МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

КУРСОВАЯ РАБОТА

по дисциплине «Обучение с подкреплением»

Тема: Исследование архитектур DQN

Студент гр. 0310 Бодунов П.А.

Студент гр. 0310 Середенков А.А.

Преподаватель Глазунов С.А.

Санкт-Петербург

2025

ЗАДАНИЕ

НА КУРСОВУЮ РАБОТУ

Студент Бодунов П.А., Середенков А.А.

Группа 0310

Тема работы: Исследование архитектур DQN

Исходные данные:

Необходимо реализовать и сравнить между собой следующие версии DQN:

- Dueling networks
- Multi-step learning
- Distributional RL

В качестве окружения для тестирования:

- LunarLander-v3
- Mountain-Car

Содержание пояснительной записки:

«Содержание», «Введение», «Среды обучения», «Архитектуры DQN», «Сравнение архитектур», «Заключение»

Предполагаемый объем пояснительной записки:

Не менее 10 страниц.

Дата выдачи задания: 11.02.2025

Дата сдачи реферата: 29.05.2025

Дата защиты реферата: 29.05.2025

Студент		Бодунов П.А.
Студент		Середенков А.А.
Преподаватель		Глазунов С.А.

АННОТАЦИЯ

В данной работе реализуются и сравниваются три модификации алгоритма DQN: Dueling Networks, Multi-step Learning и Distributional RL. Тестирование проводится в средах LunarLander-v3 и MountainCar, чтобы оценить эффективность каждого подхода. Результаты экспериментов позволяют определить, какие модификации лучше справляются с задачами управления в различных условиях.

SUMMARY

In this paper, three modifications of the DQN algorithm are implemented and compared: Dueling Network, Multi-step Learning, and Distributive RL. Testing is conducted in LunarLander-v3 and Mountain Car environments to evaluate the effectiveness of each approach. The experimental results allow us to determine which modifications are better able to cope with management tasks in various conditions.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время нейронные сети стремительно развиваются и находят применение во многих сферах жизни — от поиска информации до предоставления рекомендаций с использованием различных моделей. Для создания таких моделей требуется процесс обучения, в том числе с применением алгоритмов обучения с подкреплением. Одним из таких алгоритмов является DQN (Deep Q-Learning), который стал значительным шагом вперед по сравнению с классическим Q-Learning, решив проблемы масштабируемости и стабильности. Однако и сам DQN продолжает совершенствоваться. В данной работе рассматриваются усовершенствованные версии DQN, такие как Dueling Networks, Multi-step Learning и Distributional RL, на примере сред LunarLander-v3 и MountainCar.

1. СРЕДЫ ОБУЧЕНИЯ

1.1. Среда LunarLander-v3

В данной среде решается классическая задача оптимизации траектории полёта ракеты. Согласно принципу максимума Понтрягина, оптимальным решением будет либо запустить двигатель на полную мощность, либо выключить его. Именно поэтому в этой среде выполняются дискретные действия: включение или выключение двигателя.

Существует два варианта среды: дискретный и непрерывный. Посадочная площадка всегда находится в точке с координатами (0,0). Координаты — это первые два числа в векторе состояния. Возможна посадка за пределами посадочной площадки. Запас топлива неограничен, поэтому агент может научиться летать и приземлиться с первой попытки. Визуальное представление среды представлено на рис. 1.

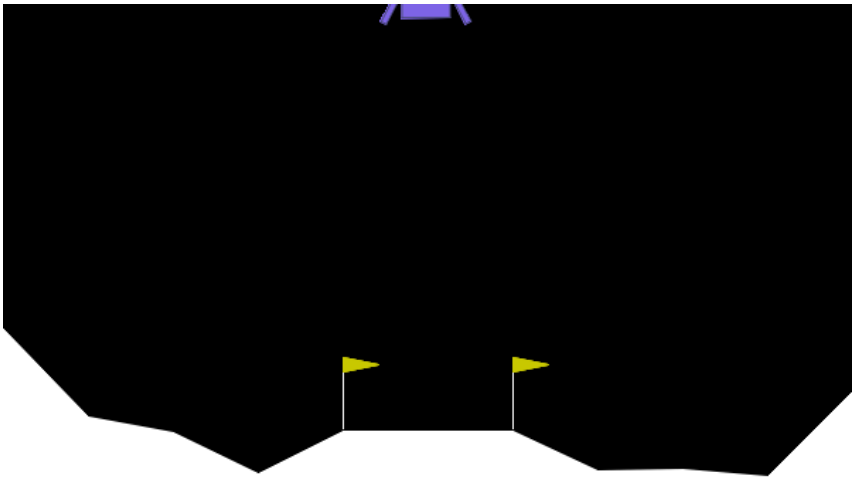

Рисунок 1 - Визуальное представление среды LunarLander-v3

Пространство действий состоит из 4 значений:

- 0: Ничего не делать
- 1: Использовать левый двигатель
- 2: Использовать основной двигатель
- 3: Использовать правый двигатель

Пространство наблюдений представляет собой вектор из 8 значений: координаты посадочного модуля в x и y, его линейные скорости в x и y, его угол, его угловая скорость и два логических значения, которые показывают, соприкасается ли каждая опора с землей или нет.

После каждого шага в эпизоде начисляется вознаграждение. Общая сумма за эпизод — это результат всех действий в нём.

Награда за действие:

- растёт или уменьшается в зависимости от того, приближается или отдаляется спускаемый аппарат от места посадки.
- увеличивается или уменьшается в зависимости от того, замедляется или ускоряется движение аппарата.
- уменьшается, если посадочный модуль наклоняется (не горизонтально). увеличивается на 10 баллов за каждый контакт с землёй.
- уменьшается на 0,03 балла за каждый кадр, когда включается боковой двигатель.
- уменьшается на 0,3 очка за каждый кадр, когда запускается основной двигатель.

За успешную посадку или аварию в эпизоде участники получают дополнительную награду в размере -100 или +100 очков соответственно.

Эпизод считается успешным, если он набирает не менее 200 очков.

1.2. Среда MountainCar

Mountain Car MDP — это детерминированный MDP, в котором автомобиль случайным образом располагается на дне впадины, напоминающей по форме синусоиду. В этой игре доступны только два действия — ускорение в любом направлении. Задача игрока — разработать стратегию, которая позволит автомобилю достичь вершины холма. В рамках задачи Mountain car есть две версии: с дискретными и непрерывными действиями. В этом случае рассматривается версия с дискретными действиями. Визуальное представление среды представлено на рис. 2.

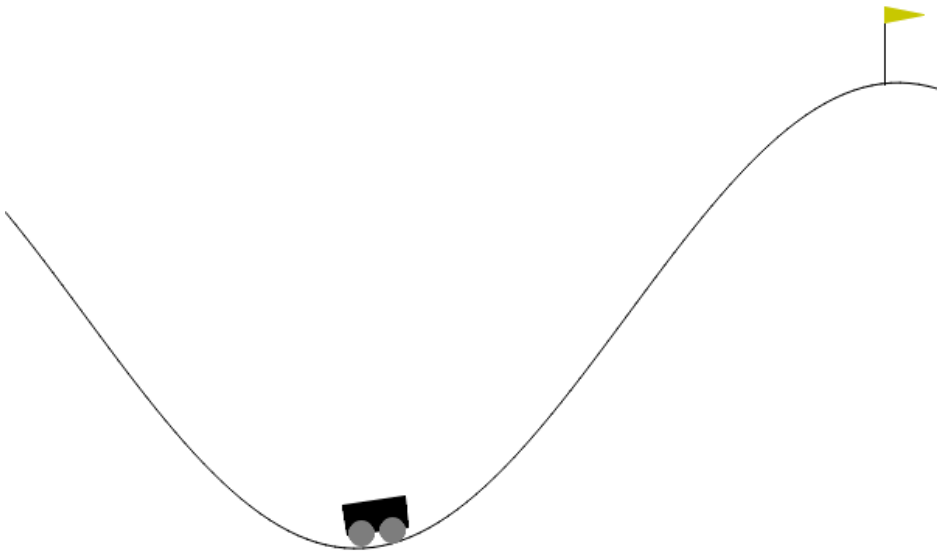

Рисунок 2 - Визуальное представление среды MountainCar

Пространство наблюдений представлено в табл. 1.

Таблица 1 - Структура пространства наблюдений

Num	Observation	Min	Max	Unit
0	position of the car along the x-axis	-1.2	0.6	position (m)
1	velocity of the car	-0.07	0.07	velocity (v)

Пространство действий из 3 детерминированных действий:

- 0: Ускориться влево
- 1: Не ускоряться
- 2: Ускориться вправо

Целью модели является достижение флага, расположенного на вершине правого холма.

2. АРХИТЕКТУРЫ DQN

2.1. Dueling Networks

В классическом DQN нейронная сеть напрямую предсказывает Q-значения для каждого действия.

В Dueling DQN архитектура разделяется на две ветви:

- Ветвь ценности состояния (Value stream) — оценивает, насколько "хорошо" находиться в данном состоянии V(s).
- Ветвь преимущества (Advantage stream) — оценивает, насколько лучше выбрать конкретное действие A(s,a) по сравнению с другими.

Итоговое Q-значение вычисляется как:

$$Q(s,a) = V(s) + A(s,a) - \frac{1}{|A|}\sum_{a'} A(s,a'),$$

где $V(s)$ – ценность состояния, $A(s,a)$ – преимущество действия a, $\frac{1}{|A|}\sum_{a'} A(s,a')$ – среднее по всем действиям. Нормализация делает обучение более устойчивым.

2.2. Multi-step Learning

Multi-step learning в DQN (Deep Q-Network) – это модификация классического алгоритма DQN, которая использует не одношаговые награды, а сумму наград за несколько шагов для более точной оценки Q-значений.

DQN вычисление таргетного значения Q:

$$Q_{target} = r_t + \gamma \max_{a'} Q_{\theta^-}(s_{t+1}, a'),$$

где θ^- – параметры целевой сети (target network), r_t – награда, s_{t+1} – следующее состояние, a' – это действие, которое максимизирует Q-значение в состоянии s_{t+1}, γ – коэффициент дисконтирования.

Multi-step Learning DQN вычисление таргетного значения Q:

$$Q_{target} = \sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^n \max_{a'} Q_{\theta^-}(s_{t+n}, a'),$$

где θ^{-} – параметры целевой сети (target network), r_{t+k} – награда, s_{t+1} – следующее состояние, a' – это действие, которое максимизирует Q-значение в состоянии s_{t+1}, γ – коэффициент дисконтирования, n – на каком количестве шагов вычисляется Q-значение.

2.3. Distributional RL

Distributional RL представляет собой подход, который расширяет подход DQN, путем оценки не только ожидаемых Q-значений, но и распределения возможных вознаграждений. Вместо предсказания одного значение *Q(s, a)*, DRL моделирует полное распределение возможных итоговых вознаграждений для каждого действия в состоянии.

В DRL используется функция ценности, которая описывает распределение вознаграждений, что позволяет агенту лучше понимать риски и неопределенности, связанные с различными действиями. Это достигается путем представления *Q*-значений в виде набора вероятностей, что позволяет учитывать различные сценарии и их вероятности.

Вместо вычисления Q-значения как ожидаемого вознаграждения, DRL вычисляет распределение *Z(s, a)*, где *Z* представляет собой распределение возможных итоговых вознаграждений для действия *a* в состоянии *s*. Обновление этого распределения происходит с использованием метода, аналогичного уравнению Беллмана, но с учетом распределений:

$$Z(s, a) = r + \gamma Z(s', a'),$$

где *r* — полученная награда, γ — коэффициент дисконтирования, а *s′* — следующее состояние. Это позволяет агенту учитывать не только среднее значение вознаграждений, но и их вариацию, что в свою очередь улучшает качество обучения и делает агента более устойчивым к неопределенности в среде.

3. СРАВНЕНИЕ АРХИТЕКТУР

3.1. Исследование в среде LunarLander-v3

Каждая ранее описанная модель обучалась в среде LunarLander-v3. Для каждой модели были построены графики наград и потери. Результат обучения представлен на рис. 3-5.

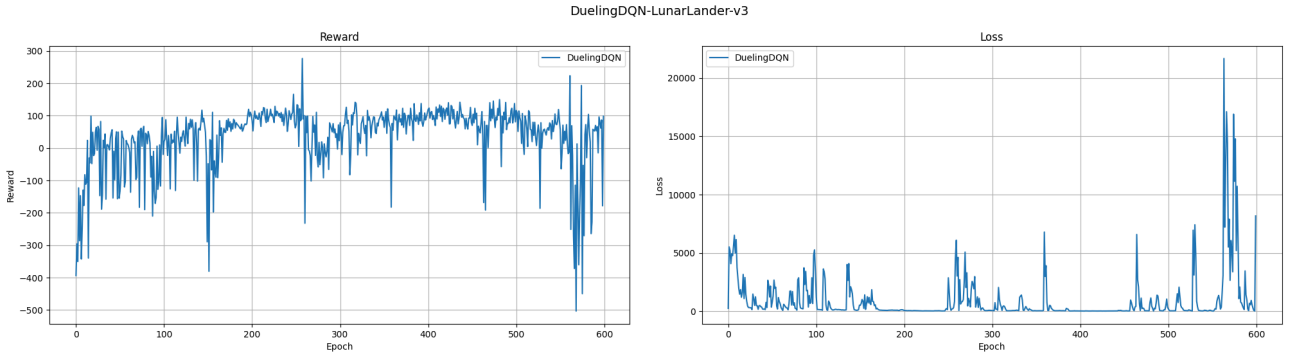

Рисунок 3 - Награда и потеря для Dueling networks в среде LunarLander-v3

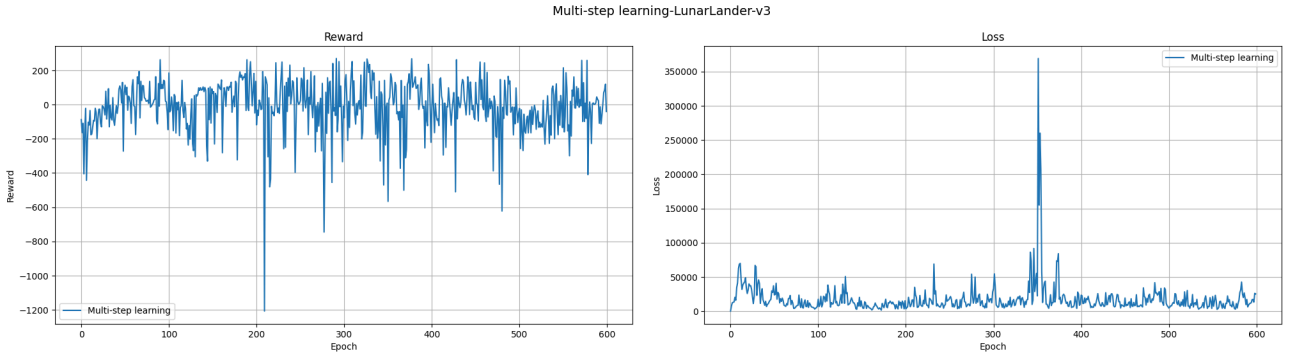

Рисунок 4 - Награда и потеря для Multi-step Learning в среде LunarLander-v3

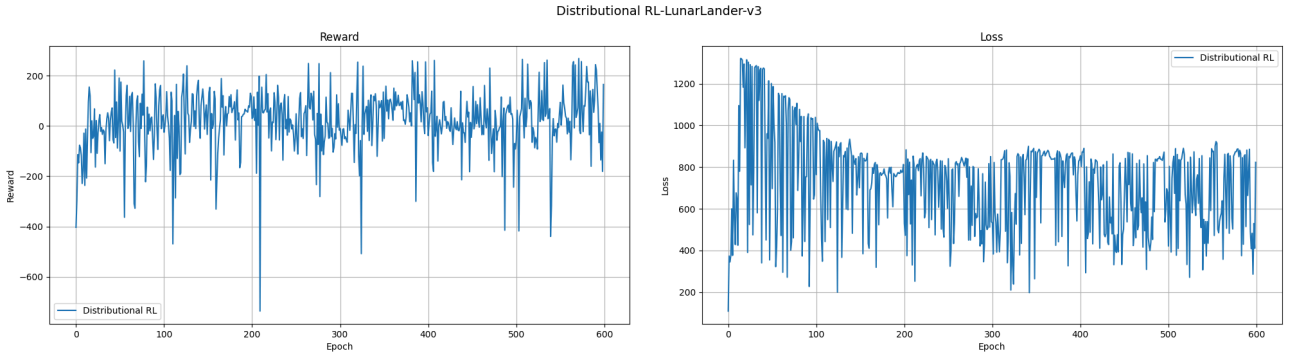

Рисунок 5 - Награда и потеря для Distributional RL в среде LunarLander-v3

Был построен общий график награды и потери для всех трёх архитектур. Результат представлен на рис. 6.

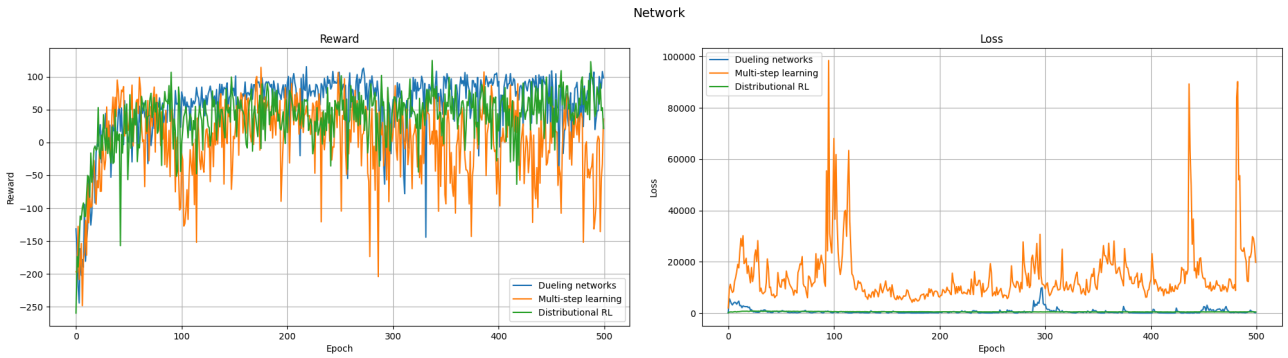

Рисунок 6 - График награды и потери для всех 3 архитектур в среде LunarLander-v3

Исходя из полученных результатов, можно сделать вывод, что Multi-step learning обучается хуже, поскольку на графике Reward в среднем она находится ниже остальных моделей, а также ее Loss значительно больше остальных. Сложно сказать какая из моделей: Dueling networks и Distributional RL лучше, потому что награда в среднем лучше у Dueling networks, зато функция потерь в среднем меньше и более стабильная у Distributional RL.

3.2. Исследование в среде MountainCar

Было проведено исследование архитектур в среде MountainCar-v0. Для каждой модели были построены графики наград и потери. Результат обучения представлен на рис. 7-9.

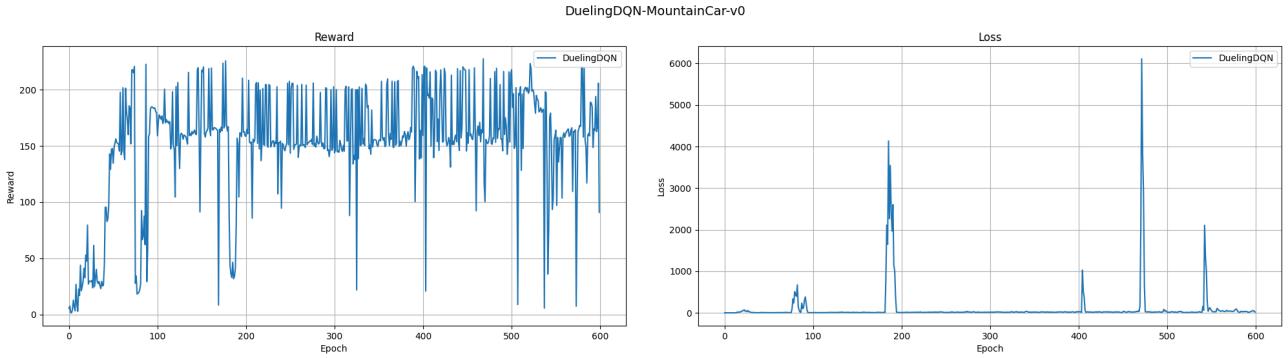

Рисунок 7 - Награда и потеря для Dueling networks в среде MountainCar-v0

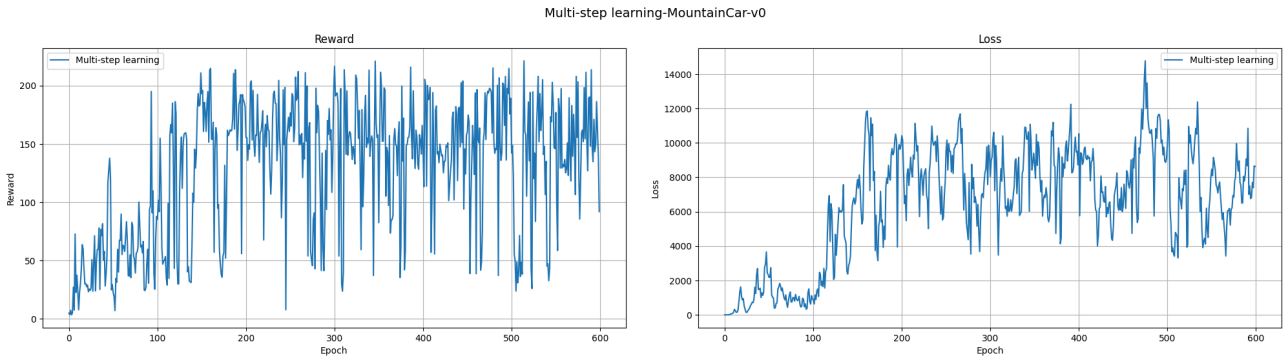

Рисунок 8 - Награда и потеря для Multi-step Learning в среде MountainCar-v0

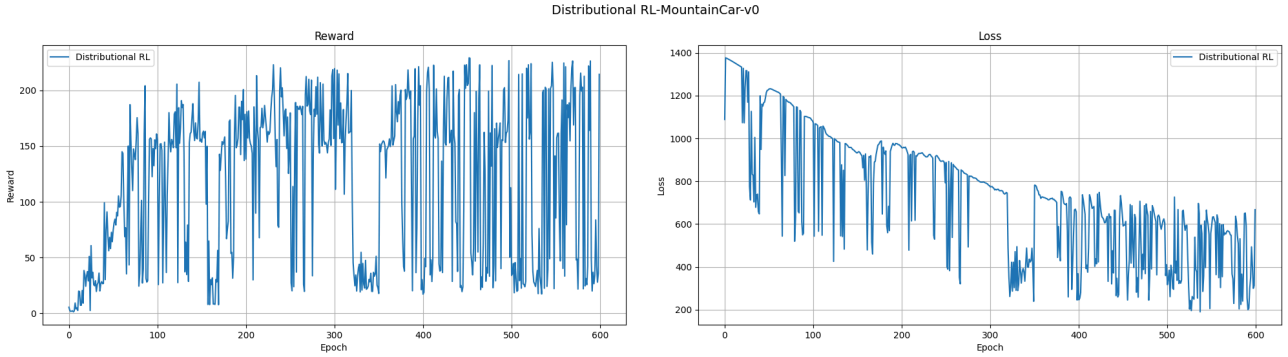

Рисунок 9 - Награда и потеря для Distributional RL в среде MountainCar-v0

Был построен общий график награды и потери для всех трёх архитектур. Результат представлен на рис. 10.

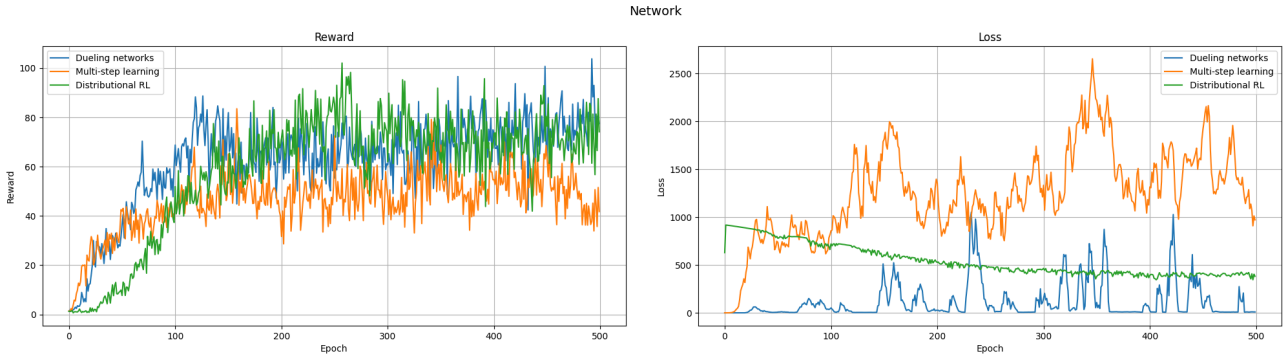

Рисунок 10 - Награда и потеря для 3 архитектур в среде MountainCar-v0

Исходя из полученных результатов, можно сделать вывод, что Multi-step learning обучается хуже, поскольку на графике Reward в среднем она находится ниже остальных моделей, а также ее Loss значительно больше остальных. У Distributional RL примерно до 350 эпохи средняя награда была выше остальных, с 350 по 420 награда приблизительна стала сравнима с Dueling networks, а после 420 эпохи средняя награда Dueling networks стала больше остальных моделей.

Также по графикам Loss видно что функция потерь у Dueling networks в среднем меньше, чем у других моделей, хоть и бывают скачки. У Distributional RL loss стабильно убывает.

ЗАКЛЮЧЕНИЕ

В ходе данного исследования были изучены и рассмотрены вариации алгоритма DQN: Dueling networks, Multi-step learning и Distributional RL.

Dueling networks и Distributional RL в окружениях LunarLander-v3 и MountainCar-v0 работают одинаково хорошо и выдают примерно одинаковую награду, агент обученный при помощи Multi-step learning также обучается, но хуже остальных моделей.

ПРИЛОЖЕНИЕ А

НАЗВАНИЕ ПРИЛОЖЕНИЯ

Ссылка на git-репозиторий:

https://github.com/AntonSeredenkov/DQN-2-research