

DQN. Отчет

Содержание

DQN. Отчет1

1. Обучить Агента решать Acrobot-v1, MountainCar-v0, или LunarLander-v2 (одну на выбор) методом DQN. Найти оптимальные гиперпараметры. Сравнить с алгоритмом Deep Cross-Entropy на графиках. 2

 Вывод: 3

2. Реализовать с сравнить (на выбранной ранее среде) друг с другом и с обычным DQN следующие его модификации: 4

- DQN с Hard Target Update; 4
- DQN с Soft Target Update;..... 4
- Double DQN..... 4

 Вывод: 6

1. Обучить Агента решать Acrobot-v1, MountainCar-v0, или LunarLander-v2 (одну на выбор) методом DQN. Найти оптимальные гиперпараметры. Сравнить с алгоритмом Deep Cross-Entropy на графиках.

Выбранная игра: **Acrobot-v1**.

Результаты можно посмотреть в [ClearML](#)

Исследуемые гиперпараметры для алгоритма **DQN** в 10 версиях:

- **gamma**: [0.9, 0.99]
- **learning rate**: [0.1, 0.01, 0.001]
- **batch size**: [64, 128, 256]
- **epsilon decrease**: [0.01, 0.001]
- **epsilon min**: [0.1, 0.01, 0.001]

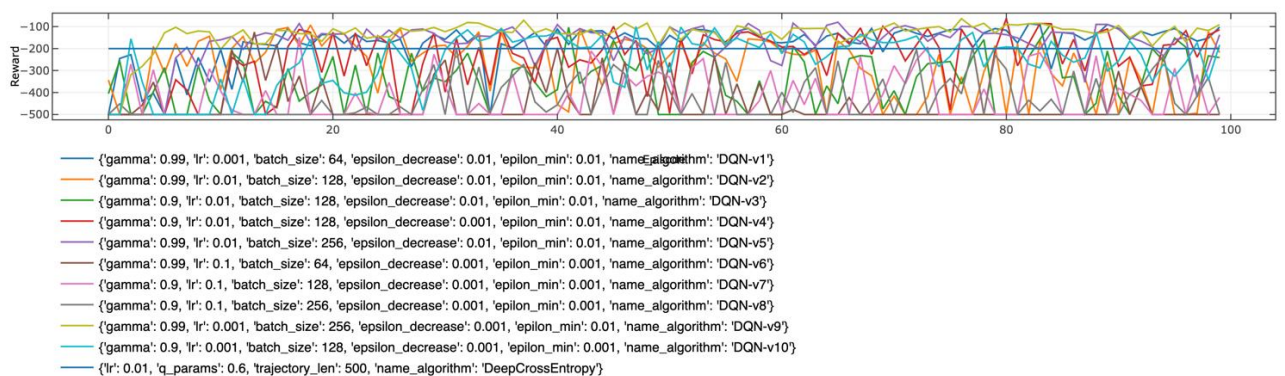
Также было произведено сравнение с алгоритмов **Deep Cross Entropy** (DCEM) с лучшими параметрами для данной игры (см. дз-2):

- **learning rate**: 0.01
- **q params**: 0.6
- **trajectory length**: 500

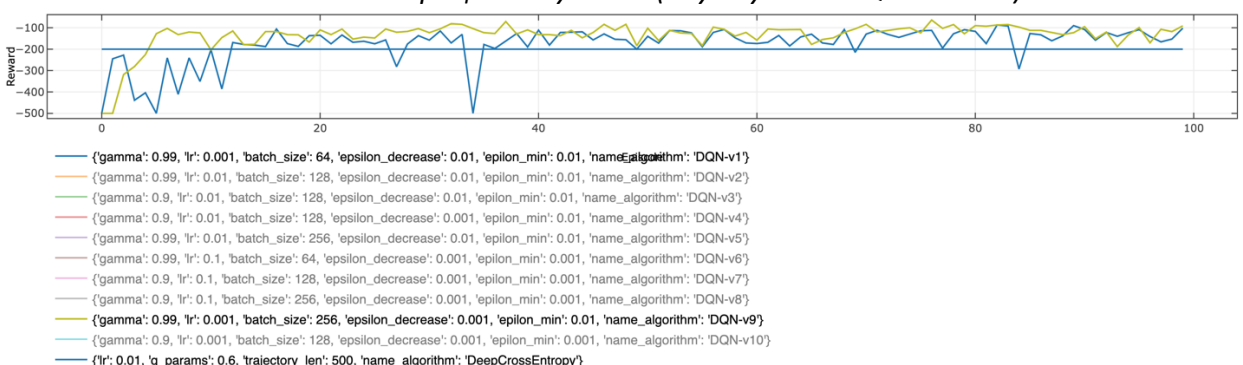
Общие гиперпараметры для алгоритмов:

- **episode_n (epochs)**: 100
- **t_max (максимальное кол-во итераций для одной игры)**: 500 (из документации gym)

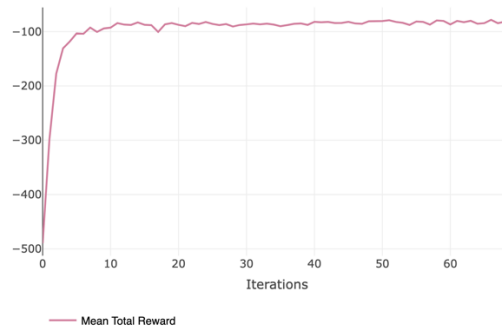
Reward на итерациях обучения:



Reward на итерациях обучения (двух лучших DQN и DCEM):



Reward на итерациях обучения для DCEM из дз-2:



*PS при запуске DCEM почему-то показывал константу на графике, поэтому для него также приведет график из дз-2

Вывод:

Лучший результат DQN достиг при следующих гиперпараметрах:

- **gamma:** 0.99
- **learning rate:** 0.001
- **batch size:** 256
- **epsilon decrease:** 0.001
- **epsilon min:** 0.1

Так как при обучении ведет себя более стабильно без выбросов. Однако на мой взгляд, если сравнивать графики обучения DCEM из ДЗ-2, DCEM показывает более лучшую и быструю сходимость, поэтому для этой задачи можно отдать пальму первенства ему. Возможно, стоило дополнительно настроить Q-function, чтобы результаты DQN стали лучше, чем DCEM, но всё же DQN не сильно уступает DCEM и тоже сходится.

2. Реализовать и сравнить (на выбранной ранее среде) друг с другом и с обычным DQN следующие его модификации:

- DQN с Hard Target Update;
- DQN с Soft Target Update;
- Double DQN.

Выбранная игра: **Acrobot-v1**.

Результаты можно посмотреть в [ClearML](#)

Исследуемые гиперпараметры для алгоритма **DQN с Hard Target Update** в 10 версиях:

- **gamma:** 0.99
- **learning rate:** 0.01
- **batch size:** 64
- **epsilon decrease:** 64
- **epsilon min:** 0.01
- **q_function_update_epoch** (обновляем зафиксированную q-function между эпохами): [None, 1, 10, 50]
- **q_function_update_step** (обновляем зафиксированную q-function между шагами, то есть может обновиться внутри одной эпохи обучения): [None, 1, 50, 250]

Исследуемые гиперпараметры для алгоритма **DQN с Soft Target Update** в 10 версиях:

- **gamma:** 0.99
- **learning rate:** 0.01
- **batch size:** 64
- **epsilon decrease:** 64
- **epsilon min:** 0.01
- **tau:** [0.01, 0.1, 0.5, 0.9]

Исследуемые гиперпараметры для алгоритма **Double DQN** в 10 версиях:

- **gamma:** 0.99
- **learning rate:** 0.01
- **batch size:** 64
- **epsilon decrease:** 64
- **epsilon min:** 0.01
- **tau:** [0.01, 0.1, 0.5, 0.9]

Описанные выше алгоритмы сравнивались с **DQN**:

- **gamma:** 0.99
- **learning rate:** 0.01
- **batch size:** 64
- **epsilon decrease:** 64
- **epsilon min:** 0.01

Legend:

- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'name_algorithm': 'DQN-best-result'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_epoch': 1, 'q_fix_update_step': None, 'name_algorithm': 'DQN-Hard-Target-Update-v1'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_epoch': 10, 'q_fix_update_step': None, 'name_algorithm': 'DQN-Hard-Target-Update-v2'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_epoch': 50, 'q_fix_update_step': None, 'name_algorithm': 'DQN-Hard-Target-Update-v3'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_epoch': 10, 'q_fix_update_step': 10, 'name_algorithm': 'DQN-Hard-Target-Update-v4'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_step': 50, 'q_fix_update_epoch': None, 'name_algorithm': 'DQN-Hard-Target-Update-v5'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_step': 250, 'q_fix_update_epoch': None, 'name_algorithm': 'DQN-Hard-Target-Update-v6'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'tau': 0.01, 'name_algorithm': 'DQN-Soft-Target-Update-v1'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'tau': 0.1, 'name_algorithm': 'DQN-Soft-Target-Update-v2'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'tau': 0.5, 'name_algorithm': 'DQN-Soft-Target-Update-v3'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'tau': 0.9, 'name_algorithm': 'DQN-Soft-Target-Update-v4'`)

Figure 1: A line plot showing the average episode reward over 200 episodes for various DQN algorithms. The y-axis is 'Reward' ranging from -500 to -100. The x-axis is 'Episode' ranging from 0 to 200. A horizontal dashed line at approximately -100 represents the 'DQN-best-result'. The legend lists 14 different algorithms with their specific hyperparameters. Most algorithms show a rapid increase in reward from episode 0 to 20, then stabilize. The 'DQN-best-result' is achieved by the algorithm with parameters: {'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'name_algorithm': 'DQN-best-result'}.

Legend:

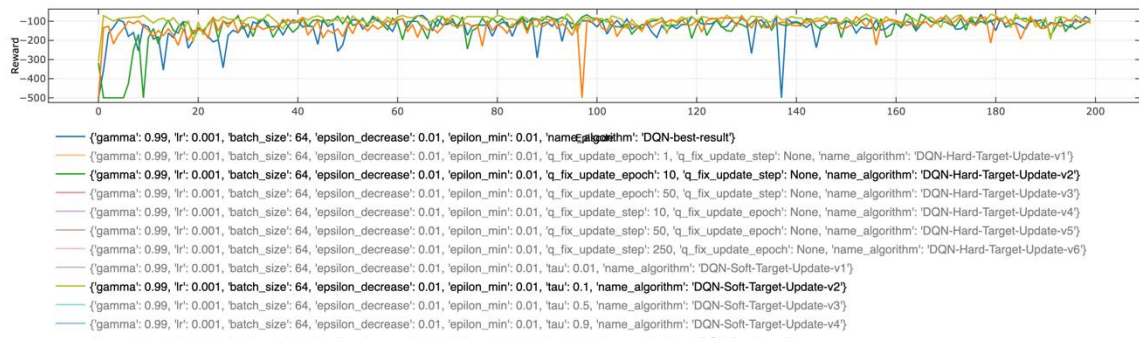
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'name_algorithm': 'DQN-best-result'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_epoch': 1, 'q_fix_update_step': None, 'name_algorithm': 'DQN-Hard-Target-Update-v1'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_epoch': 10, 'q_fix_update_step': None, 'name_algorithm': 'DQN-Hard-Target-Update-v2'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_epoch': 50, 'q_fix_update_step': None, 'name_algorithm': 'DQN-Hard-Target-Update-v3'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_epoch': 10, 'q_fix_update_step': 10, 'name_algorithm': 'DQN-Hard-Target-Update-v4'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_epoch': 50, 'q_fix_update_step': 50, 'name_algorithm': 'DQN-Hard-Target-Update-v5'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_epoch': 250, 'q_fix_update_step': None, 'name_algorithm': 'DQN-Hard-Target-Update-v6'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'tau': 0.01, 'name_algorithm': 'DQN-Soft-Target-Update-v1'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'tau': 0.1, 'name_algorithm': 'DQN-Soft-Target-Update-v2'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'tau': 0.5, 'name_algorithm': 'DQN-Soft-Target-Update-v3'`)
- (`'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'tau': 0.9, 'name_algorithm': 'DQN-Soft-Target-Update-v4'`)

Figure 1: Performance of different DQN configurations. The plot shows the reward over 200 episodes for 12 different configurations. The 'DQN-best-result' configuration (blue line) achieves the highest reward, stabilizing around -100. Other configurations show varying degrees of performance, with some exhibiting significant drops or oscillations.

Legend:

- {'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'name_algorithm': 'DQN-best-result'}
- {'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_epoch': 1, 'q_fix_update_step': None, 'name_algorithm': 'DQN-Hard-Target-Update-v1'}
- {'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_epoch': 10, 'q_fix_update_step': None, 'name_algorithm': 'DQN-Hard-Target-Update-v2'}
- {'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_epoch': 50, 'q_fix_update_step': None, 'name_algorithm': 'DQN-Hard-Target-Update-v3'}
- {'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_step': 10, 'q_fix_update_epoch': None, 'name_algorithm': 'DQN-Hard-Target-Update-v4'}
- {'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_step': 50, 'q_fix_update_epoch': None, 'name_algorithm': 'DQN-Hard-Target-Update-v5'}
- {'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'q_fix_update_step': 250, 'q_fix_update_epoch': None, 'name_algorithm': 'DQN-Hard-Target-Update-v6'}
- {'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'tau': 0.01, 'name_algorithm': 'DQN-Soft-Target-Update-v1'}
- {'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'tau': 0.1, 'name_algorithm': 'DQN-Soft-Target-Update-v2'}
- {'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'tau': 0.5, 'name_algorithm': 'DQN-Soft-Target-Update-v3'}
- {'gamma': 0.99, 'lr': 0.001, 'batch_size': 64, 'epsilon_decrease': 0.01, 'epsilon_min': 0.01, 'tau': 0.9, 'name_algorithm': 'DQN-Soft-Target-Update-v4'}

Reward на итерациях обучения (оригинальный DQN и лучшие модификации этого алгоритма):



Вывод:

Самые **стабильные результаты и более быструю сходимость** показал алгоритм DQN в модификации **Soft Target Update** с гиперпараметрами:

- **gamma:** 0.99
- **learning rate:** 0.01
- **batch size:** 64
- **epsilon decrease:** 64
- **epsilon min:** 0.01
- **tau:** 0.1

DQN в модификации **Hard Target Update** так же показывает лучше результаты, чем обычный DQN. Лучшие гиперпараметры:

- **gamma:** 0.99
- **learning rate:** 0.01
- **batch size:** 64
- **epsilon decrease:** 64
- **epsilon min:** 0.01
- **q_function_update_epoch:** 10
- **q_function_update_step:** None

Double DQN в целом показывает лучше результаты, чем обычный DQN, так как ведет себя во время обучения более стабильно, но уступает двум другим модификациям этого алгоритма. Лучшие гиперпараметры:

- **gamma:** 0.99
- **learning rate:** 0.01
- **batch size:** 64
- **epsilon decrease:** 64
- **epsilon min:** 0.01
- **tau:** 0.01

Все модификации показали себя лучше, чем обычный DQN, так как они более стабильные и быстрее сходятся во время обучения.