Министерство науки и высшего образования Российской Федерации

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)

Институт прикладной информатики и компьютерных наук

ОТЧЕТ

По дисциплине «Глубинное обучение»

Задание 4 «Нейросетевое управление агентом»

Студент

Группы 932101

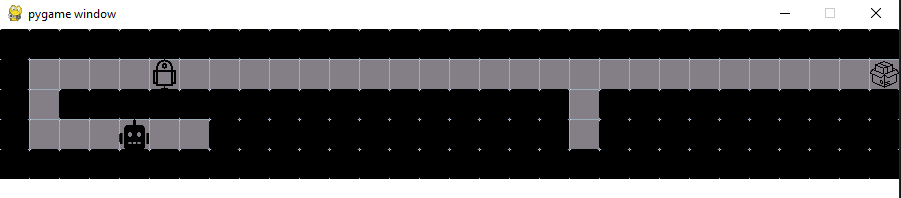
Насутион Рафли Аулиа Ризки

Томск – 2025

# **Цель работы**

получить навыки создания и обучения нейронных сетей для управления агентом в среде с неопределённостью.

1. **Карта области**

****

**Карта области**

На карте представлена сетка размером 30x5, которая содержит логистический центр (начальное положение робота), цель (посылка) и препятствия. Робот начинает в левом верхнем углу карты (ячейка [1, 1]). Цель и препятствия генерируются случайным образом при каждом запуске среды, при этом гарантируется, что:

* Цель и препятствия не занимают одно и то же место.
* Препятствия не блокируют путь робота к цели.

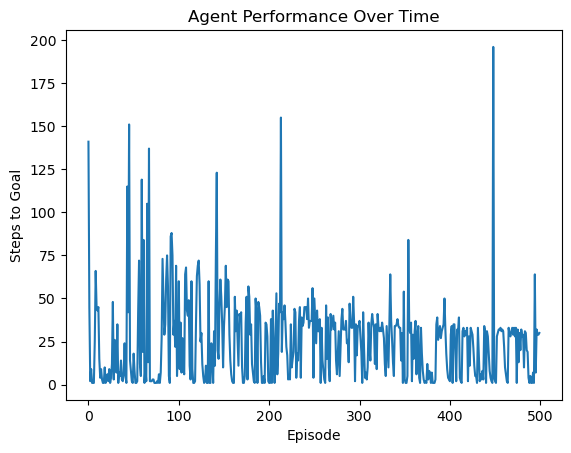
1. **Описание свёрточной архитектуры, использованной для настройки и управления агентом**

Архитектура модели состоит из следующих слоев:

1. **Conv2D (32 фильтра, размер ядра 3x3, активация ReLU):**  
   Этот слой применяется для извлечения пространственных признаков из входных данных (в данном случае, карты города). Он использует 32 фильтра с размером ядра 3x3 и функцию активации ReLU для нелинейности. Это помогает модели распознавать сложные паттерны в данных.
2. **Conv2D (64 фильтра, размер ядра 3x3, активация ReLU):**  
   Следующий сверточный слой увеличивает количество фильтров до 64, что позволяет модели захватывать более сложные и высокоуровневые признаки из карты города.
3. **Flatten:**  
   Этот слой преобразует двумерные данные из предыдущего слоя в одномерный вектор, который затем будет передан в полносвязный слой.
4. **Dense (64 нейрона, активация ReLU):**  
   Полносвязный слой с 64 нейронами и функцией активации ReLU. Этот слой позволяет модели комбинировать признаки, извлеченные в предыдущих слоях, для принятия более обоснованных решений.
5. **Dense (Размер действия, активация линейная):**  
   Выходной слой, который имеет количество нейронов, равное количеству доступных действий (в данном случае, 5). Используется линейная активация, поскольку агент должен предсказать Q-значения для каждого действия.
6. **Описание процесса настройки нейросети**

Процесс настройки нейросети заключается в следующем:

1. **Инициализация модели:**  
   При создании объекта класса Agent происходит инициализация модели с использованием указанной архитектуры. Входом модели служит карта города с размерностью (m, n, 1), где m и n — это размеры карты, а 1 — количество каналов (в данном случае, только один канал, так как используется черно-белая карта).
2. **Обучение модели:**  
   Во время каждой итерации обучения агент принимает решение о действии, используя модель, и получает обновление Q-значений с использованием алгоритма Q-обучения. Модель обновляется с помощью ошибки (разница между предсказанным значением Q и актуальным значением). Этот процесс повторяется для каждого действия, и модель обновляется с использованием метода обратного распространения ошибки.
3. **Функция потерь и оптимизация:**  
   Для обучения используется функция потерь MSE (среднеквадратичная ошибка), которая минимизируется с помощью оптимизатора Adam. Обновление весов происходит в процессе каждой итерации обучения, что позволяет агенту улучшать свою политику.
4. **Обучение с использованием эпизодов:**  
   Процесс обучения продолжается на протяжении нескольких эпизодов (по умолчанию 500). В каждом эпизоде агент выполняет действия в среде, получает вознаграждения и обновляет свою модель.
5. **Использование памяти:**  
   Агент использует память (deque), чтобы запоминать прошлые состояния, действия, вознаграждения и новые состояния. Это используется для того, чтобы агент мог обучаться на случайных выборках из истории, что способствует лучшему обобщению.
6. **Графики, иллюстрирующие изменение среднего количества шагов агента для достижения требуемой области на каждой итерации обучения сети**



1. **Программный код с комментариями**

<https://github.com/raflyaulya/lab4_deepLearning>