Отчет по решению задачи Taxi с использованием алгоритма кросс-энтропии

1. Введение

Цель данного эксперимента — обучить агента решать задачу Taxi из среды OpenAI Gym с помощью алгоритма кросс-энтропии, исследовать влияние гиперпараметров на процесс обучения, а также протестировать различные варианты сглаживания (Laplace и Policy Smoothing), модификации. Мы сравним эффективность на основе средних вознаграждений и визуализируем процесс обучения на графиках.

2. Описание задачи Taxi

Taxi — это среда из библиотеки Gym, где агент (таксист) должен забрать пассажира в одном месте на карте и доставить его в другое. Агенту предоставляется ограниченное количество шагов для выполнения задачи. Важными компонентами среды являются:

- **Количество состояний**: 500 (агент находится в разных позициях и в разных состояниях с пассажиром).
- **Количество действий**: 6 (вверх, вниз, влево, вправо, подобрать пассажира, высадить пассажира).
- **Награды**: положительные награды за успешную доставку и штрафы за некорректные действия (например, попытку высадить пассажира в неправильном месте).

3. Метод кросс-энтропии (без сглаживаний и модификаций) (Пьянков_practice1_1.py)

3.1. Описание алгоритма

Алгоритм кросс-энтропии состоит из двух основных шагов:

- **Оценка политики**: Агент выполняет несколько эпизодов (траекторий), использует свою текущую модель, основанную на вероятностях выбора действий в разных состояниях.

- **Улучшение политики**: На основании "элитных" траекторий (траекторий с максимальными наградами) обновляется модель вероятностей действий для каждого состояния.

3.2. Гиперпараметры

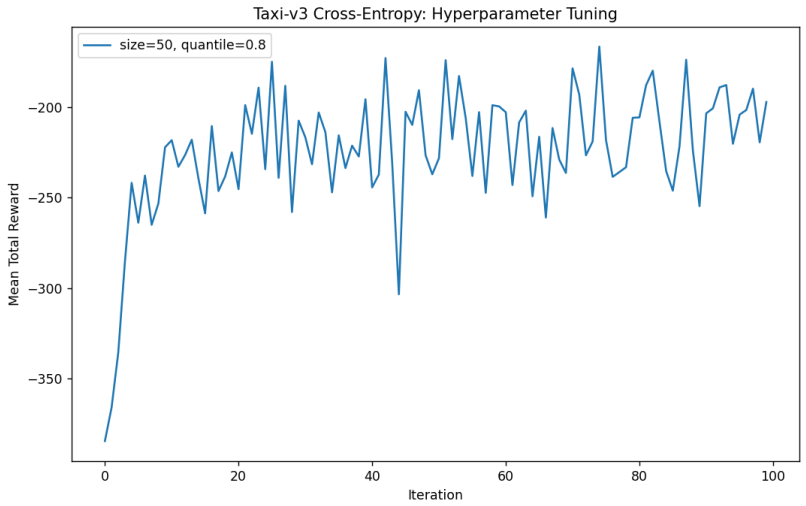
При обучении с использованием алгоритма кросс-энтропии существует несколько ключевых гиперпараметров, которые влияют на процесс обучения агента. Каждый из них имеет важное значение для того, как быстро и эффективно агент научится решать задачу.

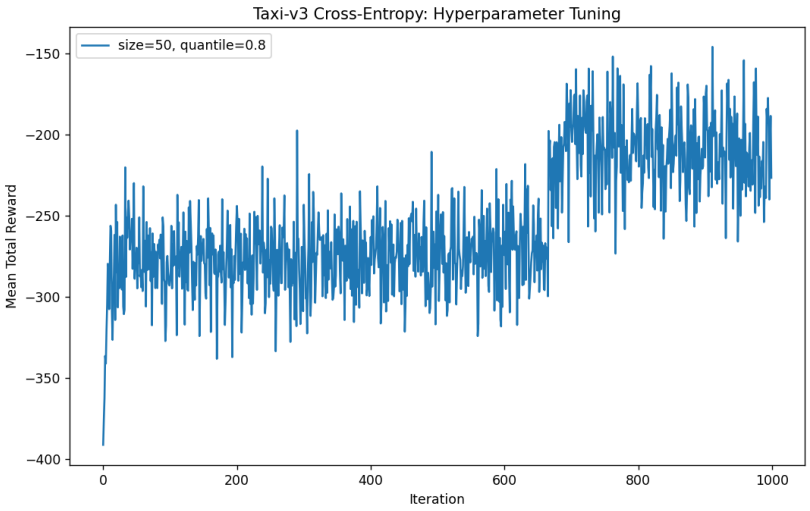
- *iteration_n* — Количество итераций обучения

  Описание: Это количество циклов обучения, в которых агент обновляет свою политику на основе собранных траекторий.

  Влияние: Большее количество итераций позволяет агенту более тщательно обновить свою политику и лучше адаптироваться к задаче. Однако слишком большое количество итераций может привести к тому, что агент будет переобучаться, особенно если в какой-то момент он достигнет почти оптимальной политики.

  iteration_n = 100 (оптим)

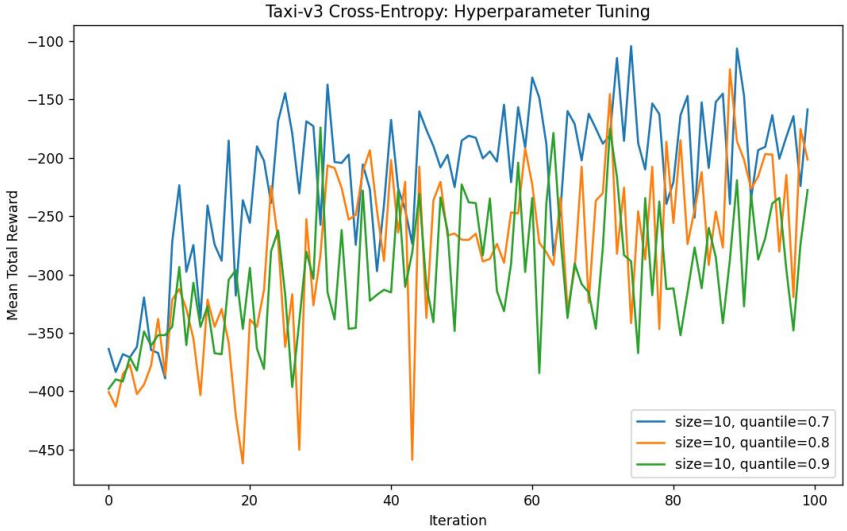


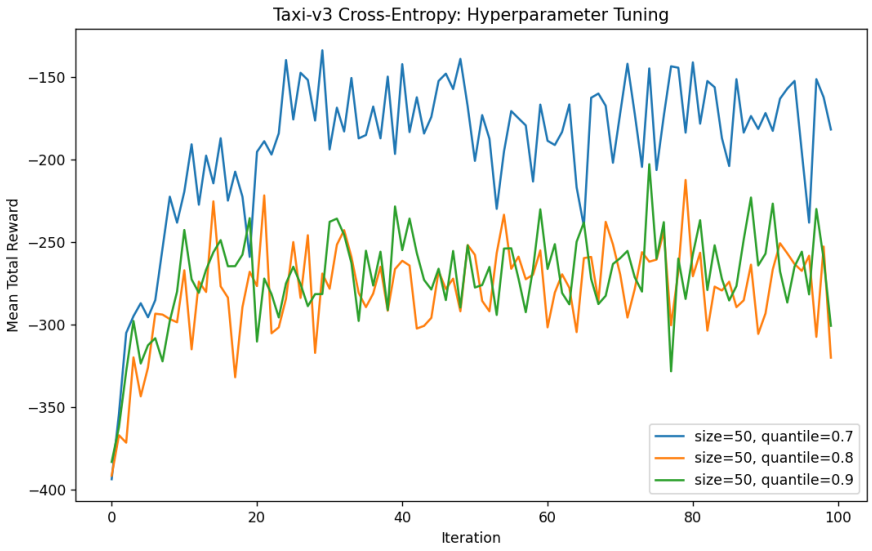


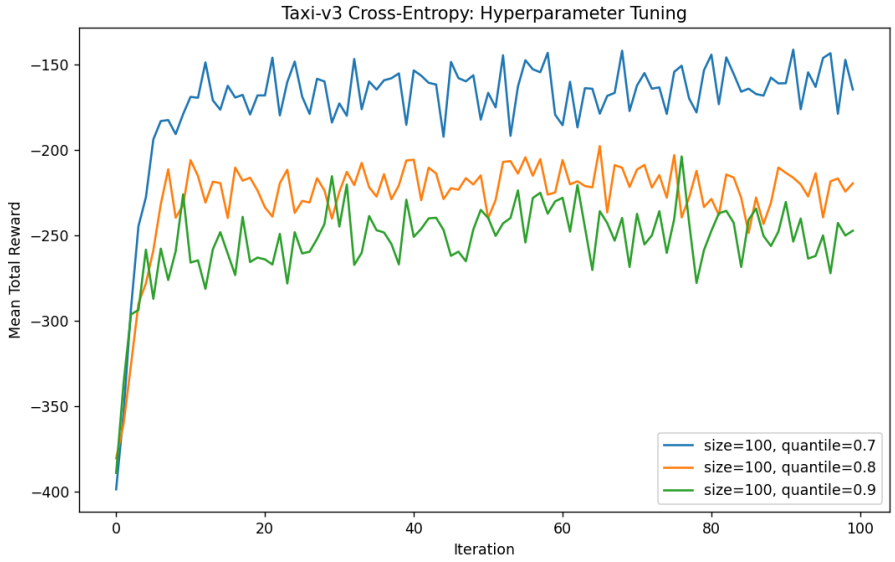
- *trajectory_n* — Количество траекторий за итерацию

  Описание: Это количество эпизодов (траекторий), которые агент проходит в каждой итерации, чтобы собрать данные для обновления своей политики.

  Влияние: Чем больше траекторий агент собирает, тем больше информации о среде и действиях он получает, что позволяет делать более точное обновление модели. Однако, большое количество траекторий замедляет процесс, так как требует больше вычислений и симуляций.





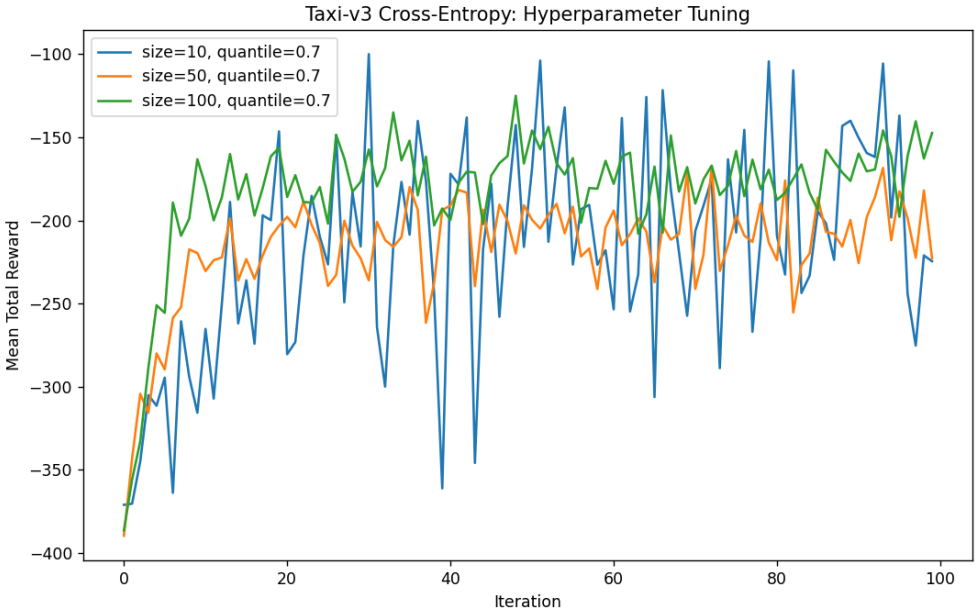


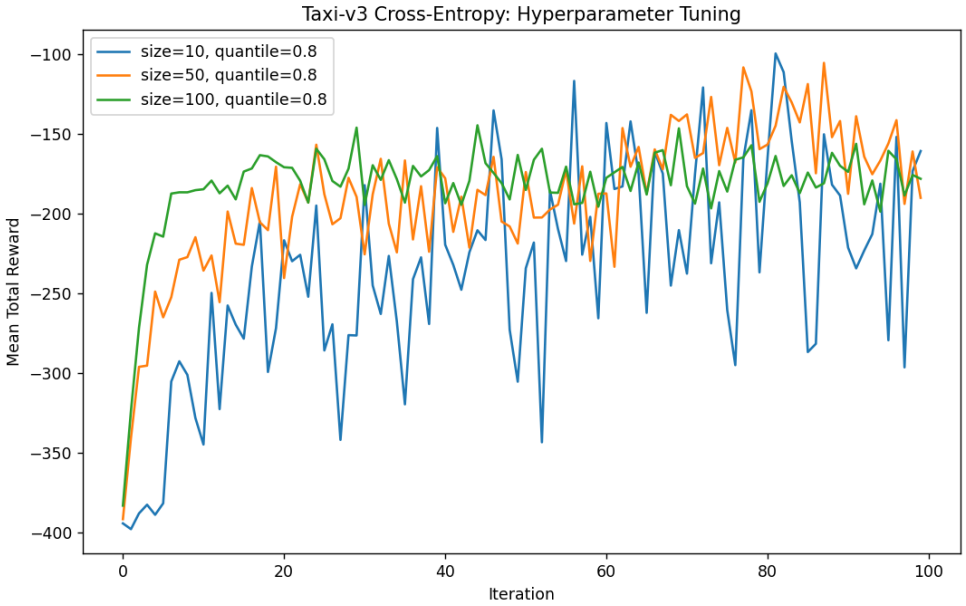
- *q_param* — Квантиль для отбора элитных траекторий

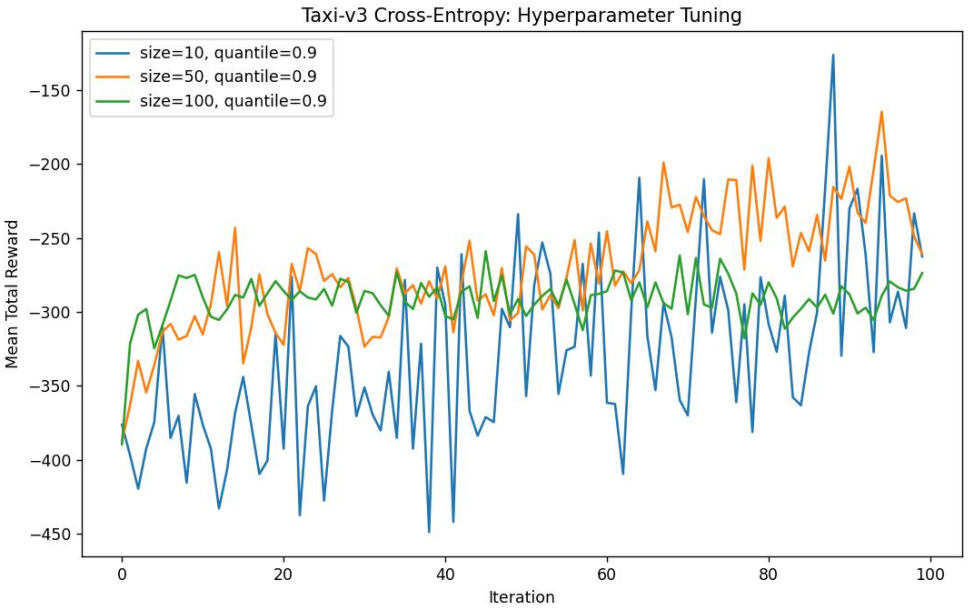
  Описание: Это процент (или квантиль) от общего числа траекторий, которые считаются "элитными" — траекториями с наивысшей суммарной наградой. Например, если q_param = 0.9, то в отбор попадут 10% лучших траекторий.

  Влияние: Чем выше значение квантиля, тем меньше траекторий будет выбрано для обновления модели, что сделает обновление модели более "жестким" (только лучшие действия будут использованы для обновления). Более низкие значения квантиля позволяют учитывать более широкий

диапазон траекторий, что может сделать обучение более "гибким", но увеличивает риск того, что агент будет использовать и не самые оптимальные действия.







- max_len — Максимальная длина траектории

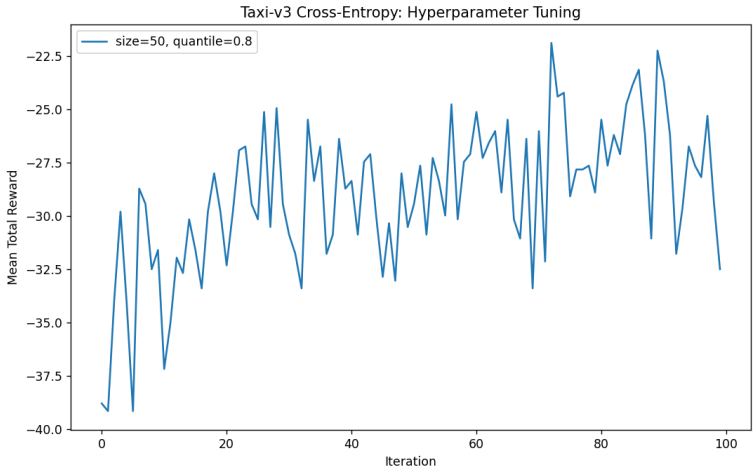
  Описание: Это максимальное количество шагов, которое агент может сделать в одном эпизоде перед его завершением.

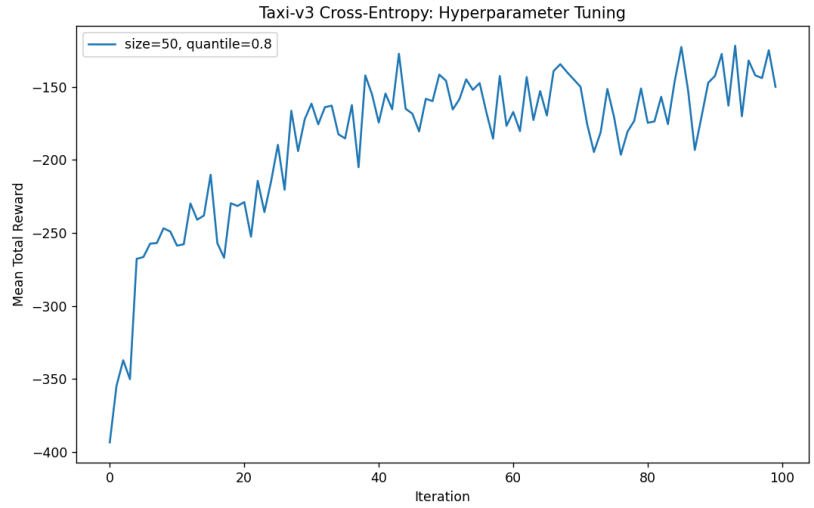
  Влияние: Если max_len слишком мал, агент может не успеть достичь цели или получить достаточно опыта за эпизод, что сделает процесс обучения менее эффективным. Если max_len слишком велик, то агент может бесцельно блуждать, и обучение может стать излишне долгим и неэффективным.

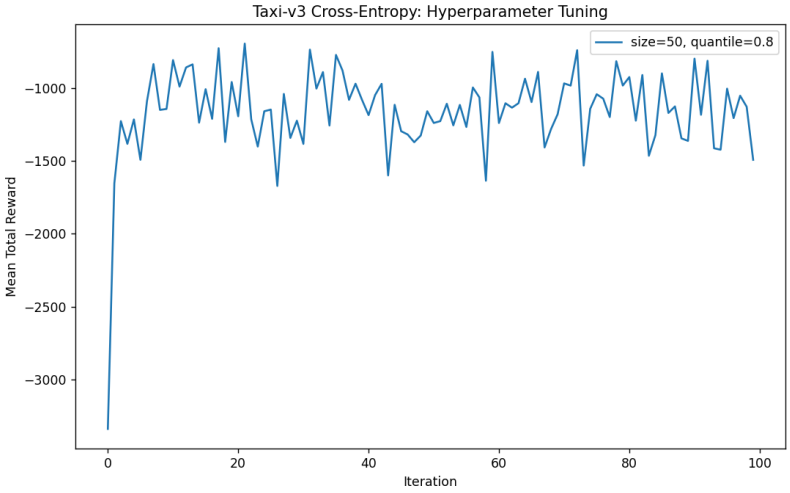
  (1 график – max_len = 10

  2 график – max_len = 100 (оптим)

  3 график – max_len = 1000)







3.3 Вывод

Каждый из гиперпараметров играет важную роль в обучении агента, и их оптимальные значения могут сильно зависеть от конкретной задачи и среды. Чтобы найти лучшие значения, необходимо проводить эксперименты, варьируя каждый из параметров и отслеживая, как изменяются результаты обучения на основе средней награды и скорости сходимости.

Из проведенного эксперимента видно, что увеличение размера выборки траекторий приводит к более стабильной и быстрой сходимости, особенно при значении 100 траекторий за итерацию. Однако слишком большой размер выборки может замедлять процесс обучения из-за

увеличенной вычислительной нагрузки. Квантиль 0.8 показал оптимальный баланс между скоростью и качеством решения задачи. При квантиле 0.9 наблюдается повышение стабильности, но с замедленной сходимостью.

4. Использование сглаживания (Пьянков_practice1_2.py)

В рамках данного эксперимента были изучены методы кросс-энтропии для обучения агента в среде **Taxi**. Основное внимание уделялось сравнительному анализу трех подходов:

1. **Без сглаживания** (No Smoothing)
2. **Laplace Smoothing**
3. **Policy Smoothing**

Целью эксперимента было выяснить, какой из методов наиболее эффективен в обучении агента, и проанализировать результаты.

4.1 Laplace Smoothing

В сглаживании Лапласа к каждому элементу вероятности добавляется небольшая константа, чтобы избежать нулевых вероятностей.

Гиперпараметры

- Константа сглаживания: 0.1
- Остальные гиперпараметры аналогичны варианту без сглаживания.

4.2 Policy Smoothing

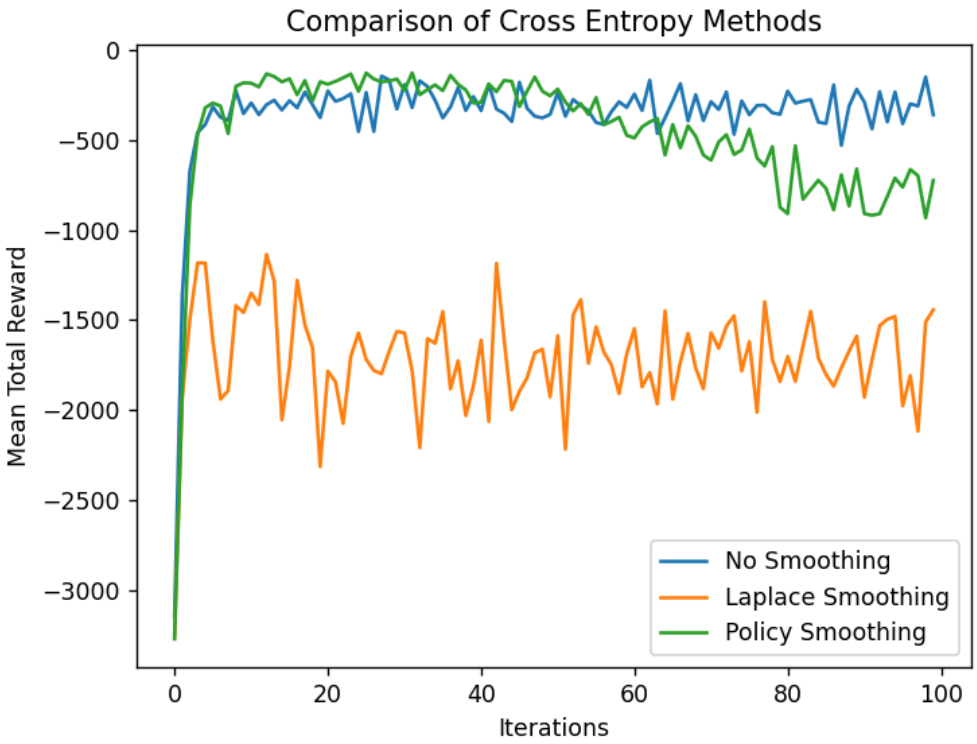
Сглаживание политики заключается в добавлении вероятностей к возможным действиям с учётом небольшого случайного компонента. Это позволяет агенту сохранять некоторое разнообразие действий.

Гиперпараметры

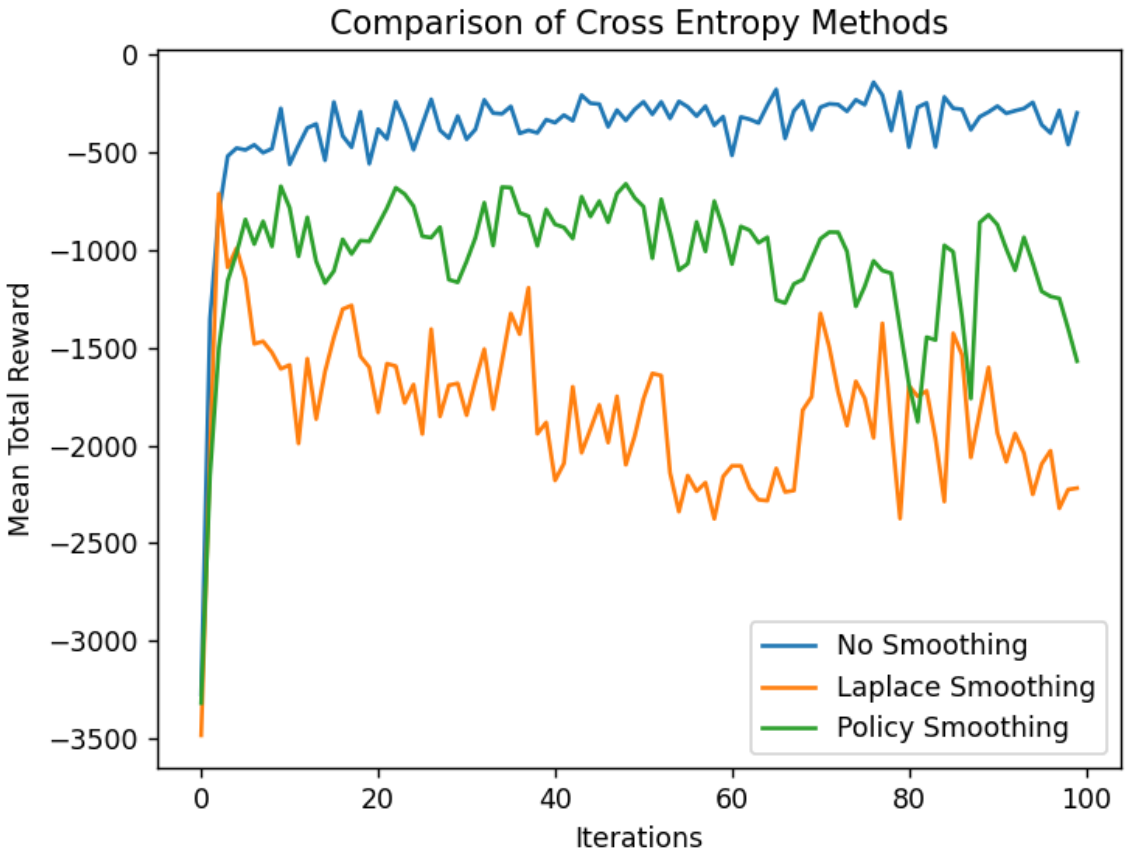
- Параметр случайного выбора: 0.1

- Остальные гиперпараметры остаются теми же.

4.3 Результаты (q_param = 0.8, iteration_n = trajectory_n = 100)



(Laplace) alpha = 1; (Policy) epsilon = 0.1



(Laplace) alpha = 0.1; (Policy) epsilon = 0.5

(При изменении других параметров: **q_param, iteration_n, trajectory_n,**

улучшений не наблюдается)

На графиках, которые иллюстрируют результаты обучения, видно, что метод без сглаживания демонстрирует наилучшие результаты, достигая наибольшего среднего вознаграждения. В то время как методы с применением сглаживания, особенно Лаплас, показывают худшие результаты и более низкие значения среднего вознаграждения.

4.4Вывод

Сглаживание по Лапласу:

Этот метод был рассчитан на смягчение влияния редких действий, однако, судя по полученным результатам, он привел к ухудшению эффективности. Это может быть связано с тем, что добавление

псевдосчетов в вероятность действия не всегда адекватно отражает реальную среду. Вместо улучшения, это сглаживание могло ввести дополнительный шум в оценки действий.

Policy Smoothing:

Данный метод также не продемонстрировал ожидаемого улучшения в обучении. Это может происходить из-за недостаточного учета случайности в политике, что могло снизить разнообразие действий агента и ограничить его возможности исследовать среду.

Без сглаживания:

Метод без сглаживания показал наилучшие результаты, что подтверждает предположение о том, что в данной среде кросс-энтропия, использующая точные вероятности, более эффективна. Удаление сглаживания позволяет модели лучше адаптироваться к конкретной структуре среды, что приводит к более оптимальным стратегиям.

5. Модификация кросс-энтропии с детерминированными политиками (Пьянков_practice1_3.py)

5.1. Введение

В данном отчете мы исследуем модификацию алгоритма кросс-энтропии, которая использует детерминированные политики вместо стохастических. Цель эксперимента — сравнить эффективность предложенного подхода с другими методами в среде Taxi.

5.2. Описание метода

Модифицированный алгоритм кросс-энтропии предполагает использование детерминированных политик, которые формируются на основе "элитных" траекторий. На каждой итерации выбираются элитные траектории (те, которые привели к наивысшим суммам вознаграждений), и на их основе обновляется политика агента. В отличие от традиционного подхода, агент всегда выбирает действие с максимальной вероятностью, тем самым следуя детерминированной политике.

Формально, модель обновляется следующим образом:

- Политика агента обновляется на основе частотных подсчетов действий, выполненных в элитных траекториях.

- Для каждого состояния агент выбирает действие с наибольшим количеством выборов.

5.3. Результаты и вывод

В отличие от стохастических политик, где агент может исследовать различные действия, детерминированная политика приводит к тому, что агент "застревает" в определенной траектории. Это не позволяет исследовать новые стратегии.
Процесс обучения оказался крайне длительным. Время, необходимое для достижения результатов, значительно увеличилось по сравнению с другими алгоритмами, использующими стохастические подходы.