Оглавление

Введение

Обучение с подкреплением – это один из способов машинного обучения. Отличительной особенностью этого метода является то, что для обучения не требуются наборы тренировочных данных. Обучение происходит в среде, которую можно описать марковским процессом принятия решений. Марковские процессы принятия решений представляют собой инструмент для постановки задачи обучения, где достижение цели осуществляется через взаимодействие и последовательное принятие решений. В процессе обучения агент (испытуемая система) взаимодействует со средой и получает обратную связь на свои действия, которую использует для своего обучения. В частности, агент получает данные о состоянии среды, на их основе совершает в среде действие, после чего среда снова передает агенту обновленные данные о состоянии среды, а также оценку оптимальности действия агента. Графически этот процесс можно представить следующим образом (Рисунок 1 - Марковский процесс принятия решений).

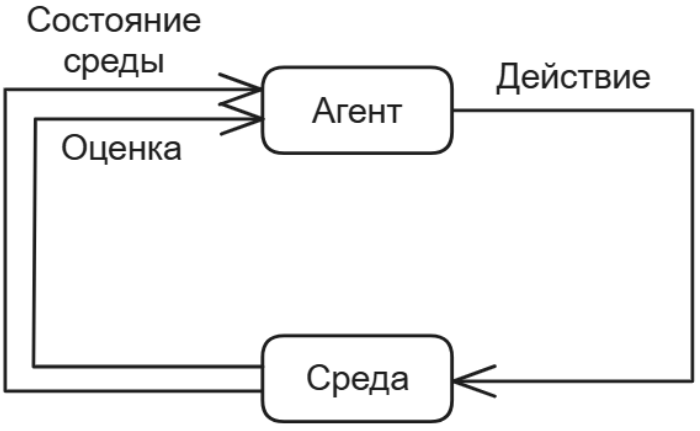

Рисунок 1 - Марковский процесс принятия решений

Целью учебной данной практики является реализация метода проксимальной оптимизации политики для сред как с дискретным, так и с непрерывным пространством действий. А также обучение агента с помощью данного метода.

1. Оптимизация политики

1.1. Общие сведения

Методы проксимальной оптимизации политики (ПОП) являются частью семейства методов градиента политики. В то время как стандартные методы градиента политики на одной выборке данных производят один шаг градиентного спуска, методы ПОП позволяют производить несколько эпох обучения на одной выборке данных. На эпохе обучения из главной выборки случайным образом формируется подвыборка, на которой и происходит обучение.

Также одним из преимуществ ПОП является то, что метод придерживается пессимистичного подхода при обновлении политики. Это помогает избежать резких скачков политики, которые могут “по инерции” увести ее от оптимальных значений. Таким образом обновление политики происходит небольшими шагами, что может сказаться на скорости обучения, но значительно повышают стабильность тренировки.

Еще одно преимущество методов ПОП заключается в том, что, в отличие от других предшествующих методов, они хорошо себя показывают как в средах с дискретным пространством действий, так и в средах с непрерывным пространством действий.

1.2. Методы градиента политики

Методы градиента политики вычисляют функцию оценки градиента политики и оптимизируют ее с помощью алгоритмов градиентного спуска. В общем виде функция оценки градиента политики выглядит следующим образом:

$$\bar{g} = \bar{E}_t\left[\nabla_\theta \log \pi_\theta\left(a_t \mid s_t\right)\bar{A}_t\right] \tag{1}$$

Где π_θ - стохастическая политики, $\bar{A}_t$ - оценка функции преимущества на шаге t, $\bar{E}_t$ - ожидаемое среднее из конечной выборки. На практике составляется заместительная функция, градиент которой является функцией оценки градиента $\bar{g}$. Продифференцировав данную функцию, получим функцию потерь, на которой проводится оптимизация:

$$L^{PG}(\theta) = \bar{E}_t\left[\log \pi_\theta\left(a_t \mid s_t\right)\bar{A}_t\right] \tag{2}$$

Но использование функции L^{PG} для оптимизации не эффективно, так как используется одна траектория действий. Так же, как говорилось ранее, метод может разойтись из-за слишком большого скачка в изменении оптимизации

1.3. Проксимальная оптимизация политики

Пусть отношение политик $r_t(\theta) = \dfrac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_c}\left(a_t \mid s_t\right)}$, где π_{θ_c} - предыдущая политика. Тогда в качестве замещающей функции возьмем:

$$L^{CLI}(\theta) = \max_\theta \bar{E}_t\left[\frac{\pi_\theta\left(a_t \mid s_t\right)}{\pi_{\theta_c}\left(a_t \mid s_t\right)}\bar{A}_t\right] = \max_\theta \bar{E}_t\left[r_t(\theta)\bar{A}_t\right] \tag{3}$$

В таком виде, без ограничений, изменение политики будет деструктивно большим. Для этой цели модифицируем замещающую функцию следующим образом:

$$L^{CLIP}(\theta) = \max_\theta \bar{E}_t\left[\min\left(r_t(\theta)\bar{A}_t, \text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\right)\right] \tag{4}$$

Где $\text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon)$ ограничивает $r_t(\theta)$ значениями $1-\varepsilon$ и $1+\varepsilon$ снизу и сверху соответственно, ε является гиперпараметром. При такой замещающей функции оптимизация будет иметь консервативный характер, так как отношение старой и новой политик не должно превышать определенного значения. Таким образом, $L^{CLI}(\theta) = L^{CLIP}(\theta)$ при значениях θ близких к θ_c и

ограничения никак не влияют на функцию. Но при большой разнице между θ и θ_c ограничения вступают в силу. Если допустить, что любые большие изменения политики деструктивно влияют на процесс оптимизации, то можно сказать, что при оптимизации на функции $L^{CLIP}(\theta)$ положительные изменения пропускаются, а негативные отбрасываются.

Также стоит отметить, что знак функции оценки преимущества $\overline{A}_t$ не влияет на правильность вычислений:

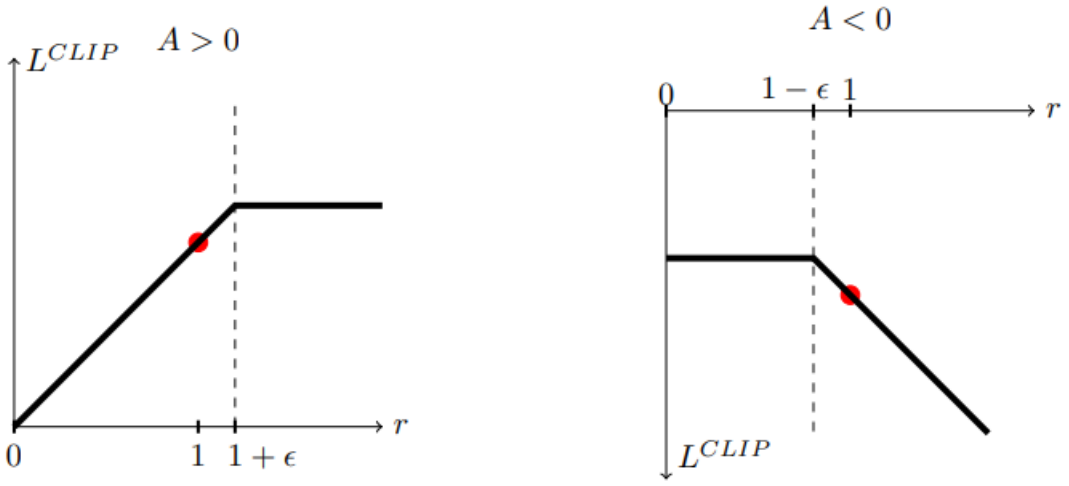

Рисунок 2 - значение замещающей функции при A>0 и A<0

2. Реализация метода проксимальной оптимизации политики

2.1. Составление функции потерь

В большинстве реализаций функция состояния-значения $V(s)$, использующаяся в вычислении оценки функции преимущества $\bar{A}_t$, является обучаемой. Например, в виде ограниченной обобщенной оценки преимущества:

$$\begin{aligned} \bar{A}_t &= \delta_t + (\gamma\lambda)\delta_{t+1} + \ldots + (\gamma\lambda)^{T-t+1}\delta_{T-1} \\ \delta_t &= r_t + \gamma V(s_{t+1}) - V(s_t) \end{aligned} \tag{5}$$

Где t лежит в промежутке $[0;T]$ на траектории действий длины T, а γ и λ - гиперпараметры.

Удобно использовать архитектуру нейросети, где политика и функция состояния-значения имеют общие параметры и не разделены. Тогда в качестве функции потерь можно использовать следующую функцию:

$$L^{CLIP+VS+S}(\theta) = \max_{\theta} \bar{E}_t \left[L^{CLIP}(\theta) - c_1 L^{VS}(\theta) + c_2 S(\pi_\theta) \right] \tag{6}$$

Где $L^{CLIP}(\theta)$ - функция потерь при оптимизации политики, $L^{VS}(\theta)$ - квадратичная погрешность $\left[V_\theta(s_t) - V_t^{цель} \right]^2$, S - бонус энтропии для поощрения исследования и предотвращения преждевременной сходимости. Коэффициент c_1 обычно берется равным 0,5, а коэффициент c_2 - гиперпараметр. Бонус энтропии не является обязательной частью функции потерь, он лишь вспомогательное средство и не обязателен в простых задачах. Бонус энтропии можно вычислить следующим образом:

$$S(\pi_\theta) = -\sum \pi_\theta \log(\pi_\theta) \tag{7}$$

Такая реализация позволяет использовать разные оптимизаторы и темпы обучения для политики и функции состояния-значения. В качестве оптимизатора обычно предлагается брать Adam.

2.2. Среды с дискретным пространством действий

В описанном выше варианте реализации метода агент представляет собой две нейросети, оптимизируемые по одной функции $L^{CLIP}(\theta)$. Первая нейросеть – функция состояния-значения, а вторая – политика. Реализация первой не зависит от пространства действий. Размерность входного слоя равна количеству S параметров состояния среды. Размерность выходного слоя всегда равна одному – предполагаемое значение $V_\theta(s_t)$.

Пусть среда с дискретным пространством действий позволяет совершать N возможных действий. Размерность входного слоя нейросети политики также равна S, а размерность выходного слоя равна N. Выходное значение сети – массив размерности N с числовым значением для каждого возможного действия в среде. После, для этого массива применяется функция softmax, которая преобразует значения массива в вероятности выбора соответствующего действия в диапазоне $[0;1]$. В результате получается массив, где каждому действию соответствует вероятность его выбора для текущего состояния среды. На основе этого массива формируется категориальное распределение (обобщенное распределение Бернулли), случайные значения из которого и будут являться действиями, принимаемые средой.

Внутри сети в качестве функции активации используется линейный выпрямитель:

$$f(x) = \begin{cases} x, x \geq 0 \\ 0, x < 0 \end{cases} \tag{8}$$

2.3. Среды с непрерывным пространством действий

Пусть среда с непрерывным пространством действий позволяет совершать N возможных действий, причем каждое действие – это значение в диапазоне $[-\infty;+\infty]$. Входная размерность нейросети политики всё так же равна S. Выходная размерность сети равна 2N, так как для каждого действия нужно по два значения. В результате получается массив размерности N, состоящий из пар значений для каждого действия. После, к каждому второму значению пары массива применяется функция $\ln(1+e^x)$, чтобы убедиться, что значение положительное. Далее для каждого действия, на основе соответствующих пар значений, составляется нормальное распределение. Для этого в качестве математического ожидания берется первое значение пары, а в качестве дисперсии – второе. Значение каждого действия выбивается случайным значением из соответствующего распределения.

В качестве функции активации используется гиперболический тангенс:

$$f(x)=\frac{e^x-e^{-x}}{e^x+e^{-x}} \tag{9}$$

3. Обучение агента

3.1. Выбор среды обучения

В качестве фреймворка для реализации метода был выбран PyTorch, а среды взяты из Gymnasium.

Изначально в качестве среды была выбрана среда с перевернутым маятником, классической проблемой динамики теории управления. Маятник установлен вертикально на тележке, и задача состоит в том, чтобы уравновесить шест, прикладывая к тележке силы в левом и правом направлениях. Однако обучить агента в этой среде не удалось.

Проблема заключалась в том, что награда за эпизод увеличивается на 1 на каждом шаге. То есть фактически агент не получает конкретной обратной связи на свои действия, а лишь косвенную, основанную на длине эпизода. Такой подход применим к методам с какой-либо памятью, но в случае ПОП успешность обучения абсолютно случайна. В качестве эксперимента агент был обучен на 500 эпизодах 200 раз с разными гиперпараметрами. В результате хоть сколько-нибудь успешными оказались только 4 модели.

Поэтому было решено в качестве среды обучения взять игру “Lunar lander”.

3.2. Описание среды обучения

Среда игры “Lunar lander” представляет собой классическую задачу оптимизации траектории полета ракеты. Агенту требуется успешно посадить лунный модуль на поверхность, потратив как можно меньше топлива. Попытка считается неудачной если модуль был перевернут или приземлился на поверхность на большой скорости. В качестве параметров состояния среда передает 8 значений: координаты по x и y, линейная скорость по x и y, угол наклона относительно горизонта, угловая скорость модуля и два булевых значения, показывающие коснулись ли ножки модуля поверхности.

Награда за эпизод тренировки высчитывается следующим образом:

- Увеличивается/уменьшается за приближение/отдаление от поверхности
- Увеличивается/уменьшается за медленную/быструю скорость передвижения
- Уменьшается пропорционально углу поворота относительно горизонта
- Увеличивается на 10 за каждую ножку, коснувшуюся поверхности
- Уменьшается на 0,03 за каждый шаг, когда работает боковой двигатель
- Уменьшается на 0,3 за каждый шаг, когда работает основной двигатель
- Увеличивается/уменьшается на 100 за удачную/неудачную посадку

В версии среды с дискретным пространством действий агент может выполнять 4 действия: бездействие, запустить левый боковой двигатель, запустить главный двигатель, запустить правый боковой двигатель. Согласно принципу максимума Понтрягина, оптимальным будет запустить двигатель на полную мощность или выключить его. Именно этот принцип используется в данной версии среды.

В версии среды с непрерывным пространством действий агент может выполнять 2 действия: контроль главного двигателя, контроль боковых двигателей. Главный двигатель контролируется значениями в диапазоне $[-1;1]$, где значения меньше 0 соответствуют отключенному двигателю. Боковые двигатели контролируются значениями в диапазоне $[-1;1]$, где значения меньше 0 активируют левый двигатель, а значения больше 0 активируют правый двигатель.

3.3. Процесс обучения агента в варианте среды с дискретным пространством действий

Для обучения с дискретным пространством действий были выбраны следующие гиперпараметры (Таблица 1 - Гиперпараметры для обучения в дискретном пространстве действий).

Таблица 1 - Гиперпараметры для обучения в дискретном пространстве действий

Гиперпараметр	**Значение**
Размерность внутренних слоев нейросети	512
Темп обучения	3e-4
Размер выборки	7
Эпохи	5
Гамма	0,97
Лямбда	0,93
Эпсилон	0,2
Коэффициент бонуса энтропии	1e-4

Обучение было проведено на 400 эпизодах. О результатах обучения можно судить по следующим метрикам (Рисунок 3 - Зависимость награды от эпизода (дискретное пространство действий) и Рисунок 4 - Зависимость количества шагов от эпизода (дискретное пространство действий)).

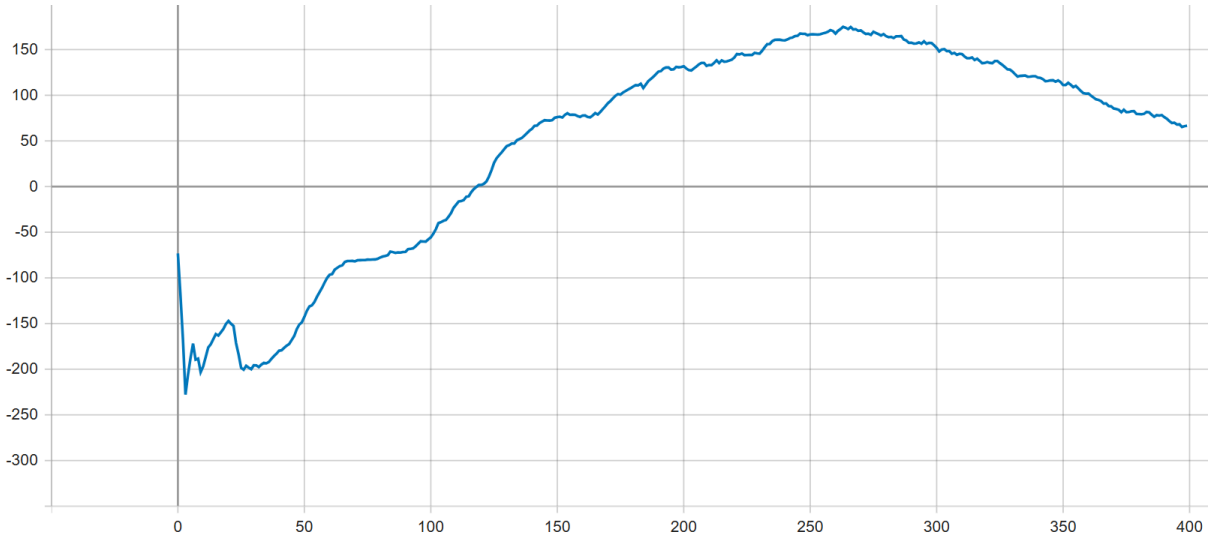

Рисунок 3 - Зависимость награды от эпизода (дискретное пространство действий)

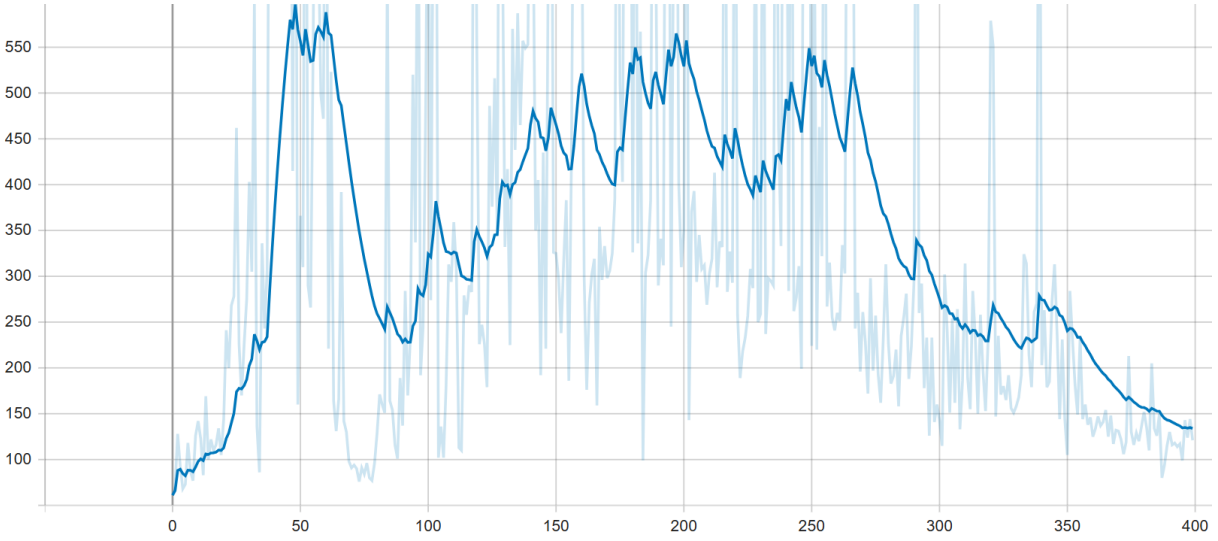

Рисунок 4 - Зависимость количества шагов от эпизода (дискретное пространство действий)

Как видно из графика зависимости средней награды, наибольшее значение было достигнуто после примерно 260 эпизодов. При дальнейшем обучении среднее значение награды уменьшается, что говорит о переобучении. Значение количества шагов также начало стремительно уменьшаться, что говорит о более частых экстренных завершениях эпизода (неудачная посадка).

График функции потерь в случае обучения с подкреплением, в отличие от обучения с учителем, не является надежным показателем производительности

агента. Набор данных для обучения не постоянный и меняется на каждой эпохе. Поэтому функция потерь может показывать производительность агента только в рамках одного эпизода, но не всего обучения.

3.4. Процесс обучения агента в варианте среды с непрерывным пространством действий

Для обучения с непрерывным пространством действий были выбраны следующие гиперпараметры (Таблица 2 - Гиперпараметры для обучения в непрерывном пространстве действий).

Таблица 2 - Гиперпараметры для обучения в непрерывном пространстве действий

Гиперпараметр	**Значение**
Размерность внутренних слоев нейросети	512
Темп обучения	3e-4
Размер выборки	30
Эпохи	5
Гамма	0,97
Лямбда	0,93
Эпсилон	0,2
Коэффициент бонуса энтропии	1e-4

Обучение было так же проведено на 400 эпизодах. Метрики обучения следующие (Рисунок 5 - Зависимость награды от эпизода (непрерывное пространство действий) и Рисунок 6 - Зависимость количества шагов от эпизода (непрерывное пространство действий))

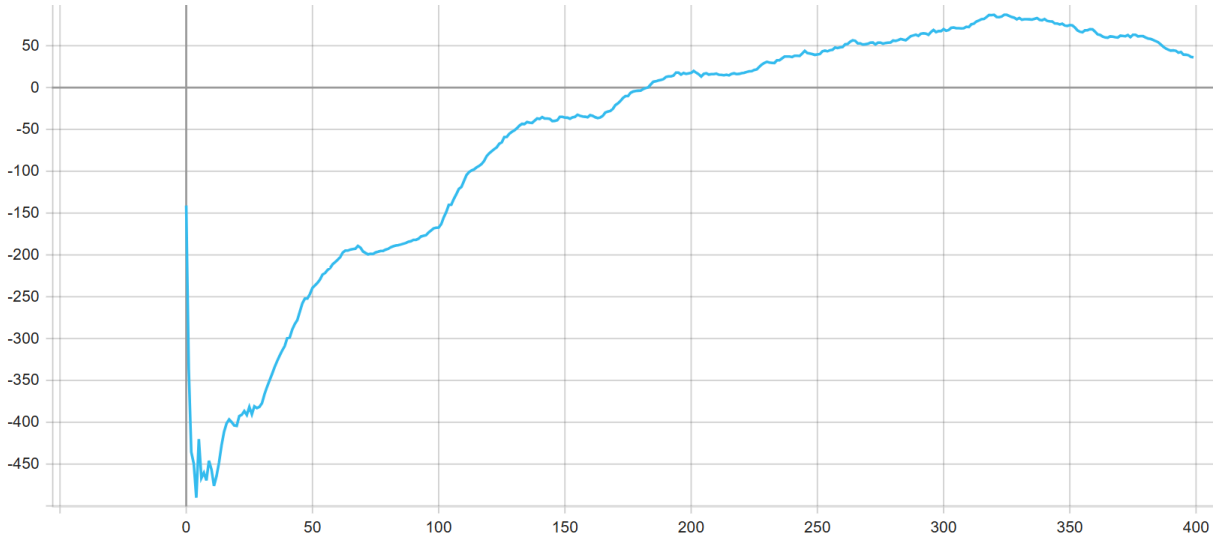

Рисунок 5 - Зависимость награды от эпизода (непрерывное пространство действий)

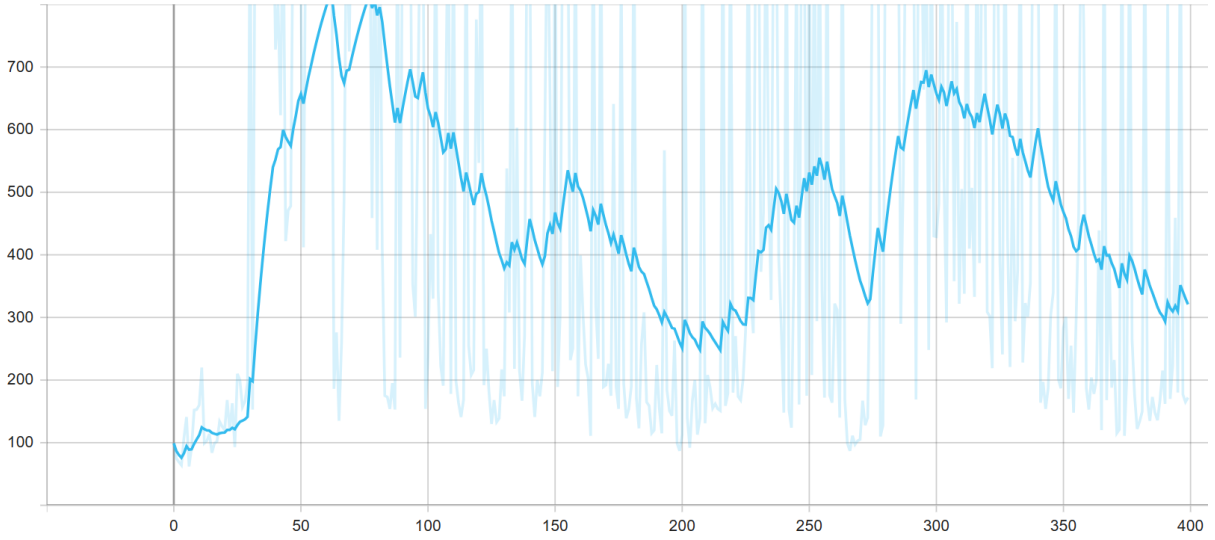

Рисунок 6 - Зависимость количества шагов от эпизода (непрерывное пространство действий)

В данном случае оптимальный результат был достигнут после примерно 350 эпизодов, что подтверждает и график зависимости количества шагов от эпизода.

В результате агент показал себя хуже в среде с непрерывным пространством действий. Но это говорит лишь о том, что конкретной среде лучше подходит пространство непрерывных действий.

Заключение

В результате выполнения практики был успешно реализован метод проксимальной оптимизации политики. Реализованный метод работает как в средах с дискретным пространством действий, так и в средах с непрерывным пространством действий. Также в выбранной среде с помощью реализованного метода был успешно обучен агент в обоих вариантах пространства действий.

Кроме того, выяснилось, что ПОП не подходит для обучения в средах, где награда за эпизод оценивает действия агента лишь косвенно.

Источники

1. Proximal Policy Optimization Algorithms [Electronic resource] / John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov – Mode of access: https://arxiv.org/pdf/1707.06347 - Title from screen
2. Asynchronous methods for deep reinforcement learning [Electronic resource] / V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu – Mode of access: https://arxiv.org/pdf/1707.06347 - Title from screen
3. Entropy Regularization [Электронный ресурс]. URL: https://paperswithcode.com/method/entropy-regularization (дата обращения 10.05.2024).
4. PPO Hyperparameters and Ranges [Электронный ресурс]. URL: https://medium.com/aureliantactics/ppo-hyperparameters-and-ranges-6fc2d29bccbe (дата обращения 25.05.2024)
5. Документация Gymnasium, среда “Lunar Lander” [Электронный ресурс]. URL: https://gymnasium.farama.org/environments/box2d/lunar_lander/ (дата обращения 05.04.2024)
6. Документация TorchRL [Электронный ресурс]. URL: https://pytorch.org/rl/stable/index.html - (дата обращения 10.04.2024)