as np

from gym.utils import seeding
from ray.rllib.agents import ppo
from ray.tune.registry import register_env
from ray.tune.logger import pretty_print
```

코드 - 환경 구성

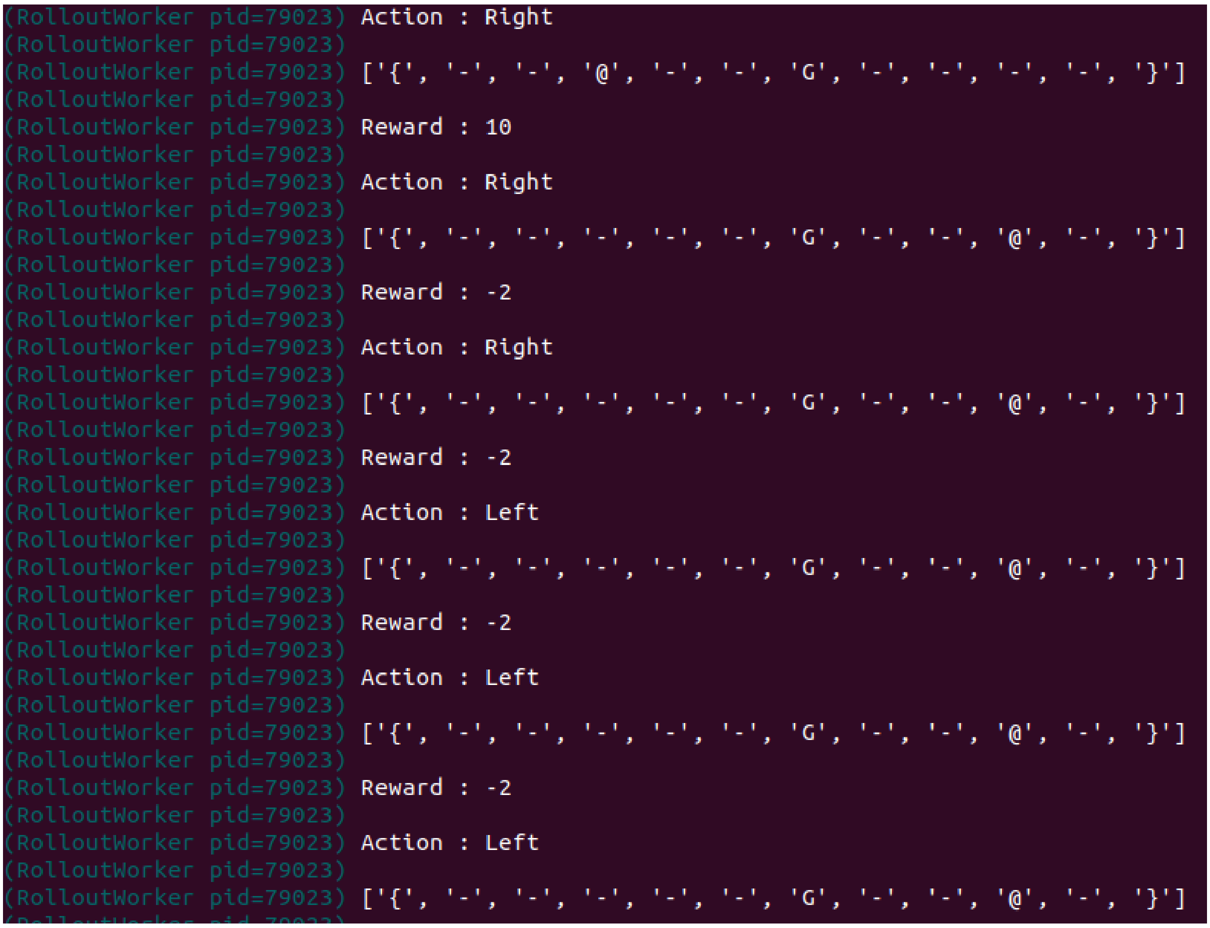
```
class MyEnv(gym.env):
  MOVE_LF = 0
  MOVE_RT = 1

  LF_MIN = 1
  RT_MAX = 10

  MAX_STEPS = 10
  REWARD_AWAY = -2
  REWARD_STEP = -1
  REWARD_GOAL = MAX_STEPS

  metadata = {
      "render.moes" : ["human"]
      }

  def __init__(self,config):

  def reset(self):

  def step(self,action):

  def render(self,action):

  def seed(self,seed=None):

  def close(self):
```

코드 - main function

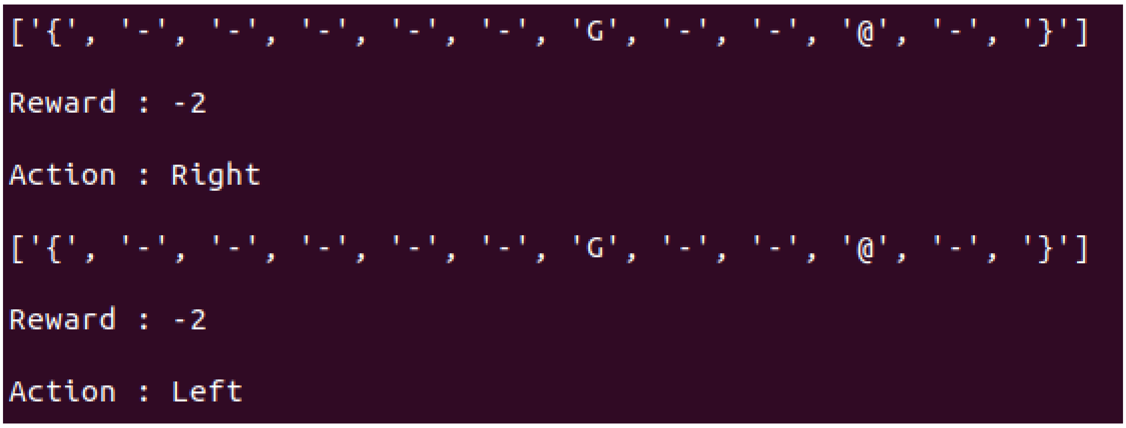
```
def main():
  ray.init()
  register_env("my_env", lambda config : MyEnv(config))
  trainer = ppo.PPOTrainer(env="my_env")

  for i in range(100):
      result = trainer.train()
      if i % 10 == 0 :
          checkpoint = trainer.save()
          print("Checkpoint saved at" , checkpoint)


if __name__=="__main__":
  main()
```

2. 코드 분석

3. 코드 결과값



```
(RolloutWorker pid=79023) Action : Right
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) ['{', '-', '-', '@', '-', '-', 'G', '-', '-', '-', '-', '}']
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) Reward : 10
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) Action : Right
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) ['{', '-', '-', '-', '-', '-', 'G', '-', '-', '@', '-', '}']
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) Reward : -2
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) Action : Right
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) ['{', '-', '-', '-', '-', '-', 'G', '-', '-', '@', '-', '}']
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) Reward : -2
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) Action : Left
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) ['{', '-', '-', '-', '-', '-', 'G', '-', '-', '@', '-', '}']
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) Reward : -2
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) Action : Left
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) ['{', '-', '-', '-', '-', '-', 'G', '-', '-', '@', '-', '}']
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) Reward : -2
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) Action : Left
(RolloutWorker pid=79023)
(RolloutWorker pid=79023) ['{', '-', '-', '-', '-', '-', 'G', '-', '-', '@', '-', '}']
```



```
['{', '-', '-', '-', '-', '-', 'G', '-', '-', '@', '-', '}']

Reward : -2

Action : Right

['{', '-', '-', '-', '-', '-', 'G', '-', '-', '@', '-', '}']

Reward : -2

Action : Left
```

“@” : agent

“G” : Goal Point

Action

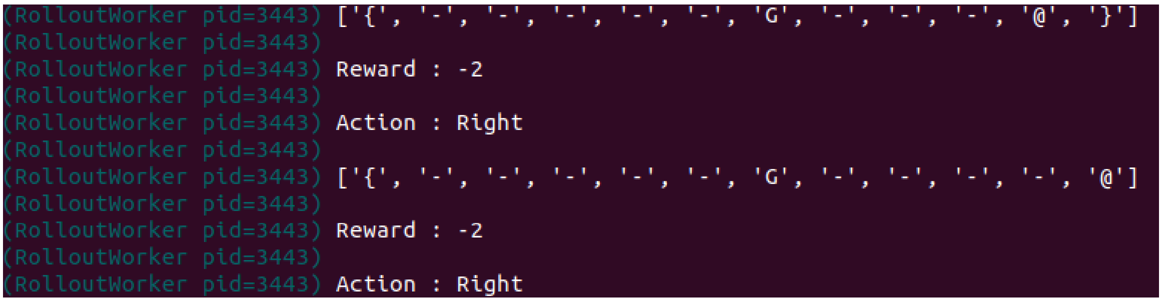
Right : 오른쪽으로 한칸 이동

Left : 왼쪽으로 한칸 이동

>> 문제점 : 캐릭터가 제대로 움직이지 않는다.

>> self.state : 캐릭터의 현재위치가 고정되어져 있다.

>> self.position 을 self.state로 설정



```
(RolloutWorker pid=3443) ['{', '-', '-', '-', '-', '-', 'G', '-', '-', '-', '@', '}']
(RolloutWorker pid=3443)
(RolloutWorker pid=3443) Reward : -2
(RolloutWorker pid=3443)
(RolloutWorker pid=3443) Action : Right
(RolloutWorker pid=3443)
(RolloutWorker pid=3443) ['{', '-', '-', '-', '-', '-', 'G', '-', '-', '-', '-', '@']
(RolloutWorker pid=3443)
(RolloutWorker pid=3443) Reward : -2
(RolloutWorker pid=3443)
(RolloutWorker pid=3443) Action : Right
```