|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | | http://mai.ru/life/brand/mai.gif | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **"МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ**  **(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)"** | |
| Институт №3 «Системы управления, информатика и электроэнергетика» |
| Кафедра 307 «Цифровые технологии и информационные системы» |

|  |
| --- |
| **Курсовая работа по курсу: «Современные методы построения математических моделей на основе искусственных нейронных сетей»** |
|  |
| «Reinforcement learning на примере шахмат» |
| Выполнили студенты группы М30-226М-21    Копылов Д.А.  Зайцев М.Д |

Москва**.**2022

Оглавление

[Введение. 2](#_Toc119943320)

[Датасет 4](#_Toc119943321)

[Шахматный движок 5](#_Toc119943322)

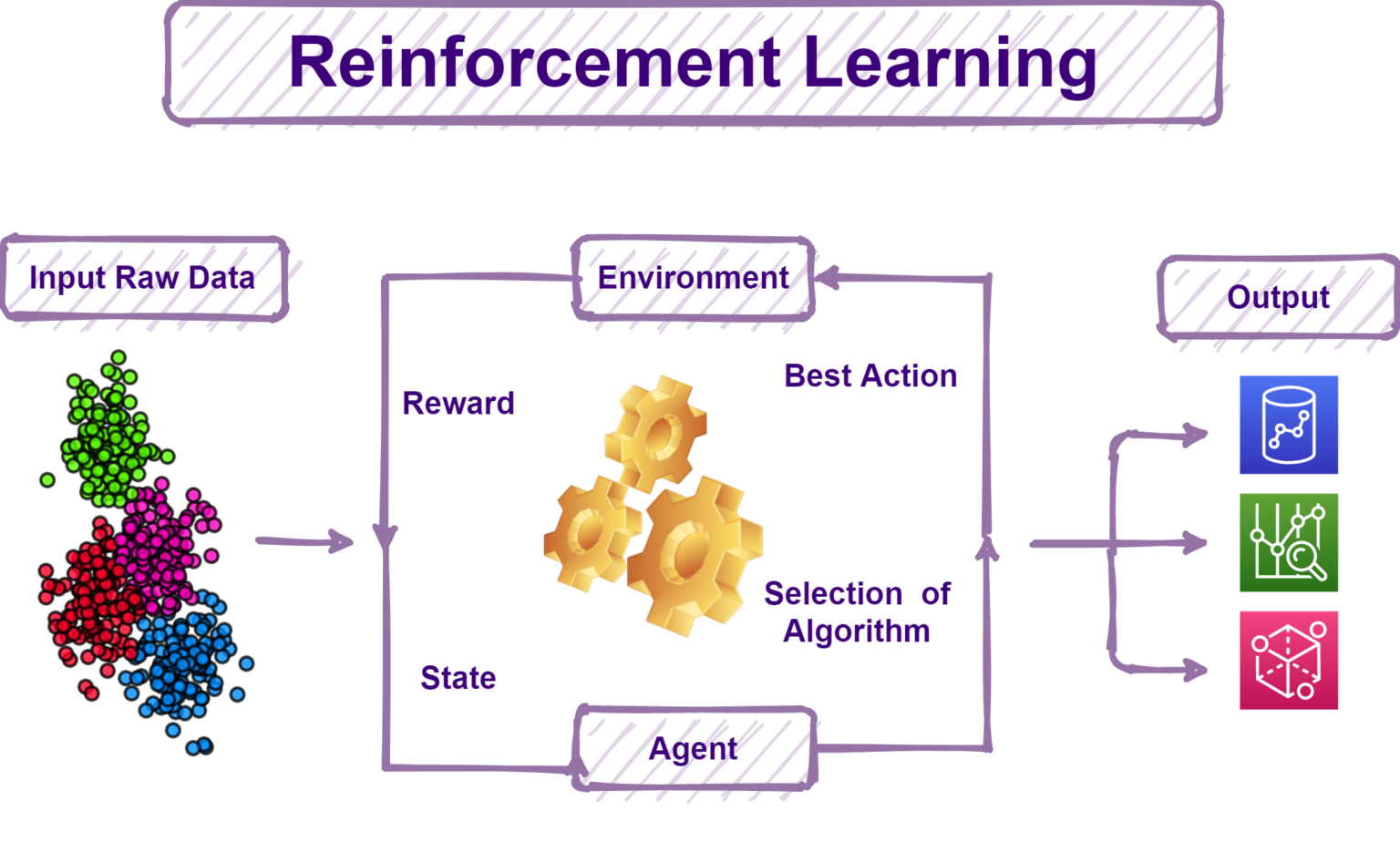
[Постановка задачи 5](#_Toc119943323)

[Обработка датасета 6](#_Toc119943324)

# Введение.

В настоящее время методы ИИ нашли своё применение почти во всех возможных отраслях, но большинство из сценариев применения являются автоматизации рутинных или давно известных процессов.

Отдельный интерес для оптимизации представляю так называемые игровые процесс. Игровые процессы представляют собой сценарий, при котором система должна взаимодействовать со средой, получая отклик и взаимодействуя с ним, а затем отправлять управляющее воздействие. В большинстве таких процессов общее количество возможных состояний среды не определено или стремится к бесконечности. Пример такой среды приведён ниже

Несмотря на то, что теория игры в шахматы не является предметом интерес науки, среда шахматной партии является хорошим тренажером для изучения алгоритмов обучения с подкреплением.

# Датасет

В целях базового исследования для данного проекта решено использовать датасет не из целых партий, а лишь из их концовок. Обучившись на таком наборе данных затем предполагается расширение его до начального положения и таким образом обучить алгоритм игре в шахматы.

На [просторах интернета](https://www.google.com/) найден датасет представляющий собой текстовые файлы с ~3000 партиями разделёнными на 4 варианта: 1 ход до победы, 2 хода до победы, 3 хода до победы, 4 хода до победы. Партии представлены в файлах в fen типе

# Шахматный движок

Для данного исследования был использован шахматный open-source движок Stockfish.

Stockfish – свободный шахматный движок с поддержкой UCI с открытым исходным кодом, доступный для различных настольных и мобильных платформ. Он разработан Марко Костальбой, Джоной Кийски, Гэри Линскоттом и Тордом Ромстадом, при большом вкладе сообщества разработчиков с открытым исходным кодом.

Stockfish занимает первые места большинства рейтинговых списков и соревнований среди компьютерных шахматных программ, и признаётся сильнейшей шахматной программой, не использующей GPU. Он выиграл неофициальный чемпионат мира по компьютерным шахматам в 6 сезоне (2014), 9 сезоне (2016), 11 сезоне (2018), 12 сезоне (2018), 13 сезоне (2018), 14 сезоне (2019) , 16 сезоне (2019),18 сезоне (2020) 19 сезоне (2020) , 20 сезоне (2021), 21 сезоне (2021) и в 22 сезоне (2022). Он финишировал вторым в 5 сезоне (2013), 7 сезоне (2014), 8 сезоне (2015), 15 (2019) и 17 (2020).

# Постановка задачи

В ходе исследовательской работы необходимо изучить алгоритмы Reinforcement learning в среде шахмат. Разработать и обучить систему способную решать шахматные задачи в один, два, три, четыре хода с использованием алгоритмов обучения с подкреплением. Для сравнения разработать игрока, выбирающего случайные ходы и проводить сравнение с ним. Дополнительно провести сравнение всех алгоритмов RE (Q-learning, DQN)

# Обработка датасета

Для последующего процесса обучения найденный датасет подавался на вход шахматному движку с целью получить не достающие шаги до победы. Недостающие шаги необходимы для последующей оценки ходов предлагаемых разрабатываемой системы. На выходе сформирован файл формата json (распространенный формат хранения и передачи данных). Пример полученного файла ниже:

# Random agent

Рисунок 1. Пример JSON файла.

3,11 строки fen позиции. 5,13 следующие состояния до победы. 8,16 ходы для победы

Как уже упоминалось ранее Random agent необходим как базовый игрок, с котором возможно проводить сравнение разрабатываемых алгоритмов.

Алгоритм работы:

Игровая среда возвращает доступные позиции для возможных ходов

Агент выбирает случайный ход из списка

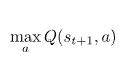
Возвращает среде, выбранный ход

# Q-learning agent

Метод q-learning основан на введении функции Q, отражающей ценность каждого возможного действия a агента для текущего состояния s, в котором сейчас находится симуляция. Или коротко:



Эта функция задает оценку агентом той награды, которую он может получить, совершив в определенный ход определенное действие. А также она включает в себя оценку того, какую награду агент может получить в будущем. Процесс обучения представляет из себя итерационное уточнение значения функции Q на каждом ходу агента. В первую очередь следует определить величину награды, которую агент получит в этот ход. Запишем ее переменной r\_t. Далее определим величину максимальной ожидаемой награды на последующих ходах:



Теперь следует решить, что для агента имеет большую ценность: сиюминутные награды или будущие. Данная проблема разрешается дополнительным коэффициентом при составляющей оценки последующих наград. В итоге предсказываемая агентом величина функции Q на данном шаге должна быть максимально приближена к значению:



Таким образом, ошибка предсказания агентом значения функции Q для текущего хода запишется следующим образом:



Введем коэффициент, который будет регулировать скорость обучения агента. Тогда формула для итерационного расчета функции Q имеет вид:



Общая формула итерационного расчета функции Q:



Алгоритм агента для исследуемого процесса:

Все эпохи пройдены?

Шах и мат?

Следующий ход

Инициализация класса RandomAgent

Инициализация класса Q-learningAgent

Цикл обучения на n эпох

Получение очередной партии из JSON

Цикл игры до стадии «Шах и Мат»

Игра/Обучение

Более подробная информация будет приведена позже

# DQN Agent

# Игра агентов между собой

# Выводы