УДК 004.8

ОБУЧЕНИЕ С ПОДКРЕПЛЕНИЕМ ДЛЯ РЕАЛИЗАЦИИ ИГРОВОГО ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Васильев Н.С.

Чувашский государственный университет имени И. Н. Ульянова,
г. Чебоксары, Российская Федерация

Научный руководитель: Щипцова А.В. – канд.пед.наук, доцент, доцент кафедры ВТ

Аннотация. Данная статья затрагивает вопросы эффективности применения искусственных нейронных сетей при проектировании поведения интеллектуальных агентов. Основное внимание уделяется анализу методов построения моделей поведения интеллектуальных агентов. Также был проведен обзор существующих программных решений, которые реализуют данные методы в игровых приложениях. На основе результатов анализа была предложена новая архитектура поведения интеллектуальных агентов, которая использует автоматный подход для описания поведения агента. Данный проект реализован на языке программирования Python и использует Godot Engine в качестве среды для агентов. Тестирование разработанной модели поведения агента показало, что она может быть использована для создания игрового искусственного интеллекта, который может предложить достаточно сложное поведение игровых персонажей в нестандартных ситуациях.

Ключевые слова: генетический алгоритм, интеллектуальный агент, искусственный интеллект, машинное обучение, неигровой персонаж, обучение с подкреплением, Godot Engine.

Введение. Обучение с подкреплением — это один из наиболее эффективных методов машинного обучения, который может быть использован для создания интеллектуальных агентов (неигровых персонажей) в компьютерных играх. Обучение с подкреплением относится к генеративному моделированию [1]. Неигровой персонаж — персонаж в компьютерных играх, которым управляет не игрок, а компьютер. Обучение с подкреплением позволяет агенту обучаться на основе опыта, полученного во время взаимодействия с окружающей средой, и принимать оптимальные решения в различных ситуациях.

В статье рассматриваются и предлагаются способы использования обучения с подкреплением в игровом движке Godot (открытый кроссплатформенный 2D и 3D игровой движок под лицензией MIT), анализируются преимущества и недостатки этого подхода для создания сложного поведения агента. Анализируются результаты эксперимента применения одной и нескольких нейронных сетей для моделирования сложного поведения агентов.

Основная часть. Для реализации игрового искусственного интеллекта, авторами ставятся для решения следующие задачи:

- создать в движке Godot идеальную окружающую среду для обучения агентов;
- определить модель позитивного и негативного подкрепления агента в зависимости от целей агента в игре;
- обучить агента с помощью нейронной сети.

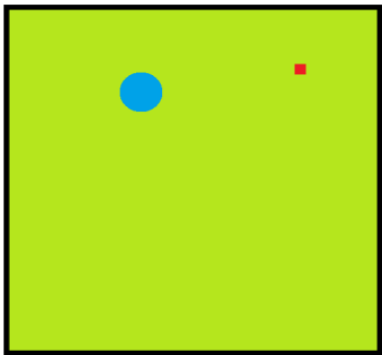

Рисунок 1 – Арена с изображением агента (круг) и бонуса (квадрат)

Для эксперимента была выбрана игра, цель которой - накопление агентом (круг на рисунке 1) бонусов (квадрат на рисунке 1), за которые назначаются очки. В качестве среды был, а создана «арена», на которой появляются бонусы.

Арена и агент были разработаны средствами Godot Engine с применением языка GDScript [2]. Обучение агента происходит по генетическому алгоритму. Создаются несколько экземпляров «арен» с агентами и бонусами, в которых агенты конкурируют между собой. Критерием успеха агента является скорость накопления бонусов. Для того чтобы обучение агента происходило быстрее, он должен получать небольшое позитивное или негативное подкрепление S, когда будет приближаться к бонусу или отдаляться от бонуса, соответственно, по формуле (1):

$$S = k * (|B - A_1| - |B - A_2|), \quad (1)$$

где B – позиция бонуса;

A_1 – позиция агента до принятия решения;

A_2 – позиция агента после принятия решения;

k – коэффициент, контролирующий размер награды агенту за факт изменения расстояния до бонуса.

Коэффициент k, подбирается в зависимости от целевого поведения агента в условиях окружающей среды (арены). В условиях «идеальной» окружающей среды (без препятствий, без других персонажей, влияющих на поведение агента) k=1. Для начала обучения агента с помощью нейронной сети, необходима синхронизация игры, работающей на движке Godot, и нейронной сети, реализуемой на Python и обучаемой с помощью генетического алгоритма. Синхронизация между Python и Godot обеспечивается по протоколу TCP. Расширение Godot «RL Agents» предоставляет эту возможность [3].

Игровой эксперимент показал, что решение вышеназванных задач, позволяет агенту успешно справляться с поиском бонусов уже через несколько минут обучения. Для исследования агента в неидеальной окружающей среде было принято решение об испытании конечно-автоматной модели поведения агента, самой известной реализацией которой является Goal-Oriented Action Planning (GOAP).

Использование только одной нейросети для создания агента со сложным поведением имеет следующие явные недостатки:

- длительный процесс обучения агента;
- сложность моделирования позитивного и негативного подкрепления агента;
- ограниченность паттернов поведения.

Как показал эксперимент, одним из решений этих проблем может являться разбиение логики поведения агента на отдельные модули и применение в них своей нейросети. В таком случае отдельная нейросеть будет отвечать за обход или преодоление агентом препятствий, укрытие агента от “негативных” персонажей т.п. Причем, переключение между нейросетями может выполняться на базе алгоритма, в котором прописано переключение, или с помощью еще одной нейросети. Переключение между нейросетями будет означать смену состояния агента. Чтобы поощрять разнообразие состояний и, как следствие, получать более интересное поведение агента, в процессе исследования была получена формула (2) для расчета коэффициента P, повышающего/понижающего сумму очков агента по итогу игры:

$$P = k / \sum_{i=1}^{n} \left(a_i \frac{T}{n} - T_i \right)^2, \quad (2)$$

где T – время существования агента;

T_i – время нахождения агента в i-ом состоянии;

a_i – коэффициент значимости для i-ого состояния;

n – количество всевозможных состояний;

k – коэффициент, контролирующий размер награды агенту за факт изменения расстояния до бонуса.

Еще один вид поощрения агентов может быть связан с временем существования агента (до гибели от «негативного» персонажа, либо до победы в игре) или от целевого состояния агента. Однако, существует проблема использования нейросетей в играх. Нейросеть становится слишком «умной» и «беспощадной» по отношению уже к игровым агентам (человеку). И в таком случае для поддержания интереса у игрока есть несколько способов «ослабить» искусственный интеллект:

- ограничение возможностей агента (замедление передвижения, ограничение действий за единицу времени);
- передача ложной информации агенту;
- ограничение входной информации агенту.

Заключение. Качественная реализация игрового искусственного интеллекта требует решения таких задач, как создание среды для обучения агентов, определение и моделирование способов позитивного и негативного подкреплений, использование набора нейронных сетей для поддержки логики агента, ограничение возможностей искусственного интеллекта.

Список литературы

1. Васильев, Н.С. Генеративные возможности нейронных сетей / Н.С. Васильев, А.В. Щипцова // Информатика и вычислительная техника: сб. науч. тр. -Чебоксары: Изд-во Чуваш. ун-та, 2022. -С. 25-28.

2. *Godot RL Agents [Электронный ресурс]* – Режим доступа : https://github.com/edbeeching/godot_rl_agents – Дата доступа : 22.03.2023.

3.Creating Custom Environments with Godot RL Agents [Электронный ресурс] – Режим доступа : https://github.com/edbeeching/godot_rl_agents/blob/main/docs/CUSTOM_ENV.md – Дата *доступа : 22.03.2023.*

UDC 004.8

REINFORCEMENT LEARNING FOR THE IMPLEMENTATION OF GAMING ARTIFICIAL INTELLIGENCE

Vasiliev N.S.

I. N. Ulyanov Chuvash State University, Cheboksary, Russian Federation

Shchiptsova A.V. – PhD, associate professor, associate professor of the Department of Computer Engineering

Annotation. This article addresses the issues of the effectiveness of the use of artificial neural networks in the design of the behavior of intelligent agents. The main attention is paid to the analysis of methods for constructing models of behavior of intelligent agents. There was also a review of existing software solutions that implement these methods in gaming applications. Based on the results of the analysis, a new behavior architecture of intelligent agents was proposed, which uses an automaton approach to describe the behavior of an agent. This project is implemented in the Python programming language and uses Godot Engine as an environment for agents. Testing of the developed agent behavior model has shown that it can be used to create a game artificial intelligence that can offer quite complex behavior of game characters in non-standard situations.

Keywords: genetic algorithm, intelligent agent, artificial intelligence, machine learning, non-player character, reinforcement learning, Godot Engine