Содержание

1 Введение

Даннная работа посвящена прохождению популярной карточной roguelike игры Slay the Spire. Для этого был использован Q-learning - один из алгоритмов вида обучение с подкреплением.

2 Среда

В рукописной среде были реализованы 2 из 4 возможных первых битв игры за класс "Латоносец". Соответственно были реализованы начальные карты, 2 вида мобов (Культист и Зубастый Червь) и базовые механики боя. Бой происходит пошагово. Игрок(агент) знает, какой следующий ход сделает противник. У игрока есть колода из 10 карт: 5 атаки, 4 защиты и одна особой атаки. На каждый ход игроку дается 5 карт, у каждой есть стоимость, за сколько энергии она может быть разыграна. На 1 ход у игрока есть 3 энергии, не обязательно тратить всю, можно пропустить свой ход. Карта особой атаки, как и монстры могут накладывать на себя и на врага различные баффы/дебаффы. На текущий момент есть Сила(повышает силу всех атак), Уязвимость(персонаж с ней получает на 50% больше урона) и Ритуал(каждый ход увеличивает Силу персонажа).

3 Взаимодействие с агентом

3.1 Рукописная среда

Взаимодействие с агентом осуществляется с помощью стандартных потоков ввода/вывода. Среда сообщает агенту текущее состояние(в формате json), награду за его последнее действие и возможные действия на текущий момент. Агент в ответ возвращает номер действия из возможных, которое он считает наилучшим. Для запуска обученной модели в среде надо запустить run.sh.

3.2 Оригинальная среда(сама игра)

Используется Communication Mod, позволяющий посторонним программам взаимодействовать с игрой. Он так же работает на стандартных потоках ввода/вывода, сообщая текущее состояние и возможные действия, но уже без награды. Агент в ответ опять же возвращает выбранное и действие. Для запуска нужно следовать инструкциям мода, указав в конфиге команду "python3 play_game.py". В случае попадания моба, против которого агент играть не обучен, будет сделан автоматический рестарт.

4 Q-learning

В качестве метода обучения используется Approximate Q-learning. Для начала, что такое обычный Q-learning.

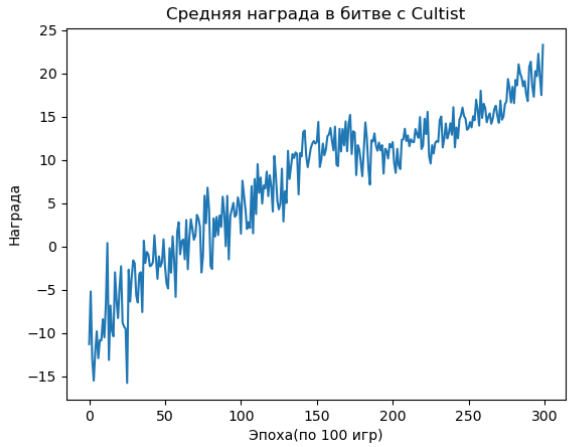
Представим себе таблицу из чисел. Каждое число означает, насколько хорошо для агента выполнить из состояния x действие y. Назовем эти числа "Q-values". Собственно, Q-learning - это алгоритм, исследующий эти значения и пытающийся сделать их как можно более оптимальными. За некоторые действия алгоритму даются какие-то численные награды, определяемые программистом. На каждой итерации алгоритм с некоторой вероятностью совершает случайный ход вместо того, который он считал наилучшим до этого. Если полученная награда внезапно окажется лучше, чем было до этого, то он изменит табличку и начнет считать этот ход выгоднее для себя.

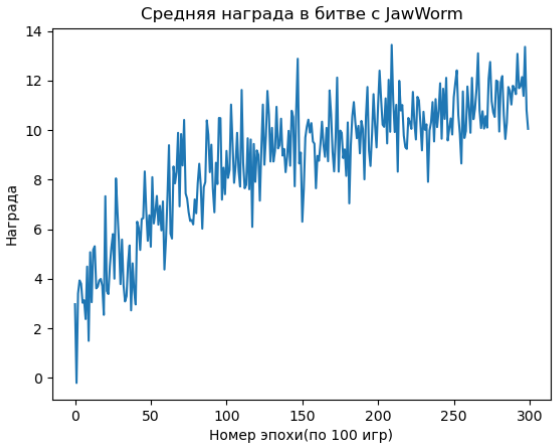
Теперь об Approximate Q-learning. Это то же самое, что обычный Q-learning, только для определения лучших ходов используется нейронная сеть из библиотеки pytorch.

5 Обучение

В качестве наград используется полученный и нанесенный урон. То есть при использовании карты атаки дается награда [кол-во нанесенного урона], при ходе противника отнимается 2· [кол-во полученного урона].

Результаты обучения в виде средних наград за эпохи(каждая эпоха это 100 игр):





6 Дальнейшая работа

Впоследствии можно продолжать обучать модель проходить игру, добавить другие битвы, обучить ходить по карте в выгодных направлениях и выбирать выгодные карты, которые дают в качестве награды за битвы, выбирать нужные пункты в событиях, встречающихся на карте наравне с битвами.