Трапер Максим. СПБГУ. Магистратура 1 курс «Искусственный интеллект и наука о данных»

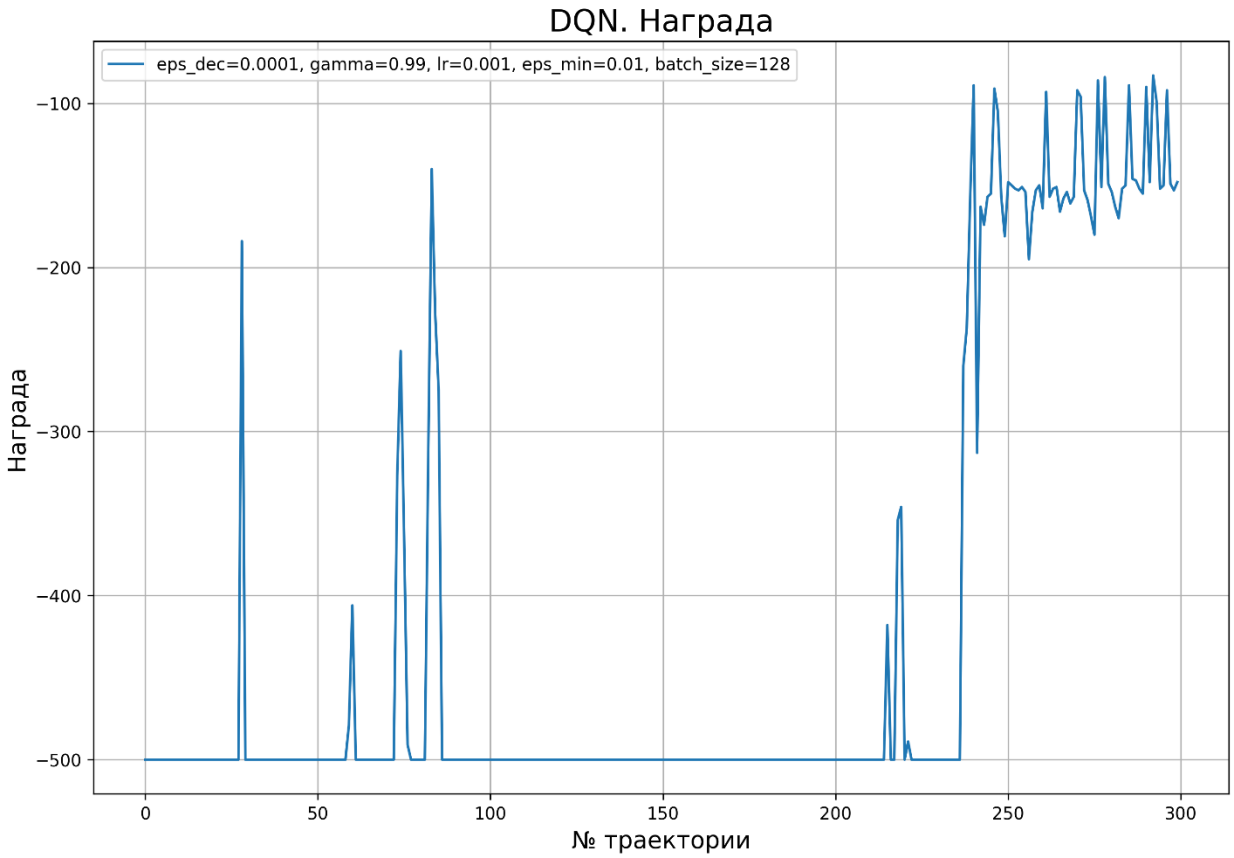
P.S: задание выполнено не Бог весть как, так что можно просмотреть по диагонали. Отправил больше для галочки, что что-то да сделал.

TL:DR: в первом задании сколько провёл кучу экспериментов, а по итогу выбрал бейзлайн параметры из первого эксперимента – лучшими (изменил только batch_size со 128 до 1024 и hidden_size с 64 до 128). Сравнить с DCEM не успел…

Во втором задании, к сожалению, модифицированные методы показали сходное качество с DQN, пусть и достигли его раньше.

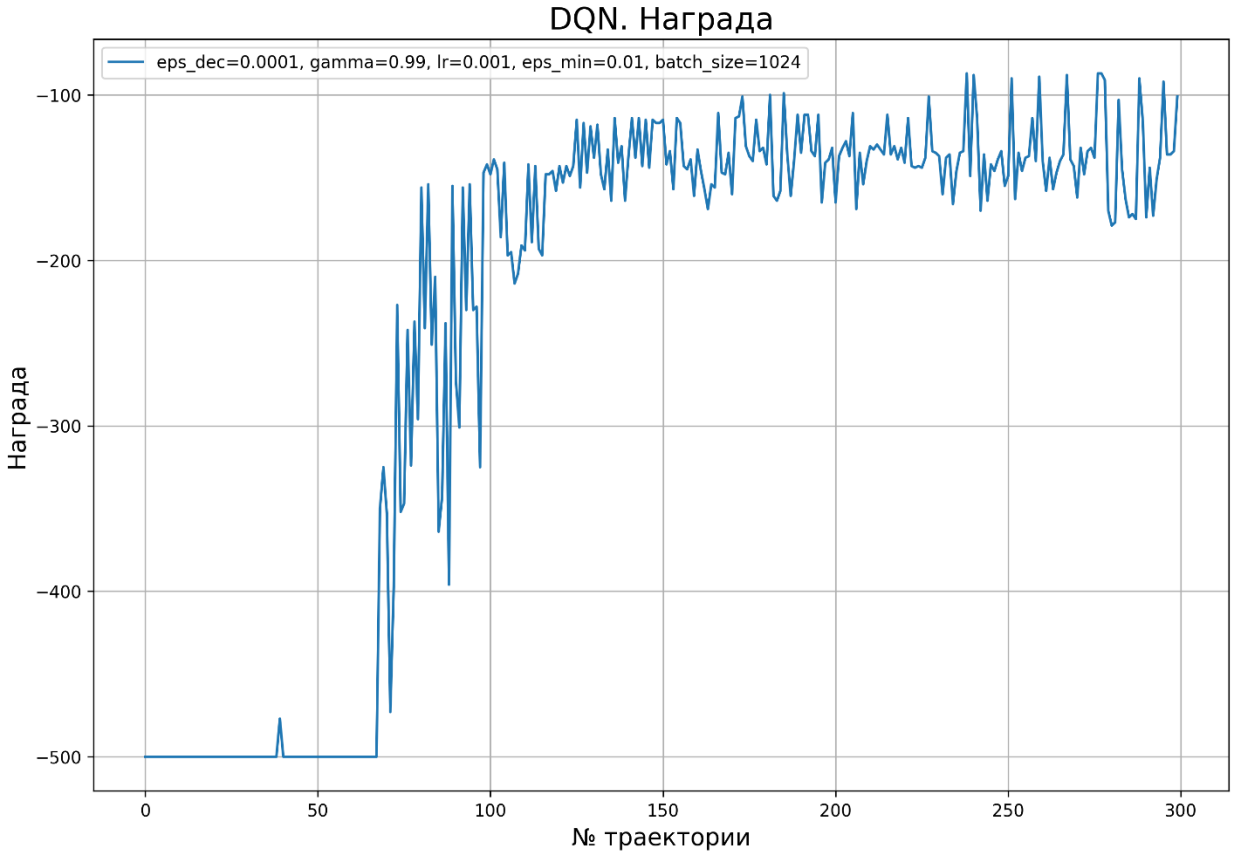
Задание №1: обучить Агента решать MountainCar-v0 методом DQN. Найти оптимальные гиперпараметры. Сравнить с алгоритмом Deep Cross-Entropy на графиках. Помним про корректность сравнения.

Эксперимент №1: берём исходные от практического занятия параметры. Trajectory_n = 300, trajectory_len = 500, epsilon_decrease = 1e-4, gamma = 0.99, batch_size = 128, lr = 1e-3, epsilon_min = 1e-2



Вывод: нуу.. попробовали, ладно. Будем брать как бейзлайн. Обучение конечно не очень стабильное, мягко говоря.

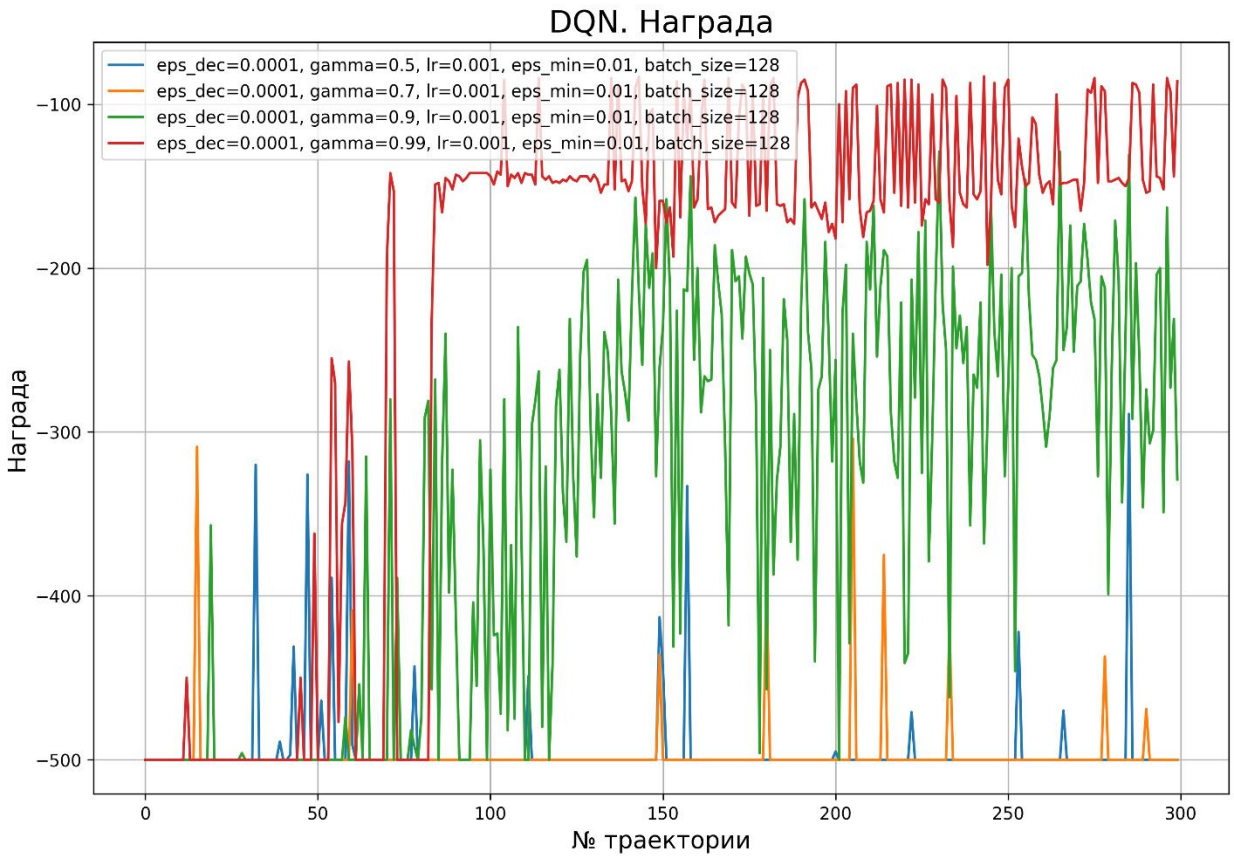
Эксперимент №2: Теперь возьмём Batch_size = 1024. Остальное остаётся прежним: Trajectory_n = 300, trajectory_len = 500, epsilon_decrease = 1e-4, gamma = 0.99, lr = 1e-3, epsilon_min = 1e-2.



Вывод: ну понятно, что это больше параметр нейронной сети для более стабильного обучения, заняло это в 2 раза больше времени. Но самое главное, что пусть и приближенно, но обучение модели всё равно качественно не преодолело потолок в награду = -100, как и в прошлом пункте. Так что пока остановлюсь на размере батча = 128, чтобы было просто быстрее.

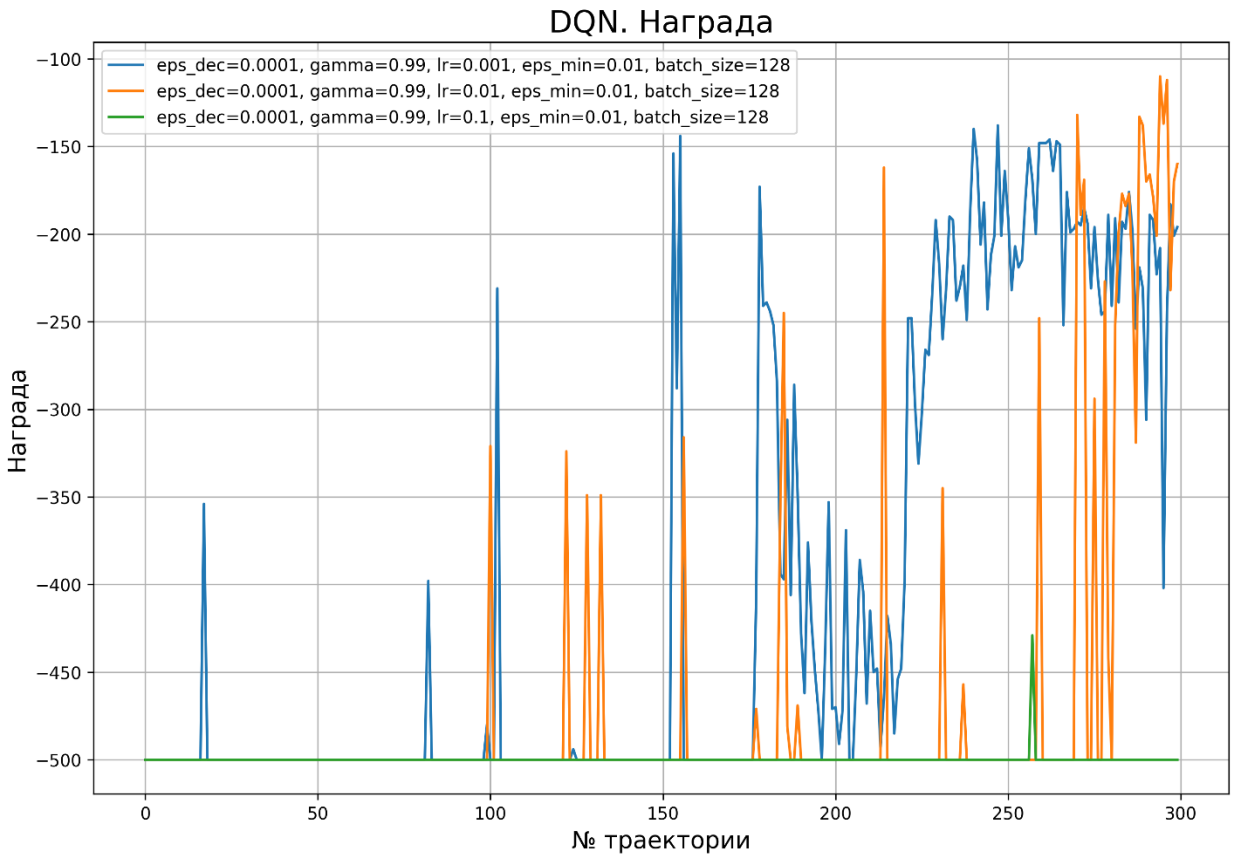
Эксперимент №3: Переберём gammas = [0.5, 0.7, 0.9, 0.99]. Остальное остаётся прежним: Trajectory_n = 300, trajectory_len = 500, epsilon_decrease = 1e-4 , lr = 1e-3, epsilon_min = 1e-2, batch_size = 128.

gammas = [0.5, 0.7, 0.9, 0.99]



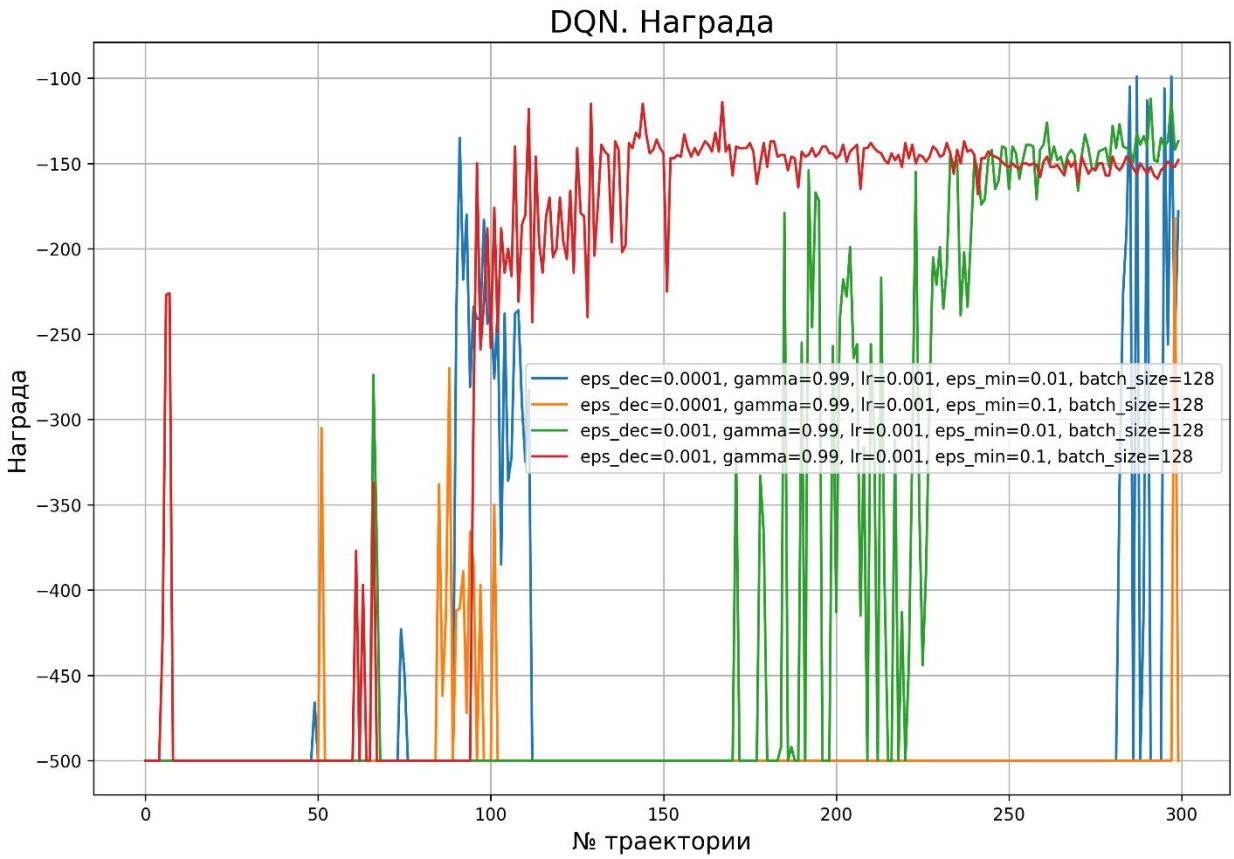
Вывод: однозначно лучше изначально значение gamma = 0.99. Оставлю его. Может стоило ещё попробовать побольше взять. Дальше постараюсь попробовать.

Эксперимент №4: Переберём lrs = [1e-3, 1e-2, 1e-1]. Остальное остаётся прежним: Trajectory_n = 300, trajectory_len = 500, epsilon_decrease = 1e-4, epsilon_min = 1e-2, batch_size = 128, gamma = 0.99.



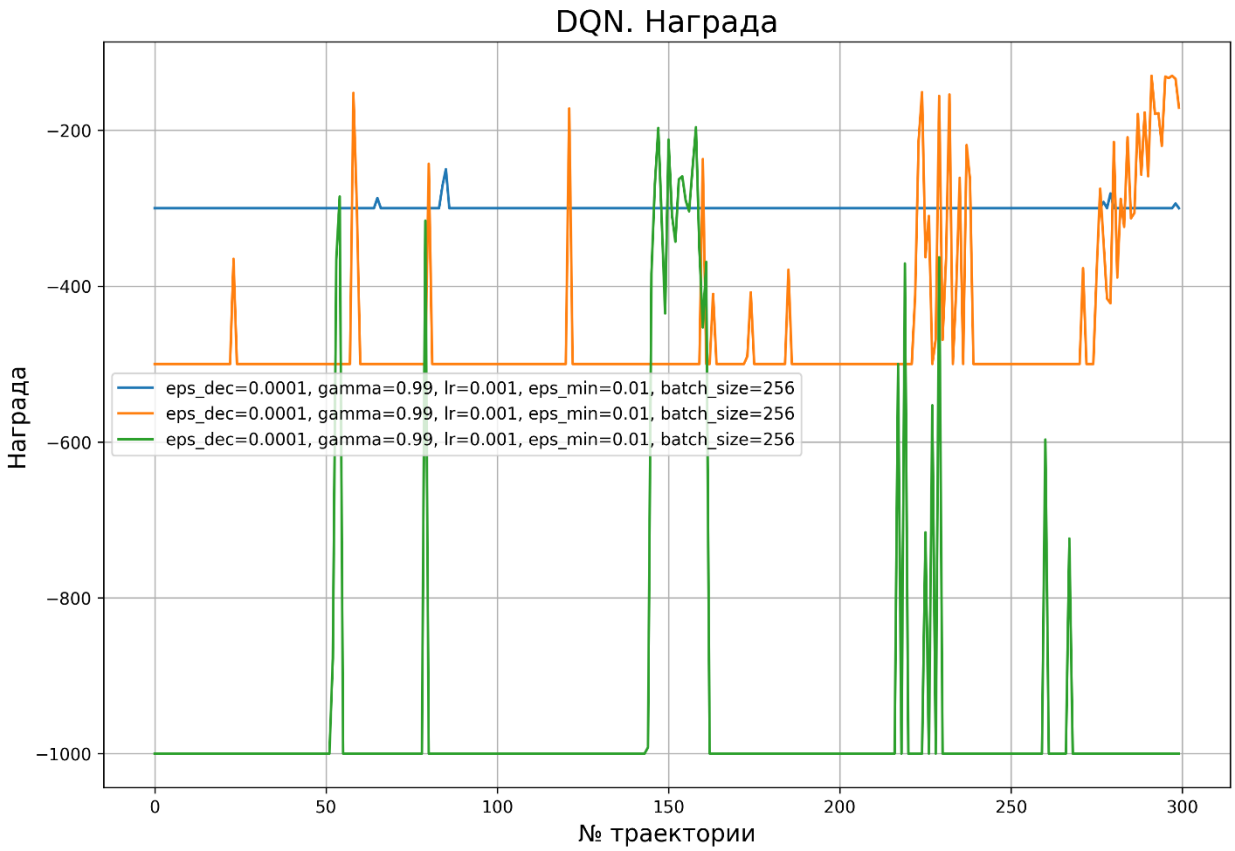
Вывод: по существу, я удивлён, что нет какого-то кратного преимущества. В среднем, даже при таком шумном обучении, изначальная скорость обучения показала себя лучше. Оставляем её (пока ничего по итогу не сменил, за 3 эксперимента)))

Эксперимент №5: Переберём epsilons_decrease = [1e-4, 1e-3] и epsilons_min = [1e-2, 1e-1]. Остальное остаётся прежним: Trajectory_n = 300, trajectory_len = 500, lr = 1e-3, batch_size = 128, gamma = 0.99.



Вывод: нуу.. так-то вывода нет). Красная линия, по сути, показала себя лучше всего (эпсилон падает быстрее до 1e-1), т.к. стабильнее и раньше всех показала высокие оценки, но я не уверен, что это очень хорошо. Оставлю пока оба параметра без изменений.

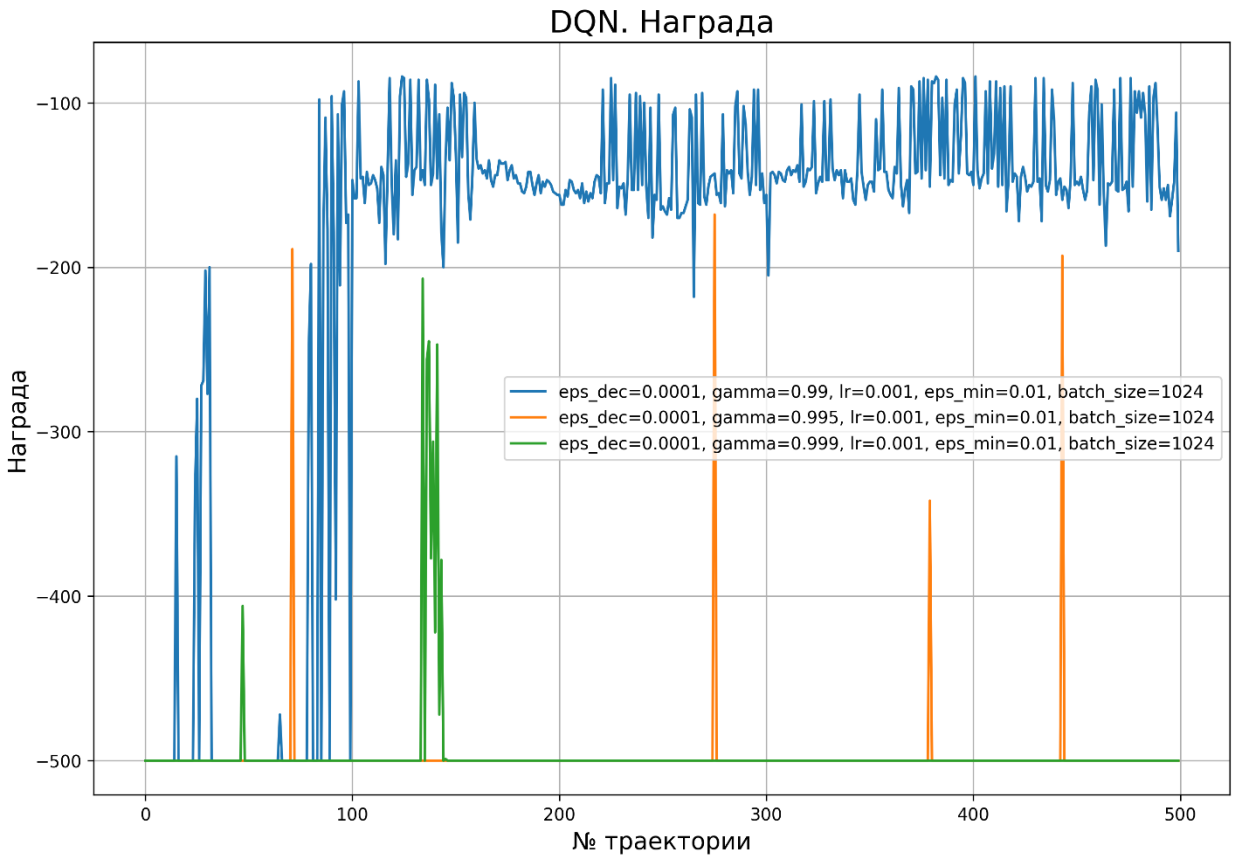
Эксперимент №6: Переберём trajectorys_len = [300, 500, 1000], поднимем батч до 256 для более устойчивого обучения. Остальное остаётся прежним: Trajectory_n = 300, epsilon_decrease = 1e-4, epsilon_min = 1e-2, gamma = 0.99, lr = 1e-3.



Вывод: я забыл написать в легенде длину и количество траекторий)). Короче, зеленая – длина = 1000, оранжевая – длина = 500, синяя – длина = 300.

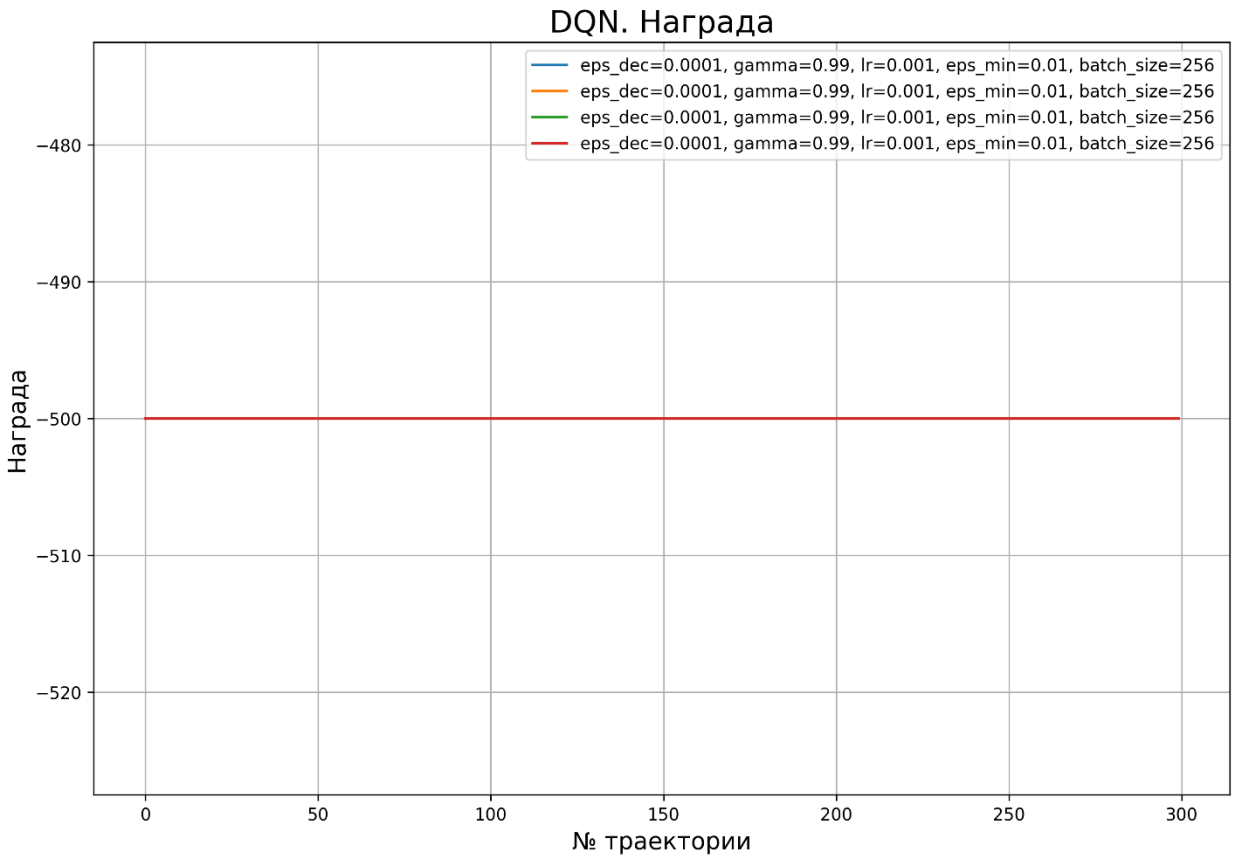
300 шагов не хватает, конечно, а 1000 видимо с переизбытком. Короче оставляем опять всё, как есть)))

Эксперимент №7: Переберём gammas = [0.99, 0.995, 0.999], поднимем батч до 1024, количество сэмплируемых траекторий поднимем до 500. Остальное остаётся прежним: epsilon_decrease = 1e-4, epsilon_min = 1e-2, gamma = 0.99, lr = 1e-3.

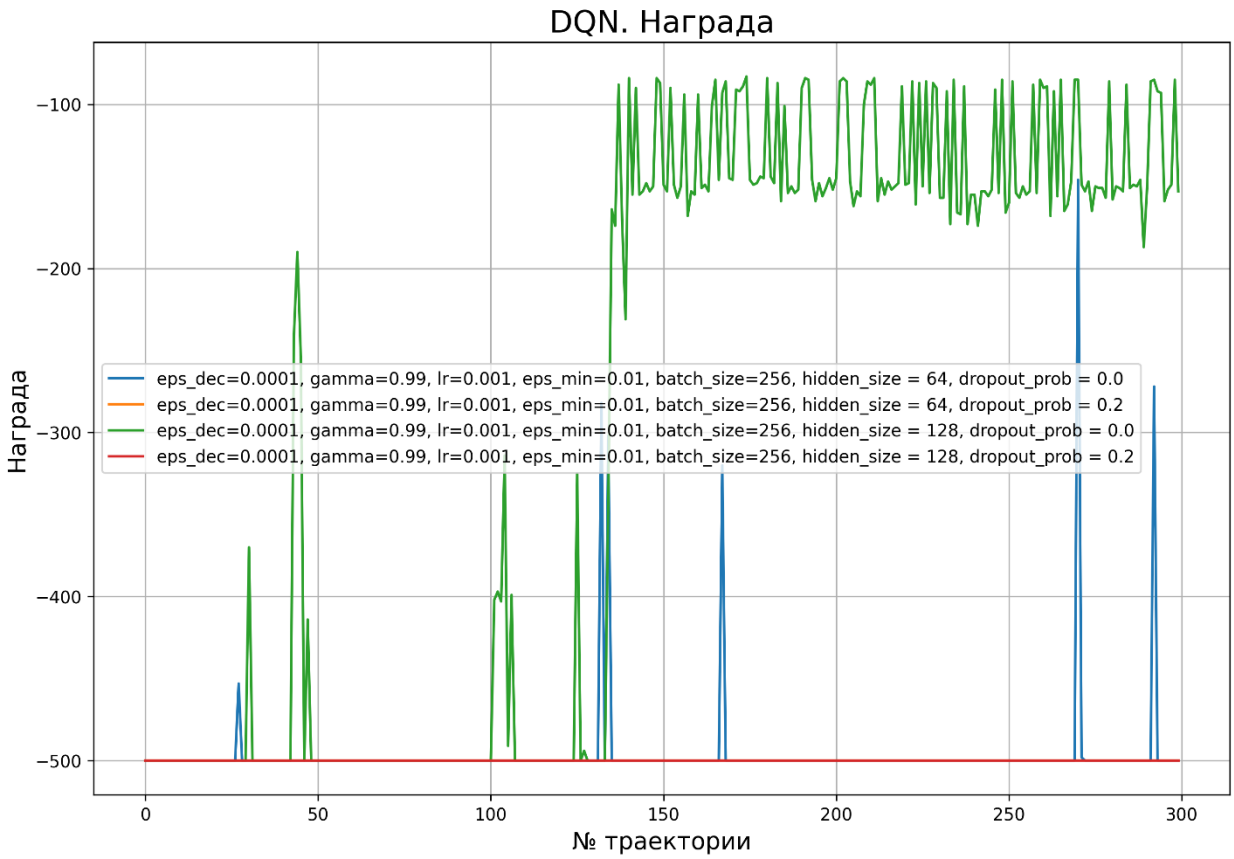


Вывод: всё по старинке, всё остаётся))). Лучшее качество дал изначально взятый gamma.

Эксперимент №8: Попробую чуть-чуть поменять архитектуру самой сети. Переберём кол-во нейронов в скрытом слое: hidden_size = [64, 128] и добавим dropout = [0.2, 0.5]. epsilon_decrease = 1e-4, epsilon_min = 1e-2, gamma = 0.99, lr = 1e-3, batch_size = 256, trajectory_n = 300, trajectory_len = 500.



Вывод: а какой вывод?) Я не знаю, что за глупость случилась, и даже то, что я забыл вписать перебираемые параметры в легенду, не мешает однозначно определить, какой график был отрисован с какими параметрами…

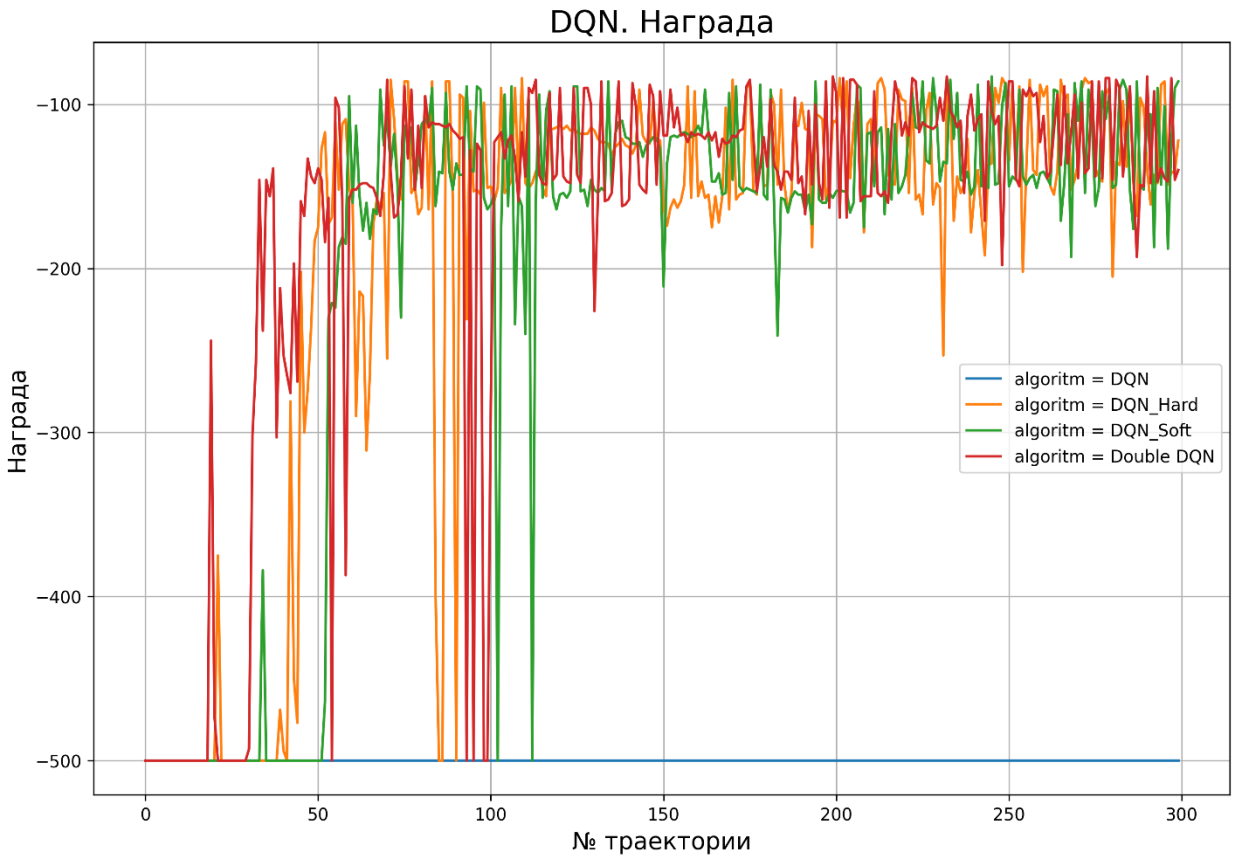


В общем, видимо, dropout не очень полезен в данной задаче. Не знаю..

Лучшие параметры: в общем, остаюсь на бейзлайн параметрах. Единственное сделаю batch_size = 1024 и hidden_size = 128. Trajectory_n = 300, trajectory_len = 500, epsilon_decrease = 1e-4, gamma = 0.99, lr = 1e-3, epsilon_min = 1e-2

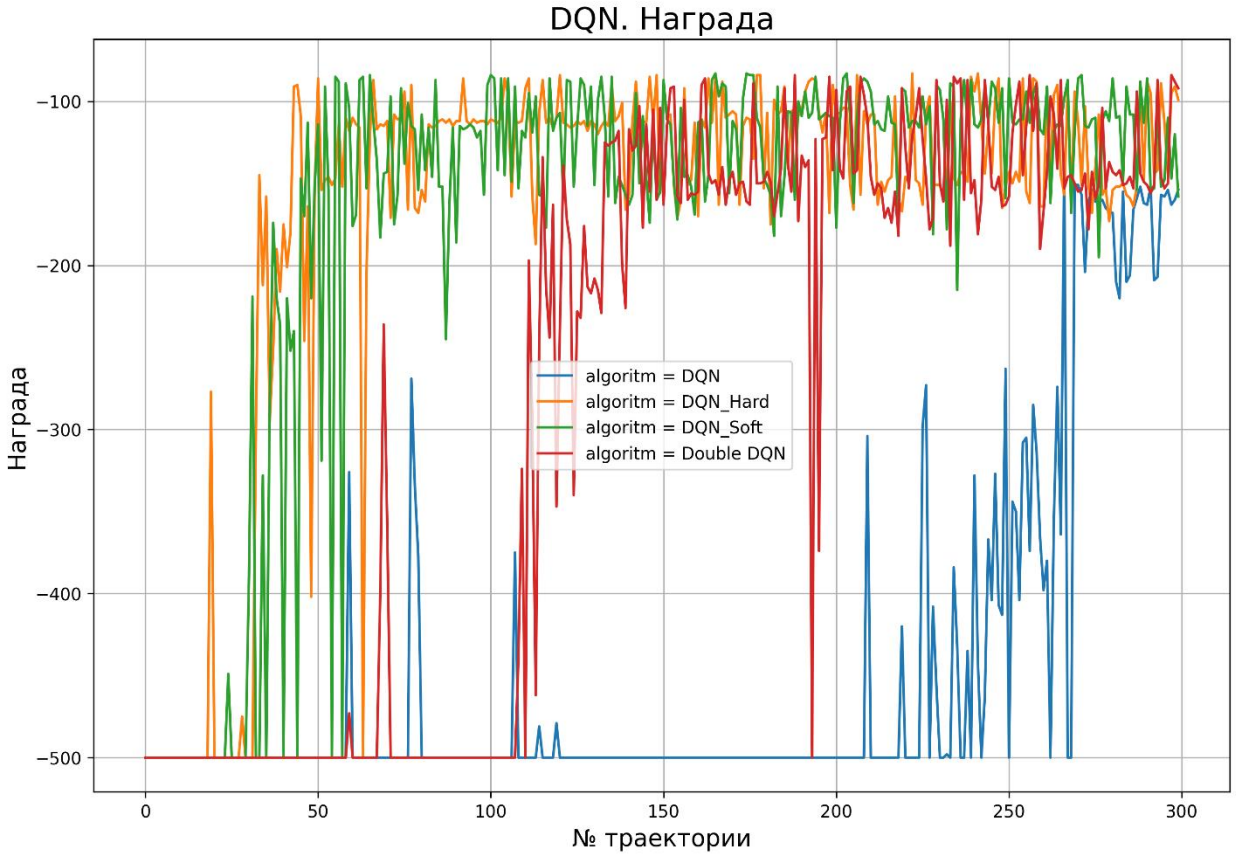
Задание №2: реализовать с сравнить (на выбранной ранее среде) друг с другом и с обычным DQN следующие его модификации: DQN с Hard Target Update; DQN с Soft Target Update; Double DQN.

Эксперимент №1: tau = 0.01, target_update_freq = 100. Всё остальное по старинке



Вывод: ну методы-то себя лучше показали, чем обычный DQN. Хотя если вспомнить предыдущие эксперименты, ну улучшений-то нет. Хотя ожидались. Попробую чутка поменять значения tau и частоту обновления таргета

Эксперимент №2: tau = 0.1, target_update_freq = 50. Всё остальное по старинке


DQN. Награда
Награда
№ траектории
algoritm = DQN
algoritm = DQN_Hard
algoritm = DQN_Soft
algoritm = Double DQN
−100
−200
−300
−400
−500
0
50
100
150
200
250
300