Transfer policy/state learning: metaworld brach, 04/09/2021

1. Learning a good policy in metaworld environment and tasks

(1) We first try the original TD-3 algorithm. This cannot reach more than 60% success rate on push task.

(2) We then try the SAC algorithm. SAC algorithm can reach 100% success rate for “push” and over 80% success rate on “pick-place” in about 1 day training with GPU. Therefore we stick with SAC algorithm

NOTE: SAC algorithm works after we tune the critic network. Specifically, we increase the number of network layers from 2 to 4. 3-layer and 4-layer work fine on these tasks. We might be able to train a policy with TD-3 if we also increase the critic layers, while I didn’t experiment on this.

First, source “env.sh” for correct environment.

To train a metaworld task “push” with SAC:

```
source env.sh
cd cross_physics/base_train_test/td3_solver/torchrl
```

*****Double check env/vecenv.py, make sure the environment is push in the reset() function***** Currently I set it to push, so there shouldn’t be extra work. If you would like to change environment, we need to change this as well.

```
python examples/twin_sac_q_continuous_vec.py --config config/push.json --seed 0 --id push --overwrite
```

To render a SAC policy on a metaworld task:

```
source env.sh
cd cross_physics/base_train_test/td3_solver/torchrl
```

*****Double check env/vecenv.py, make sure the environment is the environment you want in the reset() function***** Currently it is “push”.

*****Change collector/base.py, uncomment the lines 279-283 and 302-310 in the Vecenv::eval_one_epoch() function***** This part is responsible for rendering images. During training, this needs to be commented to increase training speed.

```
python examples/image_render_test.py --config config/push.json --seed 0 --id push_observ --overwrite
```

2. Learning to transfer a policy (fix the states)

(0) We should have the policy ready before transferring them. The policy we trained with SAC include:

- push
- door open
- pick and place
- coffee push
- push back

(1) Transfer a policy is similar to cross-physics. We fix the state distribution first, and we observe the action distribution - they should be different.

(2) We tried transferring from door open to push; from push to door open; from coffee push to push.

To collect data for a metaworld task:

```
source env.sh
cd cross_physics/cycle_transfer
python collect_data_push_dooropen.py --data_type 'push' --data_id 2004 --env push
```

To change the distribution of a metaworld task:

Navigate to your meta world python build (on my end, it is “/mnt/brain7/scratch/wuqiuche/anaconda3/lib/python3.8/site-packages/metaworld”)

```
cd envs/assets_v1
```

To transfer the policy from 2 tasks:

```
source env.sh
cd cross_physics/cycle_transfer
python forwardexp.py --data_type1 'push' --data_id1 2003 --data_type2 ‘coffee_push' --data_id2 2003
```

3. Learning to transfer a state (fix the policy)

(0) Similarly, the policy should be ready.

(1) Transfer a state is similar to cross-modality. However, we change the code because we are not changing from rendered image to gym state. We fix the action distribution, and we observe the state distribution - they should be different.

(2) We tried transferring from push to coffee push; from push to push back.

To start transferring:

```
source env.sh
cd cross_physics/cycle_transfer
python cycleexp_ours.py --data_type1 'push' --data_id1 2003 --data_type2 'push_back' --data_id2 2003
```