

✓ DQN for continuous action spaces: Normalized Advantage Function (NAF)

```
!apt-get update && apt-get install -y xvfb
```

```

Unpacking libxkbfile1:amd64 (1:1.1.0-1build3) ...
Selecting previously unselected package x11-xkb-utils.
Preparing to unpack .../3-x11-xkb-utils_7.7+5build4_amd64.deb ...
Unpacking x11-xkb-utils (7.7+5build4) ...
Selecting previously unselected package xfonts-encodings.
Preparing to unpack .../4-xfonts-encodings_1%3a1.0.5-0ubuntu2_all.deb ...
Unpacking xfonts-encodings (1:1.0.5-0ubuntu2) ...
Selecting previously unselected package xfonts-utils.
Preparing to unpack .../5-xfonts-utils_1%3a7.7+6build2_amd64.deb ...
Unpacking xfonts-utils (1:7.7+6build2) ...
Selecting previously unselected package xfonts-base.
Preparing to unpack .../6-xfonts-base_1%3a1.0.5_all.deb ...
Unpacking xfonts-base (1:1.0.5) ...
Selecting previously unselected package xserver-common.
Preparing to unpack .../7-xserver-common_2%3a21.1.4-2ubuntu1.7~22.04.12_all.deb ...
Unpacking xserver-common (2:21.1.4-2ubuntu1.7~22.04.12) ...
Selecting previously unselected package xvfb.
Preparing to unpack .../8-xvfb_2%3a21.1.4-2ubuntu1.7~22.04.12_amd64.deb ...
Unpacking xvfb (2:21.1.4-2ubuntu1.7~22.04.12) ...
Setting up libfontenc1:amd64 (1:1.1.4-1build3) ...
Setting up xfonts-encodings (1:1.0.5-0ubuntu2) ...
Setting up libxkbfile1:amd64 (1:1.1.0-1build3) ...
Setting up libxfont2:amd64 (1:2.0.5-1build1) ...
Setting up x11-xkb-utils (7.7+5build4) ...
Setting up xfonts-utils (1:7.7+6build2) ...
Setting up xfonts-base (1:1.0.5) ...
Setting up xserver-common (2:21.1.4-2ubuntu1.7~22.04.12) ...
Setting up xvfb (2:21.1.4-2ubuntu1.7~22.04.12) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for fontconfig (2.13.1-4.2ubuntu5) ...
Processing triggers for libc-bin (2.35-0ubuntu3.4) ...
/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_0.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libur_adapter_level_zero.so.0 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libur_loader.so.0 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind_2_5.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libur_adapter_llvm.so.0 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbb.so.12 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libhwloc.so.15 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtcm.so.1 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc_proxy.so.2 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbbind.so.3 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libumf.so.0 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtbbmalloc.so.2 is not a symbolic link

/sbin/ldconfig.real: /usr/local/lib/libtcm_debug.so.1 is not a symbolic link

```

```
!pip install swig
```

```

Collecting swig
  Downloading swig-4.2.1.post0-py2.py3-none-manylinux_2_5_x86_64.manylinux1_x86_64.whl.metadata (3.5 kB)
  Downloading swig-4.2.1.post0-py2.py3-none-manylinux_2_5_x86_64.manylinux1_x86_64.whl (1.8 MB)
    ----- 1.8/1.8 MB 24.7 MB/s eta 0:00:00
Installing collected packages: swig
Successfully installed swig-4.2.1.post0

```

```
!pip install gym[box2d]==0.23.1
```

```

Collecting gym==0.23.1 (from gym[box2d]==0.23.1)
  Downloading gym-0.23.1.tar.gz (626 kB)
    ----- 626.2/626.2 kB 10.7 MB/s eta 0:00:00
Installing build dependencies ... done
Getting requirements to build wheel ... done
Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: numpy>=1.18.0 in /usr/local/lib/python3.10/dist-packages (from gym==0.23.1->gym[box2d]==0.23.1) (1.26.4)
Requirement already satisfied: cloudpickle>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from gym==0.23.1->gym[box2d]==0.23.1) (2.2.1)
Requirement already satisfied: gym-notices>=0.0.4 in /usr/local/lib/python3.10/dist-packages (from gym==0.23.1->gym[box2d]==0.23.1) (0.0.8)
Collecting box2d-py==2.3.5 (from gym[box2d]==0.23.1)

```

```

Downloading box2d-py-2.3.5.tar.gz (374 kB)
374.4/374.4 kB 22.6 MB/s eta 0:00:00
Preparing metadata (setup.py) ... done
Collecting pygame==2.1.0 (from gym[box2d]==0.23.1)
  Downloading pygame-2.1.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (9.5 kB)
Downloading pygame-2.1.0-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (18.3 MB)
18.3/18.3 MB 78.4 MB/s eta 0:00:00
Building wheels for collected packages: gym, box2d-py
  Building wheel for gym (pyproject.toml) ... done
  Created wheel for gym: filename=gym-0.23.1-py3-none-any.whl size=701345 sha256=7b875f6ae3ab119a753f84b75fdb6d4918e83d5f2b69cd63f94
  Stored in directory: /root/.cache/pip/wheels/1a/00/fb/fe5cf2860fb9b7bc860e28f00095a1f42c7b726dd6f42d1acc
  Building wheel for box2d-py (setup.py) ... done
  Created wheel for box2d-py: filename=box2d_py-2.3.5-cp310-cp310-linux_x86_64.whl size=2376102 sha256=7e76b5f257c38e92b99f96a5b7d44
  Stored in directory: /root/.cache/pip/wheels/db/8f/6a/eaadff056fba10a98d986f6dce954e6201ba3126926fc5ad9e
Successfully built gym box2d-py
Installing collected packages: box2d-py, pygame, gym
  Attempting uninstall: pygame
    Found existing installation: pygame 2.6.1
    Uninstalling pygame-2.6.1:
      Successfully uninstalled pygame-2.6.1
  Attempting uninstall: gym
    Found existing installation: gym 0.25.2
    Uninstalling gym-0.25.2:
      Successfully uninstalled gym-0.25.2
Successfully installed box2d-py-2.3.5 gym-0.23.1 pygame-2.1.0

```

```
!pip install pytorch_lightning
```

```

Collecting pytorch_lightning
  Downloading pytorch_lightning-2.4.0-py3-none-any.whl.metadata (21 kB)
  Requirement already satisfied: torch>=2.1.0 in /usr/local/lib/python3.10/dist-packages (from pytorch_lightning) (2.5.0+cu121)
  Requirement already satisfied: tqdm>=4.57.0 in /usr/local/lib/python3.10/dist-packages (from pytorch_lightning) (4.66.6)
  Requirement already satisfied: PyYAML>=5.4 in /usr/local/lib/python3.10/dist-packages (from pytorch_lightning) (6.0.2)
  Requirement already satisfied: fsspec>=2022.5.0 in /usr/local/lib/python3.10/dist-packages (from fsspec[http]>=2022.5.0->pytorch_li
  Collecting torchmetrics>=0.7.0 (from pytorch_lightning)
    Downloading torchmetrics-1.5.1-py3-none-any.whl.metadata (20 kB)
    Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from pytorch_lightning) (24.1)
    Requirement already satisfied: typing-extensions>=4.4.0 in /usr/local/lib/python3.10/dist-packages (from pytorch_lightning) (4.12.2)
    Collecting lightning-utilities>=0.10.0 (from pytorch_lightning)
      Downloading lightning_utilities-0.11.8-py3-none-any.whl.metadata (5.2 kB)
      Requirement already satisfied: aiohttp!=4.0.0a0,!4.0.0a1 in /usr/local/lib/python3.10/dist-packages (from fsspec[http]>=2022.5.0->
      Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from lightning-utilities>=0.10.0->pytorch_lig
      Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from torch>=2.1.0->pytorch_lightning) (3.16.1)
      Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch>=2.1.0->pytorch_lightning) (3.4.2)
      Requirement already satisfied: Jinja2 in /usr/local/lib/python3.10/dist-packages (from torch>=2.1.0->pytorch_lightning) (3.1.4)
      Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.10/dist-packages (from torch>=2.1.0->pytorch_lightning) (1.13
      Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy==1.13.1->torch>=2.1.0->pytc
      Requirement already satisfied: numpy<2.0,>=1.20.0 in /usr/local/lib/python3.10/dist-packages (from torchmetrics>=0.7.0->pytorch_light
      Requirement already satisfied: aiohappyeyeballs>=2.3.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp!=4.0.0a0,!4.0.0a1->
      Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.10/dist-packages (from aiohttp!=4.0.0a0,!4.0.0a1->fsspec
      Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp!=4.0.0a0,!4.0.0a1->fsspec[htt
      Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from aiohttp!=4.0.0a0,!4.0.0a1->fsspec
      Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.10/dist-packages (from aiohttp!=4.0.0a0,!4.0.0a1->fsspe
      Requirement already satisfied: yarl<2.0,>=1.12.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp!=4.0.0a0,!4.0.0a1->fsspec
      Requirement already satisfied: async-timeout<5.0,>=4.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp!=4.0.0a0,!4.0.0a1->
      Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from Jinja2->torch>=2.1.0->pytorch_lightn
      Requirement already satisfied: idna>=2.0 in /usr/local/lib/python3.10/dist-packages (from yarl<2.0,>=1.12.0->aiohttp!=4.0.0a0,!4.0.0
      Requirement already satisfied: propcache>=0.2.0 in /usr/local/lib/python3.10/dist-packages (from yarl<2.0,>=1.12.0->aiohttp!=4.0.0a0
  Downloading pytorch_lightning-2.4.0-py3-none-any.whl (815 kB)
815.2/815.2 kB 17.2 MB/s eta 0:00:00
  Downloading lightning_utilities-0.11.8-py3-none-any.whl (26 kB)
  Downloading torchmetrics-1.5.1-py3-none-any.whl (890 kB)
890.6/890.6 kB 27.0 MB/s eta 0:00:00
Installing collected packages: lightning-utilities, torchmetrics, pytorch_lightning
Successfully installed lightning-utilities-0.11.8 pytorch_lightning-2.4.0 torchmetrics-1.5.1

```

```
!pip install pyvirtualdisplay
```

```

Collecting pyvirtualdisplay
  Downloading PyVirtualDisplay-3.0-py3-none-any.whl.metadata (943 bytes)
  Downloading PyVirtualDisplay-3.0-py3-none-any.whl (15 kB)
Installing collected packages: pyvirtualdisplay
Successfully installed pyvirtualdisplay-3.0

```

✓ Setup virtual display

```

from pyvirtualdisplay import Display
Display(visible=False, size=(1400, 900)).start()

```

```
<pyvirtualdisplay.display.Display at 0x7d4cd0899cf0>
```

▼ Import the necessary code libraries

```
import copy
import gym
import random
import torch

import numpy as np
import torch.nn.functional as F

from collections import deque, namedtuple
from IPython.display import HTML
from base64 import b64encode


from torch import nn
from torch.utils.data import DataLoader
from torch.utils.data.dataset import IterableDataset
from torch.optim import AdamW

from pytorch_lightning import LightningModule, Trainer

from gym.wrappers import RecordVideo, RecordEpisodeStatistics

device = 'cuda' if torch.cuda.is_available() else 'cpu'
num_gpus = torch.cuda.device_count()
```

```
def display_video(episode=0):
    video_file = open(f'/content/videos/rl-video-episode-{episode}.mp4', "r+b").read()
    video_url = f"data:video/mp4;base64,{b64encode(video_file).decode()}"
    return HTML(f"<video width=600 controls><source src='{video_url}'></video>")
```

 /usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transform` and `should_run_async` (code)

▼ Create the Deep Q-Network

```
#the nafdqN class for deep q network
#mu method for compute action highest Q-Value, which action will compute best return
#value-: compute value state, that the expected max return we can expect starting from that state
#forward method-: we will compute q-value for a state and action use value of the state and the mu value for the action
#max_action: continue action will have a min and max value, allow us to scale continous actions
#this neural network will produce three different values, common layers will be made first
#creating a layer called linear mu that will use to compute the mu value.
#creating a layer called linear value that will use to compute the value of a state.
#creating a layer for linear_matrix that will compute linear matrix for bilinear form, will compute right values for the advantage
class NafDQN(nn.Module):
    def __init__(self, hidden_size, obs_size, action_dims, max_action):
        super().__init__()
        self.action_dims = action_dims
        self.max_action = torch.from_numpy(max_action).to(device)
        self.net = nn.Sequential(
            nn.Linear(obs_size, hidden_size),
            nn.ReLU(),
            nn.Linear(hidden_size, hidden_size),
            nn.ReLU(),
        )
        self.linear_mu = nn.Linear(hidden_size, action_dims)
        self.linear_value = nn.Linear(hidden_size, 1)
        self.linear_matrix = nn.Linear(hidden_size, int(action_dims * (action_dims + 1) / 2))

    #we are taking a vector of values that resulted from this operation, those values are in range
    #[-inf,inf], to scale the values from tanh to the correct range we need to multiply by max_action
    @torch.no_grad()
    def mu(self, x):
        x = self.net(x)
        x = self.linear_mu(x)
        x = torch.tanh(x) * self.max_action
        return x

    @torch.no_grad()
    def value(self, x):
        x = self.net(x)
        x = self.linear_value(x)
        return x

    # state and action, compute mu value first
```

```
#compute value and matrix with lower range values
# L -: is the lower matrix that we will form from the vector
#we will make the index for the lower matrix
def forward(self, x, a):
    x = self.net(x)
    mu = torch.tanh(self.linear_mu(x)) * self.max_action
    value = self.linear_value(x)
    matrix = torch.tanh(self.linear_matrix(x))

    L = torch.zeros((x.shape[0], self.action_dims, self.action_dims)).to(device) #batch_size, action_dims, action_dims

    tril_indices = torch.tril_indices(row=self.action_dims, col=self.action_dims, offset=0).to(device) #an array of indices as output

    L[:, tril_indices[0], tril_indices[1]] = matrix
    L.diagonal(dim1=1,dim2=2).exp_() #we need to make all the diagonals positive, for getting a positive B matrix
    P = L * L.transpose(2, 1)

    #we will compute the advantage function

    u_mu = (a-mu).unsqueeze(dim=1)
    u_mu_t = u_mu.transpose(1, 2)

    adv = - 1/2 * u_mu @ P @ u_mu_t #matrix multiplication
    adv = adv.squeeze(dim=-1) #will remove one dimension

    return value + adv
```

▼ Create the policy

```
#noisy policy(state, environment,neural_network,epsilon)
#state tensor
#amin will get the min value for each action it can take
#amax for getting highest value of the action
#will compute mu value of the state from the net,give actions that will estimate has the highest q value
#will add some noise, will shift the value with normal values, to explore the values, epsilon is the standard deviation
#action will be clamped between max and min value
#return the action
def noisy_policy(state, env, net, epsilon=0.0):
    state = torch.tensor([state]).to(device)
    amin = torch.from_numpy(env.action_space.low).to(device)
    amax = torch.from_numpy(env.action_space.high).to(device)
    mu = net.mu(state)
    mu = mu + torch.normal(0, epsilon, mu.size(), device=device)
    action = mu.clamp(amin, amax)
    action = action.squeeze().cpu().numpy()
    return action
```

▼ Create the replay buffer

```
class ReplayBuffer:

    def __init__(self, capacity):
        self.buffer = deque(maxlen=capacity)

    def __len__(self):
        return len(self.buffer)

    def append(self, experience):
        self.buffer.append(experience)

    def sample(self, batch_size):
        return random.sample(self.buffer, batch_size)
```

```
class RLDataset(IterableDataset):

    def __init__(self, buffer, sample_size=400):
        self.buffer = buffer
        self.sample_size = sample_size

    def __iter__(self):
        for experience in self.buffer.sample(self.sample_size):
            yield experience
```

▼ Create the environment

```
#this class will do is that when we select an action that we applied in the environment
#it will keep that action and apply in the following states.
#override the step method, for n times we will apply that action in the env
#return the next_state,total_reward and done
```

```
class RepeatActionWrapper(gym.Wrapper):
    def __init__(self, env, n):
        super().__init__(env)
        self.env = env
        self.n = n

    def step(self, action):
        done = False
        total_reward = 0.0
        for _ in range(self.n):
            next_state, reward, done, info = self.env.step(action)
            total_reward += reward
            if done:
                break
        return next_state, total_reward, done, info
```

```
def create_environment(name):
    env = gym.make(name)
    env = RecordVideo(env, video_folder='./videos', episode_trigger=lambda x: x % 50 == 0)
    env = RepeatActionWrapper(env, n=8)
    env = RecordEpisodeStatistics(env)
    return env
```

▼ Update the target network

```
#the polyak_average technique, will make a soft update for updating the target network

def polyak_average(net, target_net, tau=0.01):
    for qp, tp in zip(net.parameters(), target_net.parameters()):
        tp.data.copy_(tau * qp.data + (1 - tau) * tp.data)
```

▼ Create the Deep Q-Learning algorithm

```
#env,policy,capacity of replay buffer,batch size,learning rate,hidden size
#discount factor(gamma), loss function, adam optimizer, tau will update target network
#will create environment
#max_value of the action, upper barrier
#will create q-network and target_q_network
#create buffer and policy
#save hyperparameters

class NAFDeepQLearning(LightningModule):

    def __init__(self, env_name, policy=noisy_policy, capacity=100_000,
                 batch_size=256, lr=1e-4, hidden_size=512, gamma=0.99,
                 loss_fn=F.smooth_l1_loss, optim=AdamW, eps_start=2.0, eps_end=0.2,
                 eps_last_episode=1_000, samples_per_epoch=1_000, tau=0.01):

        super().__init__()
        self.env = create_environment(env_name)

        obs_size = self.env.observation_space.shape[0]
        action_dims = self.env.action_space.shape[0]
        max_action = self.env.action_space.high

        self.q_net = NafDQN(hidden_size, obs_size, action_dims, max_action).to(device)
        self.target_q_net = copy.deepcopy(self.q_net)
        self.policy = policy

        self.buffer = ReplayBuffer(capacity=capacity)

        self.save_hyperparameters()

        while len(self.buffer) < self.hparams.samples_per_epoch:

            print(f"{len(self.buffer)} samples in experience buffer. Filling...")
            self.play_episode(epsilon=self.hparams.eps_start)

    @torch.no_grad()
    def play_episode(self, policy=None, epsilon=0.):
        obs = self.env.reset()
        done = False
```

```

while not done:
    if policy:
        action = policy(obs, self.env, self.q_net, epsilon=epsilon)
    else:
        action = self.env.action_space.sample()

    next_obs, reward, done, info = self.env.step(action)
    exp = (obs, action, reward, done, next_obs)
    self.buffer.append(exp)
    obs = next_obs

def forward(self, x):
    output = self.q_net.mu(x)
    return output

def configure_optimizers(self):
    q_net_optimizer = self.hparams.optim(self.q_net.parameters(), lr=self.hparams.lr)
    return [q_net_optimizer]

def train_dataloader(self):
    dataset = RLDataset(self.buffer, self.hparams.samples_per_epoch)
    dataloader = DataLoader(
        dataset=dataset,
        batch_size=self.hparams.batch_size,
    )
    return dataloader

def training_step(self, batch, batch_idx):
    states, actions, rewards, dones, next_states = batch
    rewards = rewards.unsqueeze(1)
    dones = dones.unsqueeze(1)

    action_values = self.q_net(states, actions)

    next_state_values = self.target_q_net.value(next_states)
    next_state_values[dones] = 0.0

    target = rewards + self.hparams.gamma * next_state_values

    loss = self.hparams.loss_fn(action_values, target)
    self.log('episode/MSE Loss', loss)
    return loss

def on_train_epoch_end(self):
    """Called at the end of the training epoch with the outputs of all training steps."""
    epsilon = max(
        self.hparams.eps_end,
        self.hparams.eps_start - self.current_epoch / self.hparams.eps_last_episode
    )

    self.play_episode(policy=self.policy, epsilon=epsilon)

    polyak_average(self.q_net, self.target_q_net, tau=self.hparams.tau)

    self.log("episode/Return", self.env.return_queue[-1])

```

✓ Purge logs and run the visualization tool (Tensorboard)

```

# Start tensorboard.
!rm -r /content/lightning_logs/
!rm -r /content/videos/
%load_ext tensorboard
%tensorboard --logdir /content/lightning_logs/

```

```
rm: cannot remove '/content/lightning_logs/': No such file or directory
rm: cannot remove '/content/videos/': No such file or directory
```

TensorBoard

TIME SERIES

SCALARS

HPARAMS

INACTIVE

 Filter tags (regex)

All

Scalars

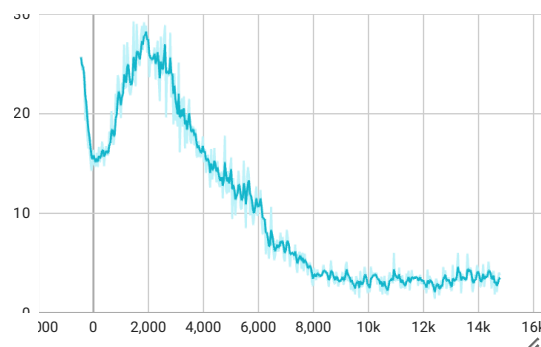
Image

Histogram

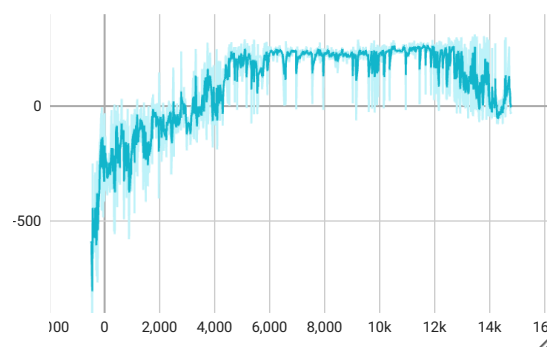
 Settings

episode 2 cards

episode/MSE Loss



episode/Return



epoch

epoch

4000

Train the policy

```
import pytorch_lightning as pl
import warnings
warnings.filterwarnings('ignore')

algo = NAFDeepQLearning('LunarLanderContinuous-v2')

#run for 100000 max_epochs for better results

trainer = pl.Trainer(
    accelerator="gpu" if num_gpus else "cpu", # Use 'gpu' if num_gpus is greater than 0, otherwise use 'cpu'
    devices=1, # Specify the number of GPUs or 'auto' for automatic detection
    max_epochs=5000
)

trainer.fit(algo)
```



```

10 samples in experience buffer. Filling...
21 samples in experience buffer. Filling...
35 samples in experience buffer. Filling...
45 samples in experience buffer. Filling...
55 samples in experience buffer. Filling...
76 samples in experience buffer. Filling...
89 samples in experience buffer. Filling...
104 samples in experience buffer. Filling...
120 samples in experience buffer. Filling...
133 samples in experience buffer. Filling...
144 samples in experience buffer. Filling...
153 samples in experience buffer. Filling...
168 samples in experience buffer. Filling...
178 samples in experience buffer. Filling...
193 samples in experience buffer. Filling...
206 samples in experience buffer. Filling...
225 samples in experience buffer. Filling...
234 samples in experience buffer. Filling...
249 samples in experience buffer. Filling...
275 samples in experience buffer. Filling...
285 samples in experience buffer. Filling...
295 samples in experience buffer. Filling...
306 samples in experience buffer. Filling...
319 samples in experience buffer. Filling...
331 samples in experience buffer. Filling...
349 samples in experience buffer. Filling...
365 samples in experience buffer. Filling...
379 samples in experience buffer. Filling...
392 samples in experience buffer. Filling...
404 samples in experience buffer. Filling...
419 samples in experience buffer. Filling...
436 samples in experience buffer. Filling...
457 samples in experience buffer. Filling...
469 samples in experience buffer. Filling...
490 samples in experience buffer. Filling...
501 samples in experience buffer. Filling...
510 samples in experience buffer. Filling...
517 samples in experience buffer. Filling...
528 samples in experience buffer. Filling...
565 samples in experience buffer. Filling...
574 samples in experience buffer. Filling...
585 samples in experience buffer. Filling...
594 samples in experience buffer. Filling...
605 samples in experience buffer. Filling...
617 samples in experience buffer. Filling...
633 samples in experience buffer. Filling...
649 samples in experience buffer. Filling...
660 samples in experience buffer. Filling...
671 samples in experience buffer. Filling...
682 samples in experience buffer. Filling...
691 samples in experience buffer. Filling...
704 samples in experience buffer. Filling...
717 samples in experience buffer. Filling...
727 samples in experience buffer. Filling...
749 samples in experience buffer. Filling...
763 samples in experience buffer. Filling...
778 samples in experience buffer. Filling...
789 samples in experience buffer. Filling...
809 samples in experience buffer. Filling...
820 samples in experience buffer. Filling...
833 samples in experience buffer. Filling...
845 samples in experience buffer. Filling...
877 samples in experience buffer. Filling...
889 samples in experience buffer. Filling...
901 samples in experience buffer. Filling...
912 samples in experience buffer. Filling...
925 samples in experience buffer. Filling...
938 samples in experience buffer. Filling...
949 samples in experience buffer. Filling...
965 samples in experience buffer. Filling...
974 samples in experience buffer. Filling...
INFO:pytorch_lightning.utilities.rank_zero:GPU available: False, used: False
INFO:pytorch_lightning.utilities.rank_zero:TPU available: False, using: 0 TPU cores
INFO:pytorch_lightning.utilities.rank_zero:HPU available: False, using: 0 HPUs
INFO:pytorch_lightning.callbacks.model_summary:
  | Name          | Type          | Params | Mode
-----|-----|-----|-----
0 | q_net          | NafDQN       | 270 K  | train
1 | target_q_net  | NafDQN       | 270 K  | train
-----|-----|-----|-----
540 K    Trainable params
0        Non-trainable params
540 K    Total params
2.163    Total estimated model params size (MB)
18       Modules in train mode
0        Modules in eval mode
987 samples in experience buffer. Filling...

Epoch 4999:

```

4/? [00:00<00:00, 19.78it/s, v_num=1]

INFO:pytorch_lightning.utilities.rank_zero: `Trainer.fit` stopped: `max_epochs=5000` reached.

✓ Check the resulting policy

```
display_video(episode=3000)
```



0:16 / 0:16

```
!zip -r /content/lightning_logs.zip /content/lightning_logs
```



```
adding: content/lightning_logs/ (stored 0%)
adding: content/lightning_logs/version_0/ (stored 0%)
adding: content/lightning_logs/version_0/events.out.tfevents.1730526399.e6202b7248cb.371.0 (deflated 68%)
adding: content/lightning_logs/version_0/checkpoints/ (stored 0%)
adding: content/lightning_logs/version_0/checkpoints/epoch=999-step=4000.ckpt (deflated 11%)
adding: content/lightning_logs/version_0/hparams.yaml (deflated 36%)
adding: content/lightning_logs/version_1/ (stored 0%)
adding: content/lightning_logs/version_1/events.out.tfevents.1730527103.e6202b7248cb.371.1 (deflated 68%)
adding: content/lightning_logs/version_1/checkpoints/ (stored 0%)
adding: content/lightning_logs/version_1/checkpoints/epoch=4999-step=20000.ckpt (deflated 9%)
adding: content/lightning_logs/version_1/hparams.yaml (deflated 36%)
```

```
!zip -r /content/videos.zip /content/videos
```

