Reinforcement Learning для игры Snake

Введение

Игра Snake: Классическая аркадная игра, где змейка перемещается по сетке, собирает еду и растет в длину. Агент должен научиться:

- эффективно находить и собирать еду
- избегать столкновений со стенами
- избегать столкновений с собственным телом
- максимизировать длину змейки и время выживания

Актуальность: Задача управления агентом в дискретной среде с разреженными вознаграждениями (на примере змейки) служит тестовой площадкой для сравнения эффективности табличных и нейросетевых методов обучения с подкреплением. Полученные результаты имеют практическое значение для задач автономной навигации, в частности — для планирования траекторий складских роботов, где требуется избегать столкновений, оптимизировать путь и адаптироваться к динамически меняющейся обстановке.

Теоретическая часть

В основе используемых алгоритмов (DQN, SARSA, Q-learning) лежит оценка temporal difference error (**TD-error**):

$$\delta = r + \gamma * Q(s', a') - Q(s, a) = \text{TD-target} - Q(s, a),$$

где Q(s, a) - оценка, TD-target — целевое значение ценности пары состояние-действие.

SARSA: $r + \gamma * Q(s', \pi(s')) - Q(s, a)$
Q_learning: $r + \gamma * max_{a'} Q(s', a') - Q(s, a)$
DQN: $r + \gamma * max_{a'} Q(s', a', \theta_{frozen}) - Q(s, a, \theta),$

где параметры θ изменяются в направлении уменьшения TD-error через MSE loss и его аналоги.

Реализация среды

Создана среда через класс `SnakeEnv`, совместимая с интерфейсом `gymnasium.Env` (проверена на корректность с помощью `gymnasium.utils.env_checker`).

Состояние среды имеет 13-мерное векторное представление:
- нормализованные координаты головы змеи (2)
- нормализованные координаты еды (2)
- направление движения в виде one-hot кодировки (4)
- бинарные признаки столкновения в 4 направлениях (вверх/вправо/вниз/влево) (4)
- нормализованное манхэттенское расстояние до еды (1)

Нормализация сделана для устойчивого обучение DQN.

Действия — дискретные (`Discrete(4)`), соответствуют направлениям: вверх, вправо, вниз, влево.
Запрещён поворот на 180°: если выбрано противоположное текущему направление, действие игнорируется, и змея продолжает движение в прежнем направлении. Это сделано в ущерб сокращения размерности действий для обработки недопустимых действий.

В среде используется базовая функция награды без reward shaping:

$$R_t = \begin{cases} +10 & \text{if food is eaten} \\ -10 & \text{if collision (death) with wall or snake's body} \\ -0.01 & \text{per step (time penalty)} \end{cases}$$

Условия завершения эпизода: terminated (столкновение со стеной или с телом змейки), truncated (достигнут максимальный лимит шагов 1_000 или змейка заполнила все поле).

Поддерживается рендеринг в режимах «human» и «rgb_array».

Реализация алгоритмов

Алгоритм 1: DQN

Использует нейросеть (QNetwork) для аппроксимации Q-функции в непрерывном 13-мерном пространстве состояний.
Код соответствует формуле, обновление $\theta_{frozen} \leftarrow \theta$ происходит по умолчанию через каждые 100 шагов с помощью внешнего вызова функции:
`self.target_net.load_state_dict(self.q_net.state_dict()).`

Для оптимизации TD-error используется MSE loss:

$$\mathcal{L}(\theta) = \mathbb{E}_{(s,a,r,s')\sim D}[(r + \gamma * max_{a'}Q(s', a', \theta_{frozen}) - Q(s, a, \theta))^2]$$

Обучение происходит по мини-батчам случайных наблюдений из Replay Buffer.

Алгоритм 2: Q-learning (табличный)

Код соответствует формуле. Алгоритм внутри себя снижает размерность состояния среды 13 → 7:

- бинарные признаки столкновения в 4 направлениях (4)
- координаты относительного направления еды (2)
- индекс текущего направление движения (1)

Обновление Q-значений выполняется независимо от того, какое действие было выбрано (off-policy).

Алгоритм 3: SARSA (табличный)

Применяет ту же дискретизацию состояния, но использует on-policy обучение: обновление Q-значений происходит на основе фактических выбранных след. действий. Требует передачи `next_action` в метод `learn`.

Во всех алгоритмах при обучении используется ε-greedy policy, при инференсе - greedy.

Эксперименты

Для сравнения обученных алгоритмов были получены статистики по 1_000 независимым эпизодам: среднее и стандартное отклонение по reward и food. При обучении и сравнении не применялся подбор гиперпараметров, использовались рекомендуемые значения.
Для обучения каждый алгоритм делал более 300 тыс. шагов в среде.

Результаты

Математическое ожидание числа съеденной еды > 10 за 1_000 шагов свидетельствует о том, что агенты успешно учатся избегать столкновений и эффективно находить еду — это хороший показатель качества обучения в данной среде.

Сопоставимое качество (близкие средние награды) у DQN, Q-learning и SARSA подтверждает, что логика обновления Q-значений и политики выбора действий реализована правильно.

Табличные методы обучались на порядок быстрее из-за маленького кол-ва параметров и отсутствием градиентной оптимизации.

Выводы

Данный проект дал:
1. освоить создание простых сред с полным наблюдением состояний, совместимых с интерфейсом gymnasium.Env
2. реализовать и углубиться в практическом применении value-based RL-алгоритмов: нейросетевого DQN с Replay Buffer и Target Network и табличных Q-learning\SARSA.

Из-за использования бинарных флагов препятствий вместо расстояния - алгоритмы получились близорукими в части планирования. Соответственно одно из направлений по улучшению - формирование более информативного представления состояния среды.

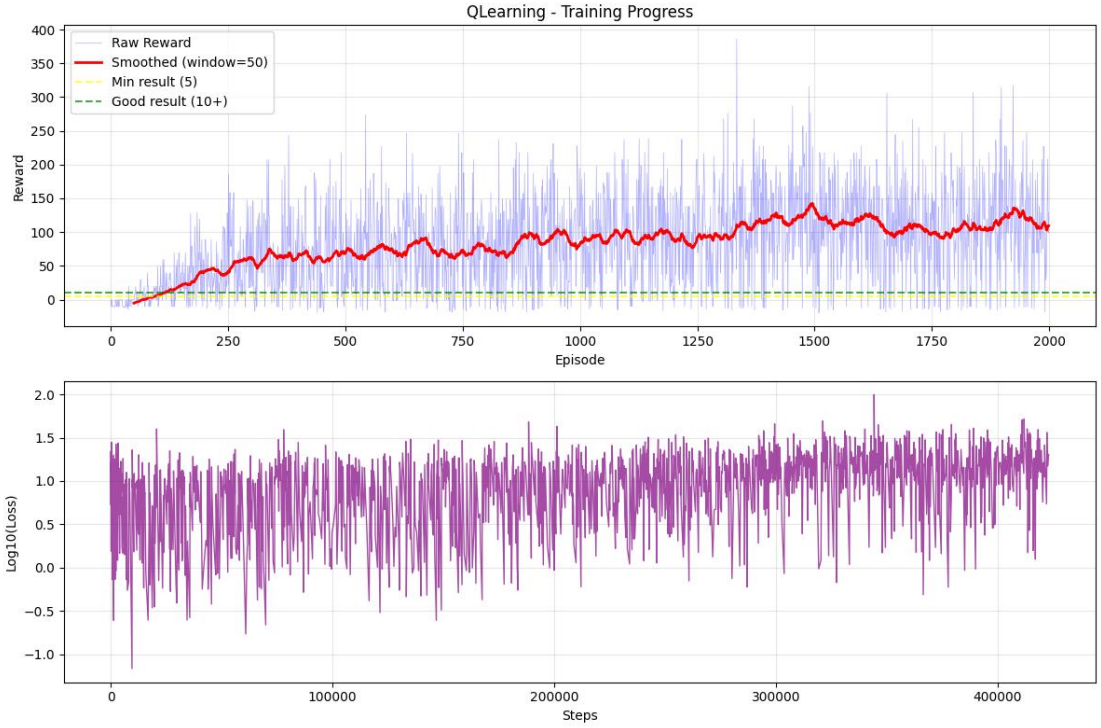
Сравнение «DQN vs Табличные алгоритмы» нельзя назвать корректным, т.к. для принятия решения использовались разные форматы состояния среды: на сыром представлении табличные методы не обучаются. Соответственно одно из направлений по улучшению - сравнить алгоритмы на сопоставимом дискретизированном состоянии среды (скорость сходимости, кол-во параметров, и матожидания отдачи).

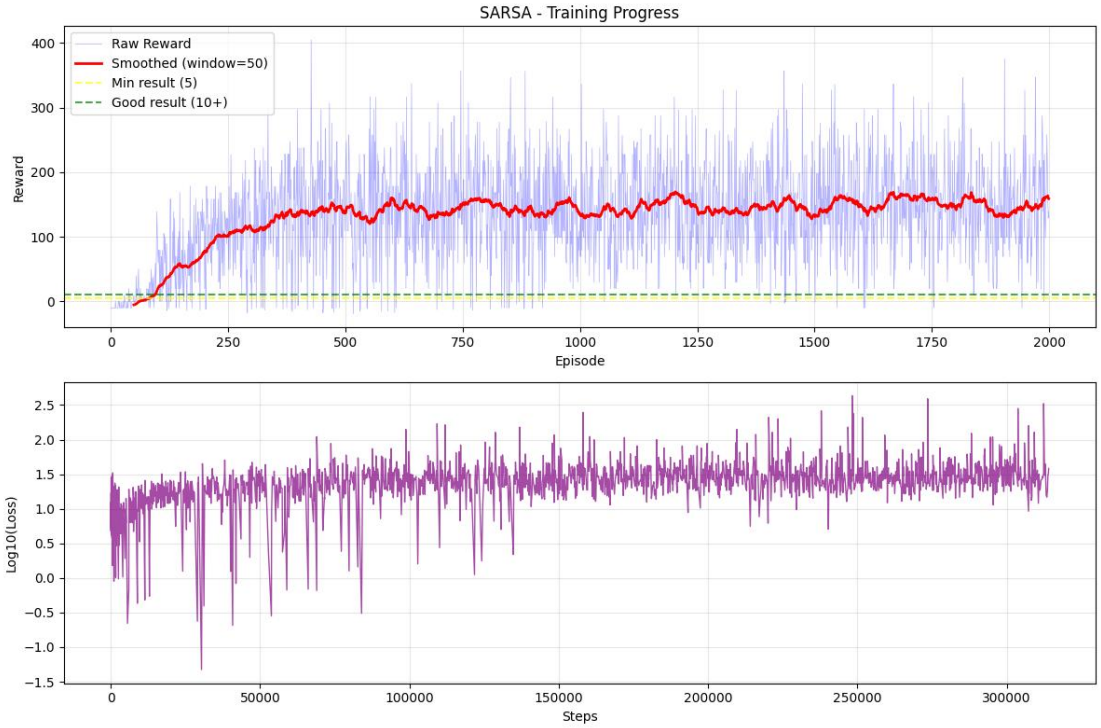
Приложение

Таблица 1. Сравнение статистик обученных алгоритмов на 1_000 эпизодах

Algorithm	Mean foods	Std foods	Mean reward	Std reward
DQN	18	8.112	168.472	80.251
QLearning	20	6.148	188.178	60.806
SARSA	17.6	5.783	164.478	57.028

Кривые обучения алгоритмов:


QLearning - Training Progress
Raw Reward
Smoothed (window=50)
Min result (5)
Good result (10+)
Reward
Episode
Log10(Loss)
Steps


SARSA - Training Progress
Raw Reward
Smoothed (window=50)
Min result (5)
Good result (10+)
Reward
Episode
Log10(Loss)
Steps