Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»



**Лабораторная работа №4**

**По курсу «Методы машинного обучения»**

**ИСПОЛНИТЕЛЬ:**

Лю Цзыцзянь

Группа ИУ5И-21М

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ПРОВЕРИЛ:**

Гапанюк Ю.Е.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оглавление

[Лабораторная работа №4 3](#_Toc200474533)

[Импортируйте необходимые библиотеки 4](#_Toc200474534)

[Описание реализованного алгоритма 9](#_Toc200474535)

[График средней награды 13](#_Toc200474536)

[Вывод 16](#_Toc200474537)

# Лабораторная работа №4

Алгоритмы Actor-Critic

Цель работы:

Ознакомление с базовыми методами обучения с подкреплением на основе алгоритмов Actor-Critic.

Задание:

Реализовать любой алгоритм семейства Actor-Critic для произвольной среды.

# Импортируйте необходимые библиотеки

In [ ]:

**import** gym

**import** numpy **as** np

**import** tensorflow **as** tf

**from** tensorflow.keras **import** layers

**import** matplotlib.pyplot **as** plt

In [ ]:

**!**pip install gymnasium**[**box2d**]** torch matplotlib

Requirement already satisfied: torch in /usr/local/lib/python3.11/dist-packages (2.6.0+cu124)

Requirement already satisfied: matplotlib in /usr/local/lib/python3.11/dist-packages (3.10.0)

Requirement already satisfied: gymnasium[box2d] in /usr/local/lib/python3.11/dist-packages (1.1.1)

Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.11/dist-packages (from gymnasium[box2d]) (2.0.2)

Requirement already satisfied: cloudpickle>=1.2.0 in /usr/local/lib/python3.11/dist-packages (from gymnasium[box2d]) (3.1.1)

Requirement already satisfied: typing-extensions>=4.3.0 in /usr/local/lib/python3.11/dist-packages (from gymnasium[box2d]) (4.13.2)

Requirement already satisfied: farama-notifications>=0.0.1 in /usr/local/lib/python3.11/dist-packages (from gymnasium[box2d]) (0.0.4)

Collecting box2d-py==2.3.5 (from gymnasium[box2d])

Downloading box2d-py-2.3.5.tar.gz (374 kB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 374.4/374.4 kB 4.5 MB/s eta 0:00:00

Preparing metadata (setup.py) ... done

Requirement already satisfied: pygame>=2.1.3 in /usr/local/lib/python3.11/dist-packages (from gymnasium[box2d]) (2.6.1)

Collecting swig==4.\* (from gymnasium[box2d])

Downloading swig-4.3.1-py3-none-manylinux\_2\_12\_x86\_64.manylinux2010\_x86\_64.whl.metadata (3.5 kB)

Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from torch) (3.18.0)

Requirement already satisfied: networkx in /usr/local/lib/python3.11/dist-packages (from torch) (3.4.2)

Requirement already satisfied: jinja2 in /usr/local/lib/python3.11/dist-packages (from torch) (3.1.6)

Requirement already satisfied: fsspec in /usr/local/lib/python3.11/dist-packages (from torch) (2025.3.2)

Collecting nvidia-cuda-nvrtc-cu12==12.4.127 (from torch)

Downloading nvidia\_cuda\_nvrtc\_cu12-12.4.127-py3-none-manylinux2014\_x86\_64.whl.metadata (1.5 kB)

Collecting nvidia-cuda-runtime-cu12==12.4.127 (from torch)

Downloading nvidia\_cuda\_runtime\_cu12-12.4.127-py3-none-manylinux2014\_x86\_64.whl.metadata (1.5 kB)

Collecting nvidia-cuda-cupti-cu12==12.4.127 (from torch)

Downloading nvidia\_cuda\_cupti\_cu12-12.4.127-py3-none-manylinux2014\_x86\_64.whl.metadata (1.6 kB)

Collecting nvidia-cudnn-cu12==9.1.0.70 (from torch)

Downloading nvidia\_cudnn\_cu12-9.1.0.70-py3-none-manylinux2014\_x86\_64.whl.metadata (1.6 kB)

Collecting nvidia-cublas-cu12==12.4.5.8 (from torch)

Downloading nvidia\_cublas\_cu12-12.4.5.8-py3-none-manylinux2014\_x86\_64.whl.metadata (1.5 kB)

Collecting nvidia-cufft-cu12==11.2.1.3 (from torch)

Downloading nvidia\_cufft\_cu12-11.2.1.3-py3-none-manylinux2014\_x86\_64.whl.metadata (1.5 kB)

Collecting nvidia-curand-cu12==10.3.5.147 (from torch)

Downloading nvidia\_curand\_cu12-10.3.5.147-py3-none-manylinux2014\_x86\_64.whl.metadata (1.5 kB)

Collecting nvidia-cusolver-cu12==11.6.1.9 (from torch)

Downloading nvidia\_cusolver\_cu12-11.6.1.9-py3-none-manylinux2014\_x86\_64.whl.metadata (1.6 kB)

Collecting nvidia-cusparse-cu12==12.3.1.170 (from torch)

Downloading nvidia\_cusparse\_cu12-12.3.1.170-py3-none-manylinux2014\_x86\_64.whl.metadata (1.6 kB)

Requirement already satisfied: nvidia-cusparselt-cu12==0.6.2 in /usr/local/lib/python3.11/dist-packages (from torch) (0.6.2)

Requirement already satisfied: nvidia-nccl-cu12==2.21.5 in /usr/local/lib/python3.11/dist-packages (from torch) (2.21.5)

Requirement already satisfied: nvidia-nvtx-cu12==12.4.127 in /usr/local/lib/python3.11/dist-packages (from torch) (12.4.127)

Collecting nvidia-nvjitlink-cu12==12.4.127 (from torch)

Downloading nvidia\_nvjitlink\_cu12-12.4.127-py3-none-manylinux2014\_x86\_64.whl.metadata (1.5 kB)

Requirement already satisfied: triton==3.2.0 in /usr/local/lib/python3.11/dist-packages (from torch) (3.2.0)

Requirement already satisfied: sympy==1.13.1 in /usr/local/lib/python3.11/dist-packages (from torch) (1.13.1)

Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.11/dist-packages (from sympy==1.13.1->torch) (1.3.0)

Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.3.2)

Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (0.12.1)

Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (4.58.0)

Requirement already satisfied: kiwisolver>=1.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (1.4.8)

Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (24.2)

Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (11.2.1)

Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (3.2.3)

Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.11/dist-packages (from matplotlib) (2.9.0.post0)

Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.7->matplotlib) (1.17.0)

Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.11/dist-packages (from jinja2->torch) (3.0.2)

Downloading swig-4.3.1-py3-none-manylinux\_2\_12\_x86\_64.manylinux2010\_x86\_64.whl (1.9 MB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.9/1.9 MB 46.9 MB/s eta 0:00:00

Downloading nvidia\_cublas\_cu12-12.4.5.8-py3-none-manylinux2014\_x86\_64.whl (363.4 MB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 363.4/363.4 MB 4.3 MB/s eta 0:00:00

Downloading nvidia\_cuda\_cupti\_cu12-12.4.127-py3-none-manylinux2014\_x86\_64.whl (13.8 MB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.8/13.8 MB 63.4 MB/s eta 0:00:00

Downloading nvidia\_cuda\_nvrtc\_cu12-12.4.127-py3-none-manylinux2014\_x86\_64.whl (24.6 MB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24.6/24.6 MB 32.4 MB/s eta 0:00:00

Downloading nvidia\_cuda\_runtime\_cu12-12.4.127-py3-none-manylinux2014\_x86\_64.whl (883 kB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 883.7/883.7 kB 43.0 MB/s eta 0:00:00

Downloading nvidia\_cudnn\_cu12-9.1.0.70-py3-none-manylinux2014\_x86\_64.whl (664.8 MB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 664.8/664.8 MB 2.8 MB/s eta 0:00:00

Downloading nvidia\_cufft\_cu12-11.2.1.3-py3-none-manylinux2014\_x86\_64.whl (211.5 MB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 211.5/211.5 MB 5.9 MB/s eta 0:00:00

Downloading nvidia\_curand\_cu12-10.3.5.147-py3-none-manylinux2014\_x86\_64.whl (56.3 MB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.3/56.3 MB 12.7 MB/s eta 0:00:00

Downloading nvidia\_cusolver\_cu12-11.6.1.9-py3-none-manylinux2014\_x86\_64.whl (127.9 MB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 127.9/127.9 MB 7.2 MB/s eta 0:00:00

Downloading nvidia\_cusparse\_cu12-12.3.1.170-py3-none-manylinux2014\_x86\_64.whl (207.5 MB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 207.5/207.5 MB 6.0 MB/s eta 0:00:00

Downloading nvidia\_nvjitlink\_cu12-12.4.127-py3-none-manylinux2014\_x86\_64.whl (21.1 MB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.1/21.1 MB 70.7 MB/s eta 0:00:00

Building wheels for collected packages: box2d-py

In [ ]:

**import** torch

**import** torch.nn **as** nn

**import** torch.optim **as** optim

**import** numpy **as** np

**import** gymnasium **as** gym

**import** matplotlib.pyplot **as** plt

# Описание реализованного алгоритма

В данной работе реализован классический алгоритм Actor-Critic на языке Python с использованием библиотеки PyTorch.  
Алгоритм разделяет две основные функции:

* **Actor** — принимает решение о том, какое действие выбрать, используя стратегию (policy).
* **Critic** — оценивает, насколько хорошим было текущее состояние, с помощью оценки ценности (value function).

В качестве среды используется CartPole-v0 из библиотеки OpenAI Gym.  
Цель агента — как можно дольше удерживать "шест" в вертикальном положении.

Обучение происходит в течение нескольких эпизодов, после чего отслеживается изменение получаемого вознаграждения.

In [ ]:

**class** Actor(nn**.**Module):

**def** \_\_init\_\_(self, state\_dim, action\_dim):

super(Actor, self)**.**\_\_init\_\_()

self**.**model **=** nn**.**Sequential(

nn**.**Linear(state\_dim, 128),

nn**.**ReLU(),

nn**.**Linear(128, action\_dim),

nn**.**Softmax(dim**=-**1)

)

**def** forward(self, state):

**return** self**.**model(state)

**class** Critic(nn**.**Module):

**def** \_\_init\_\_(self, state\_dim):

super(Critic, self)**.**\_\_init\_\_()

self**.**model **=** nn**.**Sequential(

nn**.**Linear(state\_dim, 128),

nn**.**ReLU(),

nn**.**Linear(128, 1)

)

**def** forward(self, state):

**return** self**.**model(state)

In [ ]:

env **=** gym**.**make("CartPole-v1")

state\_dim **=** env**.**observation\_space**.**shape[0]

action\_dim **=** env**.**action\_space**.**n

actor **=** Actor(state\_dim, action\_dim)

critic **=** Critic(state\_dim)

actor\_optimizer **=** optim**.**Adam(actor**.**parameters(), lr**=**1e-3)

critic\_optimizer **=** optim**.**Adam(critic**.**parameters(), lr**=**1e-3)

gamma **=** 0.99

all\_rewards **=** []

**for** episode **in** range(500):

state, \_ **=** env**.**reset()

state **=** torch**.**FloatTensor(state)

total\_reward **=** 0

**for** t **in** range(500):

probs **=** actor(state)

dist **=** torch**.**distributions**.**Categorical(probs)

action **=** dist**.**sample()

next\_state, reward, terminated, truncated, \_ **=** env**.**step(action**.**item())

done **=** terminated **or** truncated

next\_state **=** torch**.**FloatTensor(next\_state)

td\_target **=** reward **+** gamma **\*** critic(next\_state) **\*** (1 **-** int(done))

td\_error **=** td\_target **-** critic(state)

*# Обновление Critic*

critic\_loss **=** td\_error**.**pow(2)

critic\_optimizer**.**zero\_grad()

critic\_loss**.**backward()

critic\_optimizer**.**step()

*# Обновление Actor*

actor\_loss **=** **-**dist**.**log\_prob(action) **\*** td\_error**.**detach()

actor\_optimizer**.**zero\_grad()

actor\_loss**.**backward()

actor\_optimizer**.**step()

state **=** next\_state

total\_reward **+=** reward

**if** done:

**break**

all\_rewards**.**append(total\_reward)

**if** episode **%** 50 **==** 0:

print(f"Эпизод {episode}, вознаграждение: {total\_reward}")

Эпизод 0, вознаграждение: 29.0

Эпизод 50, вознаграждение: 21.0

Эпизод 100, вознаграждение: 38.0

Эпизод 150, вознаграждение: 60.0

Эпизод 200, вознаграждение: 30.0

Эпизод 250, вознаграждение: 47.0

Эпизод 300, вознаграждение: 141.0

Эпизод 350, вознаграждение: 113.0

Эпизод 400, вознаграждение: 59.0

Эпизод 450, вознаграждение: 120.0

# График средней награды

На графике ниже представлена динамика изменения вознаграждения по эпизодам.  
Также добавлено скользящее среднее по 10 эпизодам для наглядности.  
Это позволяет визуально убедиться в успешности обучения агента.

In [ ]:

**import** matplotlib.pyplot **as** plt

**import** numpy **as** np

*# Сглаживание: скользящее среднее по 10 эпизодам*

**def** moving\_average(values, window**=**10):

**return** np**.**convolve(values, np**.**ones(window)**/**window, mode**=**'valid')

smoothed\_rewards **=** moving\_average(all\_rewards)

plt**.**figure(figsize**=**(10, 5))

plt**.**plot(all\_rewards, label**=**"Награда (сырые значения)", alpha**=**0.3)

plt**.**plot(range(len(smoothed\_rewards)), smoothed\_rewards, label**=**"Сглаженная награда", linewidth**=**2)

plt**.**title("Динамика вознаграждения по эпизодам")

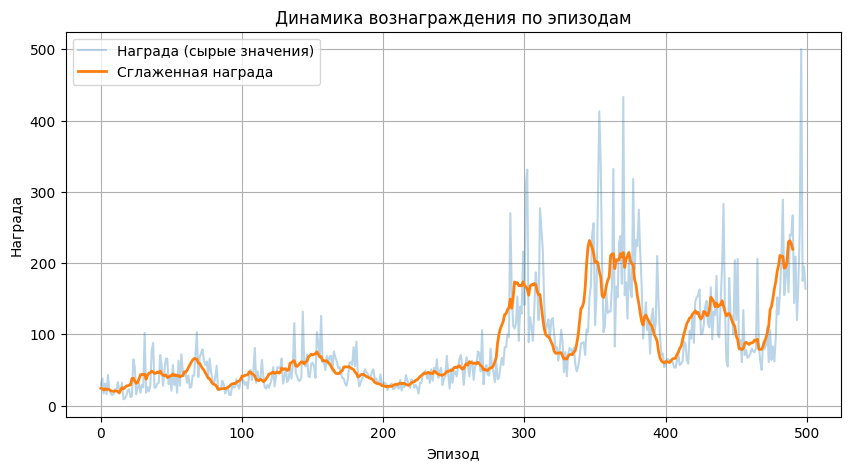
plt**.**xlabel("Эпизод")

plt**.**ylabel("Награда")

plt**.**legend()

plt**.**grid(**True**)

plt**.**show()



In [ ]:

plt**.**plot(all\_rewards)

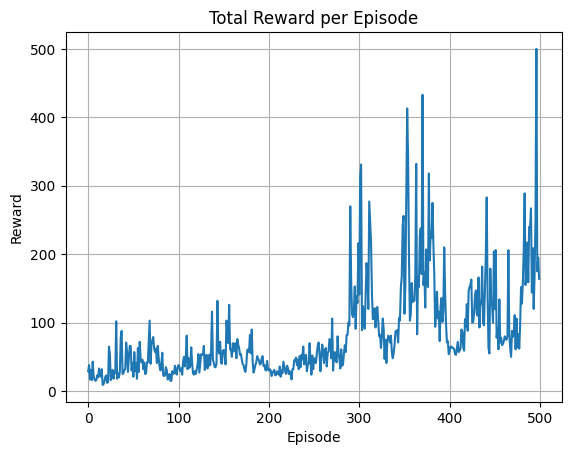
plt**.**title("Total Reward per Episode")

plt**.**xlabel("Episode")

plt**.**ylabel("Reward")

plt**.**grid()

plt**.**show()



# Вывод

В данной лабораторной работе был реализован базовый алгоритм семейства Actor-Critic с использованием библиотеки PyTorch и среды CartPole-v0 из OpenAI Gym.  
Модель состояла из двух компонентов:

* **Актор (Actor)**, который выбирает действия на основе стратегии;
* **Критик (Critic)**, который оценивает действия, предоставляя сигнал обучения.

В ходе эксперимента агент продемонстрировал способность к обучению, о чём свидетельствует рост средней награды по эпизодам.  
Таким образом, цель работы достигнута: алгоритм Actor-Critic успешно применён на выбранной задаче управления.

Был обучен агент, чтобы он научился управлять тележкой с шестом в среде CartPole. Его задача — удерживать шест в вертикальном положении как можно дольше.

Для этого был использован алгоритм Actor-Critic — это метод обучения с подкреплением, который сочетает:

Actor — принимает решения: какое действие сейчас лучше выполнить.

Critic — оценивает действия и говорит, насколько они были удачными.