ДЗ6

I. PPO: Advantage функция

Цель эксперимента: сравнить два способа вычисления Advantage-функции (A(s, a)) в задаче управления маятником (Pendulum-v1):

- **Старый метод** (на основе полного расчета возврата через returns).

- **Новый метод** (использующий представление $A(s,a)=r+\gamma V(s')-V(s)$).

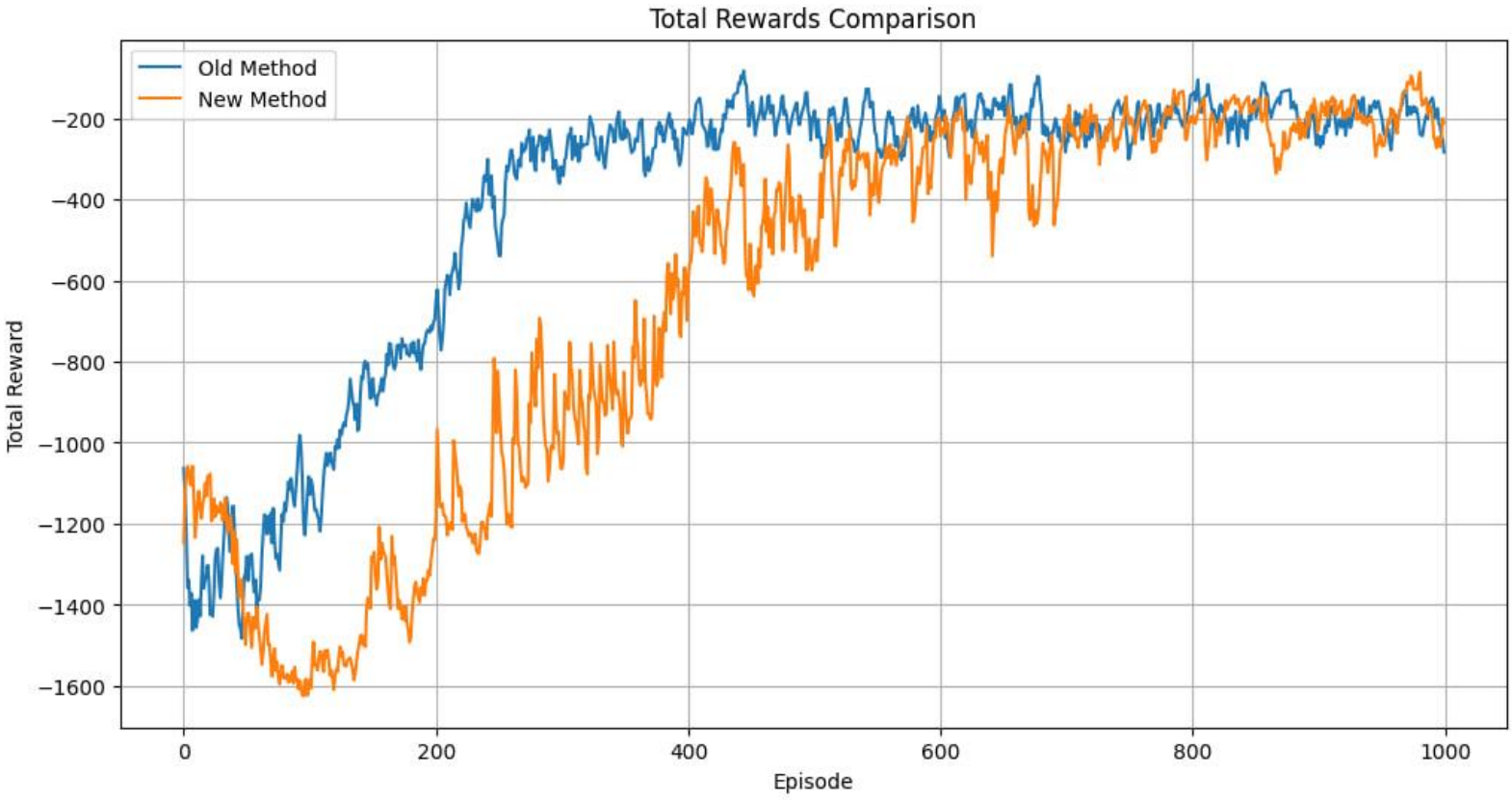
Advantage-функция A(s,a) показывает, насколько текущее действие **a**, выбранное в состоянии **s**, лучше среднего действия, доступного в данном состоянии. Она важна для корректного обучения, так как помогает уменьшить дисперсию оценок награды и направляет обновление политики в PPO на более выгодные действия.

Старый метод:

- Использует накопленные вознаграждения (returns).
- Обеспечивает более устойчивую оценку Advantage, так как учитывает полный путь траектории.
- Однако он требует больше вычислительных затрат, так как необходимо пройти всю траекторию.

Новый метод:

- Представляет Advantage как $A(s,a)=r+\gamma V(s')-V(s)$, где V(s) — оценка ценности состояния.
- Теоретически, он быстрее, так как требует только информацию о текущем и следующем состояниях.
- Однако, если V(s) плохо приближена, это может привести к более нестабильному обучению.

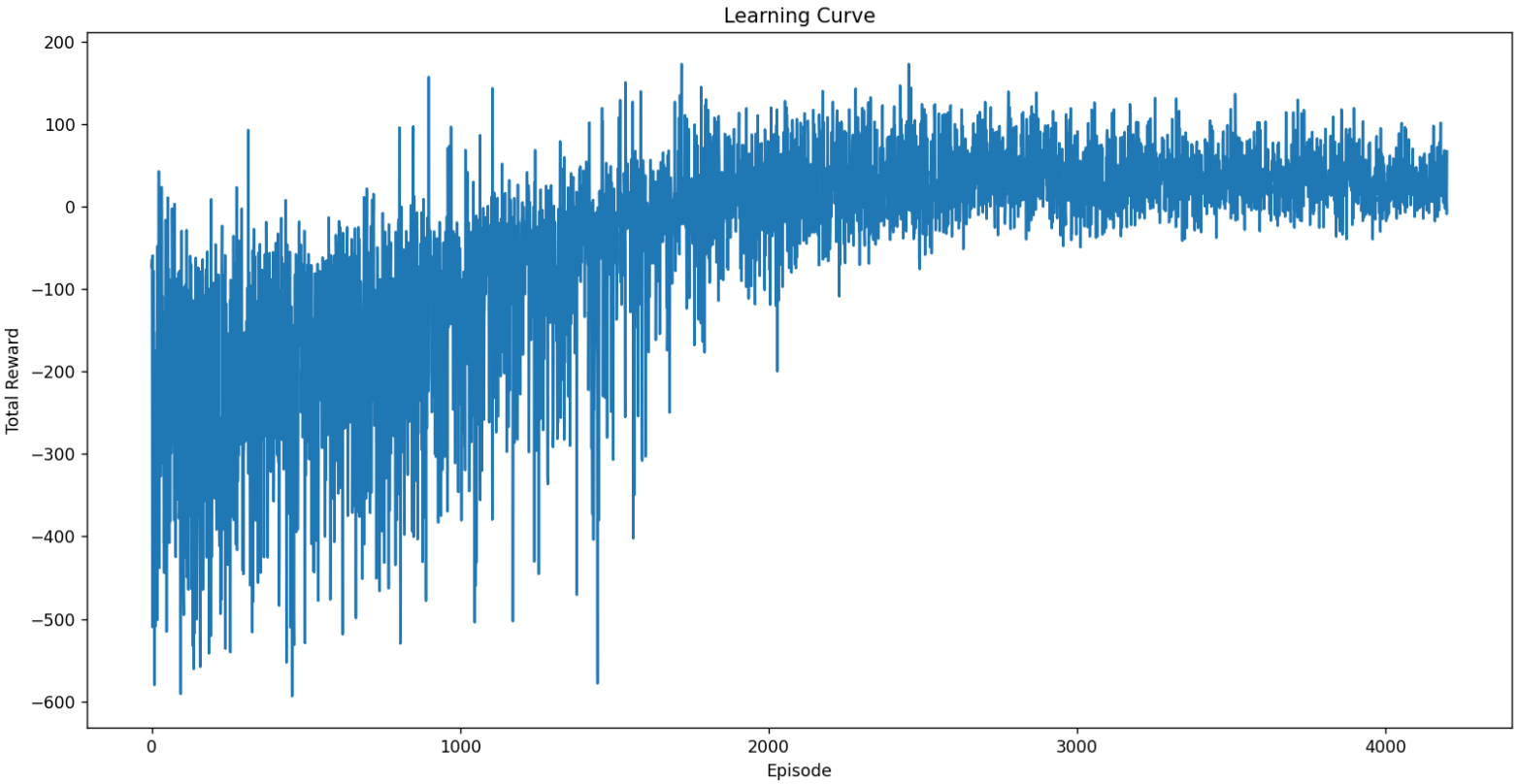


Более быстрая сходимость у старого метода из-за более точных Advantage, учитывающих всю траекторию.

Новый метод страдает от ошибки аппроксимации функции ценности V(s), что приводит к менее эффективному обучению.

II. PPO в среде с многомерным пространством действий.

В данном эксперименте мы адаптировали алгоритм Proximal Policy Optimization (PPO) для задачи с непрерывным пространством действий, используя среду **LunarLander-v2**.

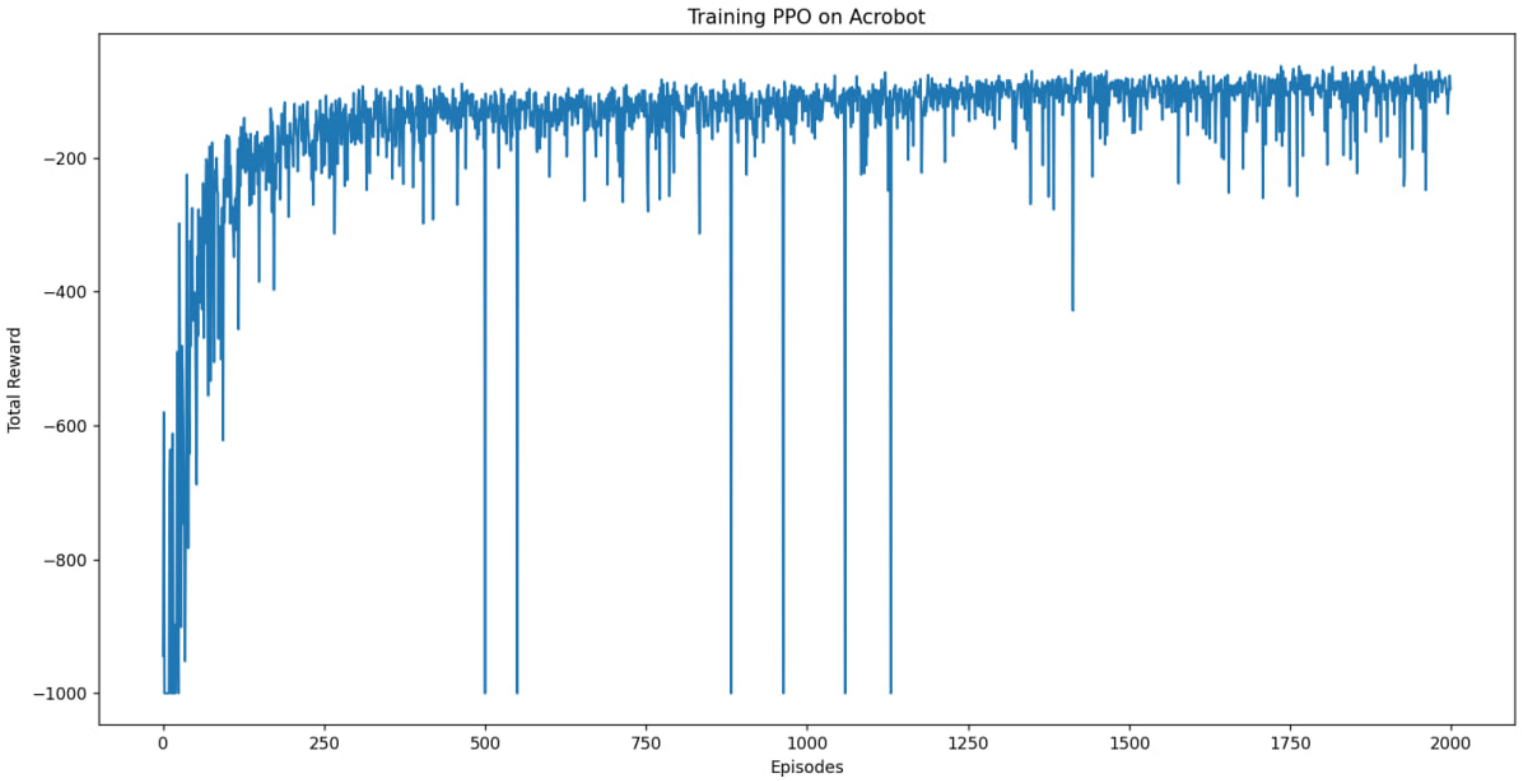


Алгоритм работает в средах с многомерным пространством действий.

III. PPO в среде с конечным пространством действий.

Данный эксперимент посвящен применению алгоритма Proximal Policy Optimization (PPO) для среды **Acrobot-v1** из библиотеки OpenAI Gymnasium. Цель эксперимента — проверить эффективность PPO в задаче с конечным пространством действий.

Политика реализована с использованием слоя Softmax для моделирования вероятностей выбора каждого действия (используется Categorical из torch.distributions).



В ходе обучения алгоритм смог достичь среднего значения награды на уровне около -100, что соответствует успешному выполнению задачи.

Наблюдался стабильный рост средней награды по мере увеличения количества эпизодов.