

Министерство науки и высшего образования Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого

Название института

Работа допущена к защите
Должность руководителя М
_____ И.О. Фамилия
«_____» _____ 202X г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

РАБОТА БАКАЛАВРА

ТЕМА ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

по направлению подготовки XX.XX.XX Наименование направления подготовки
Направленность (профиль) XX.XX.XX_YY Наименование направленности (про-
филя) образовательной программы

Выполнил

студент гр. N

И.О. Фамилия

Руководитель

должность,

степень, звание¹

И.О. Фамилия

Консультант²

должность, степень

И.О. Фамилия

Консультант

по нормоконтролю³

И.О. Фамилия

Санкт-Петербург

202X

¹ Должность указывают сокращенно, учёную степень и звание — при наличии, а подразделения — аббревиатурами. «СПбПУ» и аббревиатуры институтов не добавляют.

² Оформляется по решению руководителя ОП или подразделения. Только 1 категория: «Консультант». В исключительных случаях можно указать «Научный консультант» (должен иметь степень). Без печати и заверения подписи.

³ Обязателен, из числа ППС по решению руководителя ОП или подразделения. Должность и степень не указываются. Сведения помещаются в последнюю строчку по порядку. Рецензенты не указываются.

**САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
ПЕТРА ВЕЛИКОГО**
Название института

УТВЕРЖДАЮ

Должность руководителя М

_____ И.О. Фамилия

« _____ » _____ 202Хг.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

студенту Фамилия Имя Отчество гр. N

1. Тема работы: Тема выпускной квалификационной работы.
2. Срок сдачи студентом законченной работы⁴: дд.мм.202Х.
3. Исходные данные по работе⁵:
 - 3.1. Фреймворки для разработки алгоритмов обучения с подкреплением (pytorch и другие).
 - 3.2. Программные среды тестирования алгоритмов (OpenAI Gym и другие).
 - 3.3. Алгоритмы обучения с подкреплением.
 - 3.4. Теория управления МАС.
4. Содержание работы (перечень подлежащих разработке вопросов):
 - 4.1. Обзор технологии обучения с подкреплением.
 - 4.2. Обзор подходов к управлению мультиагентными системами (МАС).
 - 4.3. Обзор аналогичных исследований.
 - 4.4. Постановка задачи выпускной квалификационной работы.
 - 4.5. Разработка алгоритмов управления агентами в МАС. Их исследование.
 - 4.6. Внедрение разработанных алгоритмов в компьютерную игру, тестирование.
 - 4.7. Экспериментальное исследование алгоритмов.

⁴Определяется руководителем ОП, но не позднее последнего числа преддипломной практики и/или не позднее, чем за 20 дней до защиты в силу п. 6.1. «Порядка обеспечения самостоятельности выполнения письменных работ и проверки письменных работ на объем заимствований».

⁵Текст, который подчеркнут и/или выделен в отдельные элементы нумерационного списка, приведён в качестве примера.

- 4.8. Заключение, выводы об эффективности исследованных алгоритмов.
- 5. Перечень графического материала (с указанием обязательных чертежей):
 - 5.1. Графики результатов экспериментов.
 - 5.2. Блок-схемы алгоритмов.
 - 5.3. Скриншоты
- 6. Консультанты по работе⁶:
 - 6.1. Должность, степень, И.О. Фамилия (нормоконтроль).
- 7. Дата выдачи задания⁷: дд.мм.202Х.

Руководитель ВКР _____ И.О. Фамилия

Задание принял к исполнению дд.мм.202Х

Студент _____ И.О. Фамилия

⁶Подпись консультанта по нормоконтролю пока не требуется. Назначается всем по умолчанию.

⁷Не позднее 3 месяцев до защиты (утверждение тем ВКР по университету) или первого числа преддипломной практики или по решению руководителя ОП или подразделения (открытый вопрос).

РЕФЕРАТ

На 28 с., 8 рисунков, 1 таблицу, 0 приложений.

КЛЮЧЕВЫЕ СЛОВА: СТИЛЕВОЕ ОФОРМЛЕНИЕ САЙТА, УПРАВЛЕНИЕ КОНТЕНТОМ, PHP, MYSQL, АРХИТЕКТУРА СИСТЕМЫ⁸

В данной работе изложена сущность подхода к созданию динамического информационного портала на основе использования открытых технологий Apache, MySQL и PHP. Даны общие понятия и классификация IT-систем такого класса. Проведен анализ систем-прототипов. Изучена технология создания указанного класса информационных систем. Разработана конкретная программная реализация динамического информационного портала на примере портала выбранной тематики.⁹

В данной работе изложена сущность подхода к созданию динамического информационного портала на основе использования открытых технологий Apache, MySQL и PHP. Даны общие понятия и классификация IT-систем такого класса. Проведен анализ систем-прототипов. Изучена технология создания указанного класса информационных систем. Разработана конкретная программная реализация динамического информационного портала на примере портала выбранной тематики.

ABSTRACT

On 28 p., 8 figures, 1 tables, 0 appendices.

KEYWORDS: STYLE REGISTRATION, CONTENT MANAGEMENT, PHP, MYSQL, SYSTEM ARCHITECTURE

In the given work the essence of the approach to creation of a dynamic information portal on the basis of use of open technologies Apache, MySQL and PHP is stated. The general concepts and classification of IT-systems of such class are given. The analysis of systems-prototypes is lead. The technology of creation of the specified class

⁸Всего **слов:** от 3 до 15. Всего **слов и словосочетаний:** от 3 до 5. Оформляются в именительном падеже множественного числа (или в единственном числе, если нет другой формы), оформленных по правилам русского языка.

⁹До 600 печатных знаков (ГОСТ Р 7.0.99-2018 СИБИД) на русский или английский текст. Текст реферата повторён дважды на русском и английском языке для демонстрации подхода к нумерации страниц. *Внимание! Эта сноска размещена после точки. Это пример как не нужно оформлять сноску.*

of information systems is investigated. Concrete program realization of a dynamic information portal on an example of a portal of the chosen subjects is developed.

In the given work the essence of the approach to creation of a dynamic information portal on the basis of use of open technologies Apache, MySQL and PHP is stated. The general concepts and classification of IT-systems of such class are given. The analysis of systems-prototypes is lead. The technology of creation of the specified class of information systems is investigated. Concrete program realization of a dynamic information portal on an example of a portal of the chosen subjects is developed.

СОДЕРЖАНИЕ

Введение	6
0.1. Вопросы исследования	6
0.2. Сценарии	7
0.2.1. Сценарий 1: Simple Speaker Listener.....	7
0.2.2. Сценарий 2: Simple Reference	8
0.3. Практическая значимость.....	8
Глава 1. Обзор технологии обучения с подкреплением.....	10
1.1. Введение.....	10
1.2. Искусственный Интеллект.....	10
1.3. Машинное Обучение	11
1.3.1. Обучение с учителем (Supervised Learning).....	11
1.3.2. Обучение без учителя (Unsupervised Learning)	13
1.3.3. Обучение с подкреплением (Reinforcement Learning).....	13
1.4. Глубокое обучение	14
1.4.1. Искусственная нейронная сеть	14
1.4.2. Глубокая нейронная сеть	15
1.5. Обучение с подкреплением	17
1.5.1. Алгоритмы глубокого обучения с подкреплением.....	18
1.5.2. Подход, основанный на функции состояния (Value Based подход)	18
1.5.3. Линия поведения (Policy Based)	20
Заключение	23
Список сокращений и условных обозначений	24
Словарь терминов.....	25
Список использованных источников.....	26

ВВЕДЕНИЕ

В этой работе исследуется совместная работа нескольких агентов с использованием алгоритмов и методов глубокого обучения в 2D игровых средах. Имея это в виду, мы изучили и адаптировали современные алгоритмы и методы обучения глубокого обучения с подкреплением к настройкам игры с несколькими агентами.

Глубокое обучение с подкреплением - это новая область исследований алгоритмов и методов, которая сочетает в себе обучение с подкреплением и глубокое обучение. Предыдущие работы в основном были направлены на адаптацию глубоких нейронных сетей для усиления алгоритмов обучения. Например, глубокая Q-сеть (Deep Q-network, DQN) [14] интегрирует глубокие нейронные сети в Q-обучение, классический табличный алгоритм обучения с подкреплением. Обученные сети могут играть в различные игры Atari 2600 [27] лучше человека. Это считается первой успешной попыткой научиться играть в видеоигры с прямым визуальным вводом большого размера. Алгоритм глубокого детерминированного градиента политики (deep deterministic policy gradient, DDPG) [6] является еще одним примером использования глубоких нейронных сетей в контексте обучения с подкреплением и непрерывного пространства действий.

Большинство исследований посвящено обучению одного агента. Однако проблемы, связанные с мультиагентным сотрудничеством или конкуренцией, также очень распространены в социальной, экономической и инженерной областях. Игры представляют собой упрощенные версии реальных задач, которые можно сделать идеальными тестовыми площадками для экспериментов. Поэтому в этой работе игровые сценарии используются для изучения алгоритмов глубокого обучения с подкреплением для совместной работы нескольких агентов.

0.1. Вопросы исследования

В этой работе мы исследуем следующие вопросы:

1. Как несколько агентов могут научиться сотрудничать друг с другом во время обучения в определенных игровых сценариях?
2. Может ли после обучения появиться язык между агентами в определенных игровых сценариях?
3. Как можно оптимизировать и ускорить процесс обучения?

Сначала мы изучим современные алгоритмы и методы глубокого обучения с подкреплением. Затем мы проведем эксперименты игровых сценариев в модифицированной среде от компании OpenAI [19]. Наконец, мы рассмотрим и применим различные приемы для оптимизации процесса обучения.

0.2. Сценарии

Чтобы исследовать вышеизложенные вопросы, мы используем несколько сценариев, где агенты общаются друг с другом и физически перемещаются к определенным целям. Некоторые из сценариев являются точными копиями экспериментов, проведенных в недавней работе [18], а другие представляют собой новые сценарии, которые расширяют исходные с целью дальнейшего изучения многоагентного сотрудничества и коммуникации.

0.2.1. Сценарий 1: *Simple Speaker Listener*

В этом сценарии есть три ориентира, представленные как красные, зеленые и синие ориентиры, как показано на рисунке. Два агента с разными функциями должны сотрудничать для достижения общей цели. *Говорун* (серый агент) не может двигаться, но видит цвет цели и может говорить с другим агентом. *Слушатель* (отображаемый таким же цветом, что и его цель) видит все ориентиры и их цвета (но не видит собственный цвет, т.е. не знает, какой из объектов является его целью), а так же слышит *говоруна* и пытается перейти к правильному ориентиру. Более подробные настройки среды этого сценария можно найти в разделе X.X.

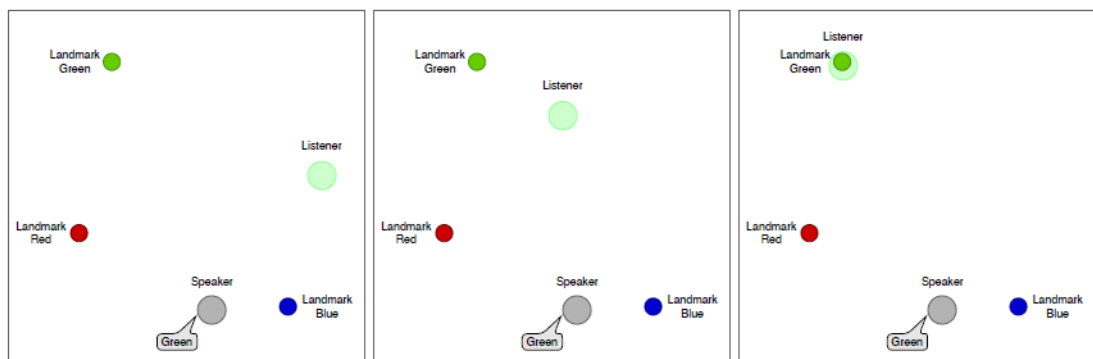


Рис.0.1. Сценарий 1. *Simple Speaker Listener*. Скриншоты слева направо показывают этапы игрового эпизода. *Говорун* (серый) выдает коммуникационное действие, представляющее «зеленый», а *слушатель*, слушает сообщение *говоруна* и направляется к цели.

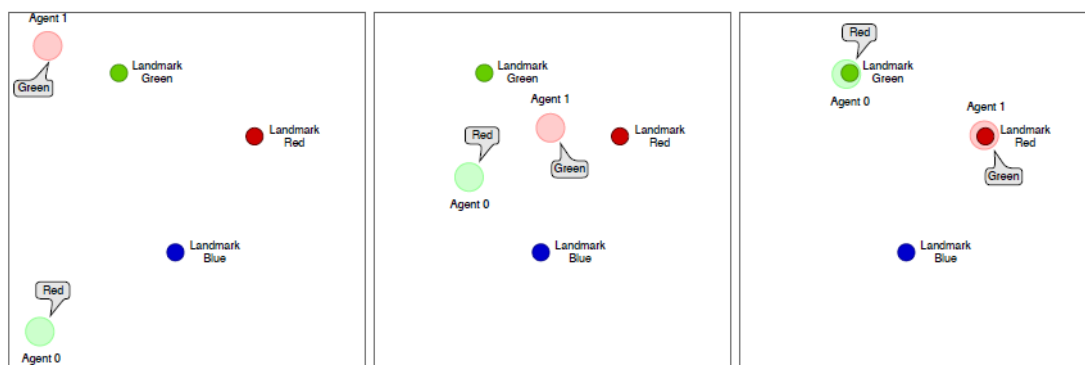


Рис.0.2. Сценарий 2: *Simple Reference*. В этом эпизоде агент 0 (отображаемый в том же цвете, что и его целевой ориентир) выдает коммуникационное действие, представляющее «красный», и слушает агента 1. А агент 1 (отображается в том же цвете, что и его целевой ориентир), слушает и издает «зеленый» для Агента 0. Скриншоты слева направо показывают, как ведут себя два агента в соответствии с оптимальными политиками, слушая друг друга и перемещаясь к целям.

0.2.2. Сценарий 2: *Simple Reference*

Этот сценарий расширяет предыдущий, поскольку оба агента являются одновременно и *говорунами* и *слушателями*. Ориентиры остаются прежними, отображаются как три разноцветных объекта, (как изображено на рис.0.2). Каждый из двух агентов пытается достичь своего целевого ориентира, который известен только другому агенту. Таким образом, он должен научиться сообщать другому агенту его цель и перемещаться к своей собственной. Что отличает сценарий от двух копий сценария *Simple Speaker Listener*, так это то, что единое общее вознаграждение, присуждаемое агентам, основано на общей производительности. Таким образом, агенты должны выяснить, что идет хорошо, а что нет. Настройка среды подробно описана в разделе X.X.

0.3. Практическая значимость

Проект сосредоточен на обучении нескольких агентов совместной работе с использованием глубокого обучения в игровой среде. Результаты исследования могут быть дополнительно разработаны и широко использованы во многих практических реальных приложениях в области экономики, управления, техники и т. д. Например, он может применяться в робототехнике, к автономным транспортным средствам, производственным линиям, фондовым рынкам и т. д.

Перед применением в этих областях, алгоритмы и методы, которые исследуются и разрабатываются в этой работе, должны быть хорошо протестированы и проверены, поскольку эти приложения непосредственно влияют на безопасность

человека, социальную и финансовую безопасность. В долгосрочной перспективе применение мультиагентных алгоритмов приведет к появлению все большего числа автономных систем во многих областях.

ГЛАВА 1. ОБЗОР ТЕХНОЛОГИИ ОБУЧЕНИЯ С ПОДКРЕПЛЕНИЕМ.

1.1. Введение.

Алгоритмы и методы глубокого обучения с подкреплением (Deep Reinforcement Learning, DRL) – это методы, которые мы используем для исследования вопросов дипломной работы. DRL – это подраздел машинного обучения, который лежит на пересечении Обучения с подкреплением (Reinforcement Learning) и глубокого обучения (Deep Learning).

Прежде чем говорить о DRL, нужно кратко рассмотреть понятия, связанные с искусственным интеллектом и машинным обучением.

На рис.1.1 представлен обзор концепций, которые мы вводим в этой главе.

Сначала мы рассмотрим искусственный интеллект, машинное обучение и его три категории. Затем мы углубимся в глубокое обучение. Наконец, мы рассмотрим основные алгоритмы DRL. Таким образом, мы можем понять, почему и как DL и RL объединяются в DRL, который позволяет агентам взаимодействовать с более сложной средой и вести себя более разумно.



Рис.1.1. Граф, показывающий концепты МО и ИИ и их отношения.

1.2. Искусственный Интеллект

Искусственный интеллект [7] (ИИ), как следует из названия, в отличие от естественного интеллекта человека и других животных, является искусственным. Именно такой тип интеллекта люди реализуют в машинах. Конечной целью ИИ является создание таких автономных систем, которые способны учиться

методом многочисленных проб и ошибок, чтобы найти оптимальное поведение для достижения максимально возможных целей в сложившемся окружении. [21]

С 21 века, благодаря нескольким прорывам, ИИ процветает в викторинах и настольных играх, превосходя уровень игры людей [28] [1]. Наряду с увеличением вычислительной мощности, улучшения алгоритмов и доступности больших наборов данных, ИИ развивался революционными темпами. В ближайшем будущем, ИИ ещё сильнее повлияет на работу и повседневную жизнь человека.

1.3. Машинное Обучение

Если ИИ – это более широкая концепция машинного интеллекта для выполнения задач, которые пока выполняют люди, то Машинное обучение (ML) - это основной метод разработки ИИ, который не требует явного программирования [22]. ML позволяет компьютерам строить модели и применять алгоритмы при изучение больших объемов данных. Модели ML обучаются с использованием методов статистики, чтобы понять структуру набора данных или последовательности экспериментов. Обученные модели могут распознавать паттерны и делать очень точные прогнозы учитывая не очевидные данные или обрабатывать определенные задачи в неочевидных сценариях [5].

Основные категории алгоритмов ML показаны на рисунке рис.1.2. А именно обучение с учителем, обучение без учителя и обучение с подкреплением. Категории определяются тем, как алгоритмы и модели наполняются данными и как данные анализируются.

1.3.1. Обучение с учителем (*Supervised Learning*)

TODO: может быть добавить рисунок

Один из способов машинного обучения, в ходе которого испытуемая система принудительно обучается с помощью примеров «стимул-реакция». С точки зрения кибернетики, является одним из видов кибернетического эксперимента. Между входами и эталонными выходами (стимул-реакция) может существовать некоторая зависимость, но она неизвестна. Известна только конечная совокупность прецедентов — пар «стимул-реакция», называемая обучающей выборкой. На основе этих данных требуется восстановить зависимость (построить модель отношений

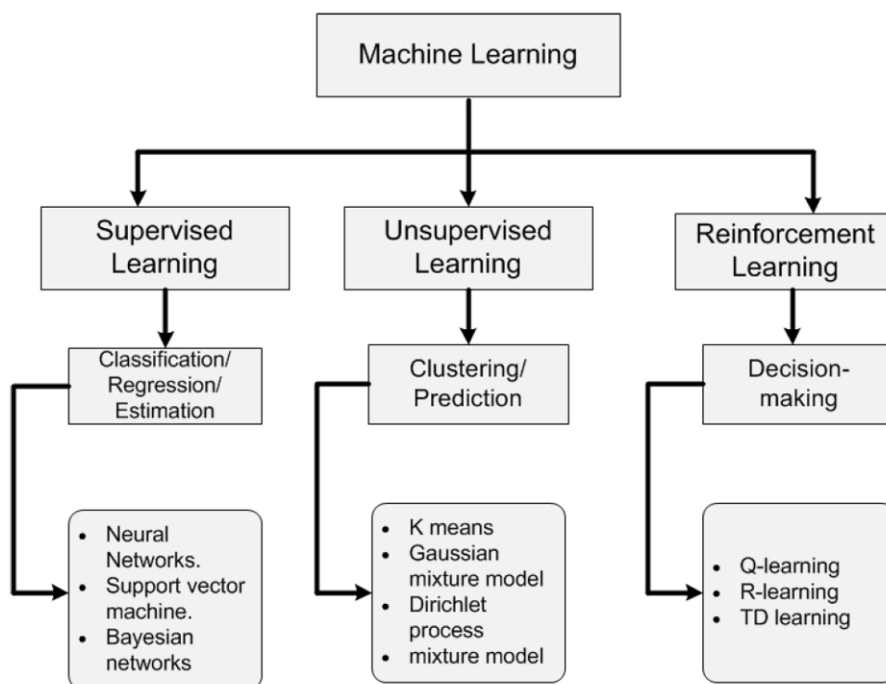


Рис.1.2. Категории ML и соответствующие алгоритмы. Основано на [23].

стимул-реакция, пригодных для прогнозирования), то есть построить алгоритм, способный для любого объекта выдать достаточно точный ответ [2].

Обучение с учителем можно разделить на *регрессию* и *классификацию*, в зависимости от того, являются ли выходные переменные количественными или качественными. Количественные переменные принимают числовые значения, в то время как качественные переменные принимают значения в одном из K различных классов или категории [2]. Например, прогноз цены на жилье по данным параметрам, таким как местоположение дома, общая площадь и количество комнат и т. д., это проблема регрессии. Диагноз рака - проблема классификации поскольку выходной сигнал либо положительный, либо отрицательный.

В обучении с учителем набор данных делится на обучающий, набор для валидации и тестовый набор. Обучающий набор представляет собой пары ввода и вывода переменных, которые напрямую вводятся в модель для обучения. Валидационный набор используется для контроля за переобучением модели. Наконец, тестовый набор используется для подтверждения того, что обученная модель обобщена и точна

Обучение с учителем - наиболее распространенная категория в машинном обучении, но оно требует больших наборов данных с правильными «ответами». Это может быть очень дорого, в некоторых случаях не практично.

1.3.2. Обучение без учителя (*Unsupervised Learning*)

TODO: может быть добавить рисунок

Это ещё одна важная категория в машинном обучении. В отличие от обучения с учителем, алгоритмы этой категории используют наборы данных, не размеченные «ответами». Задача в этой категории — обнаружить скрытую структуру в данных и распределить их по группам.

Эта категория алгоритмов получила своё название из-за отсутствия меток или выходных переменных. Кластеризация является типичным инструментом, который используется чтобы понять связь между наблюдениями и распределить их в разные группы [11]

При обучении без учителя данные не делятся на обучающие, проверочные и тестовые. Набор данных подается в модель напрямую и группируется в отдельные группы.

1.3.3. Обучение с подкреплением (*Reinforcement Learning*)

Последняя категория машинного обучения является междисциплинарной областью, которая сочетает в себе машинное обучение, неврологию, поведенческую психологию, теорию управления и т. д. Цель RL заключается в достижении целей без четких инструкций, но с наградами или штрафами, получаемыми от взаимодействий с окружающей средой.

Агент RL изучает оптимальную политику, последовательность действий, которая максимизируют общую будущую награду (Reward) [24].

В обучении с подкреплением агент RL наблюдает состояние s_t на этапе времени t , затем он взаимодействует с окружающей средой, выполняя действие a_t . Среда переходит в следующее состояние s_{t+1} , учитывая текущее состояние и выбранное действие, которое ведёт к получению агентом вознаграждения r_t . Цель агента узнать политику π , которая сопоставляет состояния с действиями, так что последовательность действий, выбранная агентом, максимизирует ожидаемое будущее вознаграждение. На каждом шаге взаимодействия со средой агент генерирует переход $\{s_t, a_t, s_{t+1}, r_t\}$, который даёт информацию, необходимую для улучшения политики, см. рис.1.3.

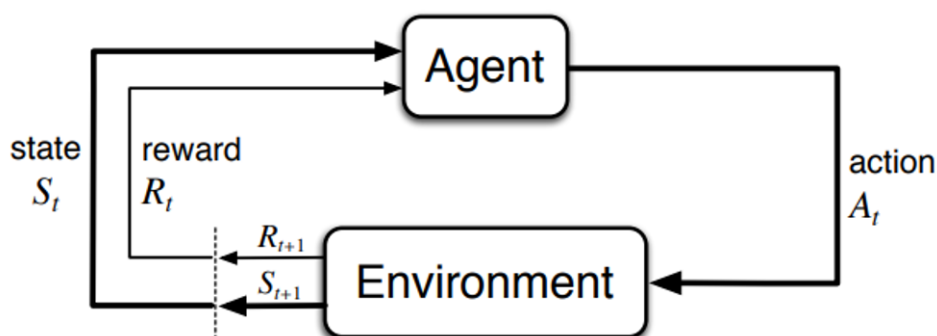


Рис.1.3. Агент взаимодействует с окружающей средой, сначала наблюдая состояние, затем совершая действие и, наконец, получая вознаграждения за выбранное действия. Через многочисленные попытки и ошибки, агент учится формулировать оптимальную политику [25].

1.4. Глубокое обучение

Глубокое обучение (Deep Learning, DL) - подраздел машинного обучения, в настоящее время достигает выдающихся результатов во многих областях, в том числе распознавание изображений, компьютерное зрение, распознавание речи и естественная языковая обработка. Глубокое обучение, как бы имитирует биологический мозг, обрабатывает информацию с помощью искусственных нейронных сетей.

Идея симуляции работы нейронов мозга человека зародилась десятилетия назад. Тем не менее, прорыв произошел в последние годы, когда стали доступны большие наборы данных и вычислительные мощности [9]. Теперь можно строить модели с большим количеством слоев искусственных нейронов, чем когда-либо прежде. С сильным увеличением глубины, сети достигают исключительной производительности в области распознавания изображений и речи. Считается, что глубокое обучение является одним из наиболее перспективных подходов для решения текущих задач ИИ.

1.4.1. Искусственная нейронная сеть

Мозг человека и животных - чрезвычайно сложный орган, который до сих пор до конца не изучен. Тем не менее, некоторые аспекты его структуры и функции были расшифрованы. Фундаментальным рабочим элементом в мозге является нейрон. Многочисленные нейроны связаны между собой сложным образом, что даёт возможность, запоминания, мышления и принятия решений.

Хотя нейроны сложны и функционируют разными способами, все они имеют некоторые базовые компоненты, такие как: ядро, дендриты, аксон и синапсы.

Дендриты действуют как входные каналы, через которые нейроны получают информацию от синапсов других нейронов. Затем ядро обрабатывает эти сигналы и превращает в вывод, который затем отправляется в другие нейроны. Связь этих компонентов играет роль линии передачи в нейронных сетях.

Хоть ИНС и не так сложны, как человеческий мозг, они имитируют базовую структуру, которая включает входные, выходные слои, а также, обычно, скрытые слои. В каждом слое есть искусственные нейроны. Нейроны в одном слое обычно связаны с каждым нейроном в следующем слое. Они передают числовые сигналы через соединения с другими нейронами, примерно, как это происходит в биологической нейронной сети [13].

Поскольку выходные сигналы нейронов ИНС представляются в виде вещественных чисел, выход обычно сравнивается с порогом. Только если порог превышен, нейрон передаёт сигнал следующим подключенным нейронам.

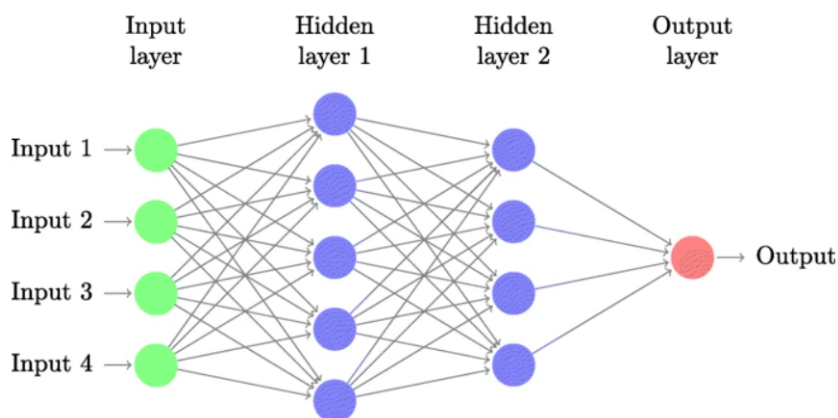


Рис.1.4. Базовая структура ИНС. ИНС обычно состоит из входного и выходного слоев, а также пары скрытых слоев. Основано на [16] [17].

Связи между нейронами параметризованы весами, которые обновляются во время обучения, чтобы отрегулировать мощность сигналов, проходящих через сеть [24]. рис.1.4 показывает базовую структуру ИНС.

1.4.2. Глубокая нейронная сеть

Как уже упоминалось выше, концепция нейронной сети не нова.

На ранних этапах из-за ограничений вычислительной мощности, нейронные сети имели очень малую глубину. Обычно он содержит только входной, выходной слои и пару скрытых слоев. Кроме того, количество нейронов в каждом слое также было ограничено. Глубокие нейронные сети не находили практического

применения до последних лет, когда стали доступны огромные вычислительные мощности и большие объемы данных.

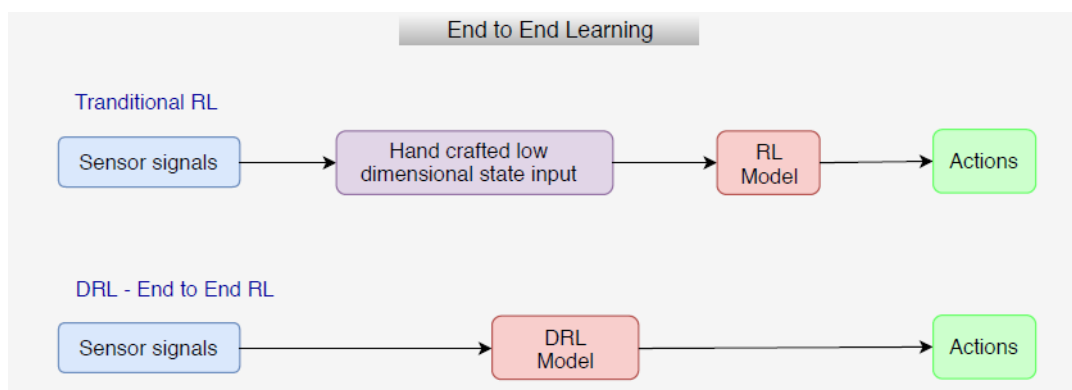


Рис.1.5. Сравнение DRL и традиционной RL, где необходимо явно указать, какие действия производить в каких состояниях. Использование глубокой нейронной сети в DRL позволяет обучаться и принимать решения на основе необработанного сенсорного ввода.

Глубокие нейронные сети с несколькими слоями хороши для извлечения скрытых свойств [15]. Каждый слой, решает свою собственную задачу. Узлы в каждом слое учатся на конкретных наборах свойств, которые приходят из предыдущего слоя. Теоретически чем глубже нейронная сеть, тем более сложны и абстрактны особенности, которые она может распознать, поскольку каждый нейрон агрегирует выводы из нейронов предыдущего слоя [3]. Например, в задаче распознавания изображений вход представляет собой матрицу пикселей. Первый слой извлекает начальные объекты, такие как ребра из пикселей, затем следующий слой кодирует расположение краев и следующий слой распознает глаза, рот, уши, ноги, крылья, хвост и т. д. Наконец, последний слой распознает на изображении кошку, собаку или птицу. Таким образом, глубокие нейронные сети не нуждаются во вмешательстве людей, а сами изучают иерархию объектов.

С математической точки зрения нейронная сеть определяет функцию $y = f(x; \theta)$. Она описывает соответствие входных данных x выходным y , где y — это категория в задаче классификации или выходное значение в задачах регрессии. Тренировка модели с использованием набора данных ведет к вычислению параметра θ .

После завершения обучения предполагается, что нейронная сеть аппроксимирует целевую функцию f^* [10]. Правильно обученная ИНС может лучше соответствовать набору данных, а также делает прогноз, учитывая не очевидные зависимости от данных.

Глубокие нейронные сети могут быть реализованы по-разному в зависимости от конкретных практических задач. Например, свёрточные нейронные сети специализируются на компьютерном зрении, а рекуррентные нейронные сети лучше подходят для обработки естественного языка.

Функцию отображения $f(x)$ можно рассматривать как цепочку из многих связанных функций в виде $f(x) = f^{(n)}(f^{(n-1)}(\dots f^{(2)}(f^{(1)}(x))))$. n связанных функций соответствуют глубине ИНС. Функция стоимости определяется на основе сравнения вывода y из $f(x)$ с целевым значением t из тренировочного набора данных. Цель обучения нейронной сети в том, чтобы приблизить функцию $f(x)$ к такой, чтобы минимизировать функцию стоимости. Минимизация функции стоимости является задачей оптимизации, и для этого часто используют алгоритм *градиентного спуска*. В *обратном распространении* (backpropagation), градиент используется для итеративного обновления нейронной сети во время обучения. Оптимизатор решает, какие параметры должны быть обновлены, а *скорость обучения* (learning rate) задает размер шага, на который параметр обновляется на каждой итерации.

1.5. Обучение с подкреплением

Хотя алгоритмы RL эффективно решают различные задачи, им не хватает масштабируемости и размерности.

С ростом глубоких нейронных сетей в последние годы, RL начинает использовать их функции приближения и представления свойств [12]. Это помогает преодолеть недостатки алгоритмов RL.

Это устраняет необходимость описывать свойства вручную, позволяя обучать модели, способные непосредственно выводить оптимальные действия, на основе необработанного и высокоразмерного ввода с сенсоров. Это проиллюстрировано на рис.1.5.

Таким образом, использование ИНС в обучении подкреплением, создает новую область – глубокое обучение с подкреплением (Deep Reinforcement Learning, DRL).

1.5.1. Алгоритмы глубокого обучения с подкреплением

Здесь нужно рассказать о Марковском процессе принятия решений (Markov Decision Process, MDP).

1.5.1.1. Марковский процесс принятия решений

Свойство Маркова означает, что следующее состояние зависит только от текущего состояния, тем самым, при принятии решения, мы можем игнорировать все прошлые состояния и учитывать только текущее.

Процесс *обучения с подкреплением* является формой МППР. Он состоит из нескольких элементов:

- набор состояний окружающей среды S ;
- набор действий A ;
- динамика перехода $T(s_{t+1}|s_t, a_t)$, которая описывает распределение новых состояний s_{t+1} , в которые может попасть агент, совершив действие a_t в состоянии s_t ;
- функция награды $R(s_t; a_t; s_{t+1})$
- дисконт фактор $\gamma \in [0, 1]$ для экспоненциального снижения будущих наград.

Политика π сопоставляет состояния вероятностям распределения действий:
 $\pi : S \rightarrow p(A = a|S)$

Эпизод — это предопределенный период времени, когда среда, начиная со случайного состояния, порождает серию переходов. Переходы в эпизоде можно рассматривать как траекторию политики.

Сумма наград, собранных в траектории политики $R = \sum_{t=0}^{T-1} \gamma^t r_{t+1}$

Цель обучения с подкреплением состоит в том, чтобы изучить оптимальную политику π^* , которая максимизирует ожидаемую награду из всех состояний, $\pi^* = \operatorname{argmax}_x \mathbb{E}[R|\pi]$ [20]

1.5.2. Подход, основанный на функции состояния (Value Based подход)

Этот подход состоит в том, чтобы оптимизировать значение функции $V(s)$.

value-функция — это функция, которая сообщает нам максимальное ожидаемое будущее вознаграждение, которое агент получит в каждом состоянии.

Значение каждого состояния — это общая сумма вознаграждений, на которую агент может рассчитывать в будущем, начиная с этого состояния.

$$V^\pi(s) = \mathbb{E}^\pi[R_{t+1} + \gamma R_t + 2 + \gamma^2 R_t + 3 + \dots | S_t = s] \quad (1.1)$$

Q-обучение Функция состояния-значения (state-value function) $V^\pi(s) = \mathbb{E}[R|s, \pi]$ — ожидаемая награда с данными состоянием s и политикой π . В то же время, в RL такой переход T не всегда возможен, поэтому, обычно используется другая функция состояния-значения или функция качества $Q^\pi(s, a) = \mathbb{E}[R|s, a, \pi]$. Оптимальная политика получается через выбор на каждом шаге действия, которое максимизирует Q-функцию [26]. Q-обучение — это алгоритм без политики (off-policy), так как он жадно выбирает действие исходя из текущего состояния, вместо того, чтобы следовать политике.

Для рекурсивного изучения Q^π , применяется *уравнение Беллмана*:

$$Q^\pi(s_t, a_t) = \mathbb{E}_{s_{t+1}}[r_{t+1} + \gamma Q^\pi(s_{t+1}, \pi(s_{t+1}))] \quad (1.2)$$

Традиционное Q-обучение способно сформулировать оптимальную политику обучением состояние-действие-значение функции. Однако, оно предназначено для дискретных пространств действий с малым количеством измерений и не способно решать более-менее сложные задачи.

Глубокая Q-сеть (DQN) Глубокая Q-сеть (DQN) — вариант Q-обучения, который использует глубокую сверточную нейронную сеть для вычисления Q-функции. Является прорывом в обучении с подкреплением. [4]

DQN была применена в игре Atari и достигла производительности, сопоставимой с человеческим уровнем. [14]

Для решения проблемы нестабильности и расхождения нелинейных функций аппроксиматоров, таких как нейронные сети, был применен метод *Experience Replay* [17]. Идея Experience Replay в том, чтобы равномерно рандомизировать предыдущие переходы при обучении модели, которое нарушает корреляцию последовательности наблюдений. Опыт агента $e_t = (s_t, a_t, r_t, s_{t+1})$ на каждом шаге t сохраняется в буфер $D_t = e_1, \dots, e_t$. Во время тренировки модели случайным образом извлекается небольшая часть опыта $(s; a; r; s') \sim U(D)$.

Двойная глубокая Q-сеть (Double DQN, DDQN) Это ещё один вариант DQN, в котором используется две Q-сети.

Текущая Q-сеть $Q(s, a; \theta_i)$ обновляется итеративно во время обучения, а целевая Q-сеть $Q(s', a'; \theta_i^-)$ используется для получения целевого Q-значения и обновляется только периодически. Целевая Q-сеть уменьшает смещение, вызванное неточностями Q-сети в начале обучения.

На каждой итерации i Q-сеть обновляется на ошибку темпоральной разницы (temporal difference, TD):

$$L_i(\theta_i) = \mathbb{E}^\pi [R_{t+1} + \gamma R_t + 2 + \gamma^2 R_t + 3 + \dots | S_t = s] \quad (1.3)$$

DQN подошел к проблеме низкоразмерных входных данных наблюдения с применением глубоких нейронных сетей для извлечения свойств из высокомерного необработанного сенсорного сигнала, такого как пиксели изображения в играх. Тем не менее, он все еще ограничен его дискретным и низкоразмерным пространством действия.

1.5.3. Линия поведения (Policy Based)

Вместо получения оптимальной политики путем поддержания Q-функции, алгоритм напрямую ищет оптимальную политику путем максимизации ожидаемого значения $\mathbb{E}[R|\pi_\theta]$. Мы хотим итеративно подгонять параметр θ сети, так чтобы максимизировать $\mathbb{E}[R|\pi_\theta]$.

Оптимизация, основанная на градиенте, используется чаще, так как это более эффективно, при работе с большими сетями со множеством параметров. [8]

1.5.3.1. Policy Gradients

(TODO: перевод?)

Нейронная сеть, которая представляет параметризованную политику обновляется при изучении сигналов в методах Policy Gradients. В RL без модели, для оценки градиента на примерах, сгенерированных в траектории политикой, используется правило *REINFORCE* или функция оценки. Предположим, что $f(x)$ – это функция оценки, где x - случайная величина для одного перехода. Градиент политики может быть рассчитан с использованием отношения правдоподобия:

$$(1.4)$$

Теперь рассмотрим траекторию τ с переходами (a_t, s_t, r_t, s_{t+1}) в соответствии с политикой, тогда градиент политики это:

TODO: ???

Недостатком policy gradient является низкая скорость работы — требуется большое количество вычислений для подсчета награды. Так же policy gradient может застрять в локальном оптимуме, не найдя глобального.

1.5.3.2. Актор-критик

Так как value-функция может предоставить обучающие сигналы для прямого оптимального поиска политики, естественным было объединить два подхода.

В DRL две нейронные сети, представляющие актора и критика соответственно, используются для приближения функции, где актор (политика) учится по Q-значениям, оцененным критиком (value-функция) [8]. рис.1.6 показывает, как актор и критик сети взаимодействуют с окружающей средой.

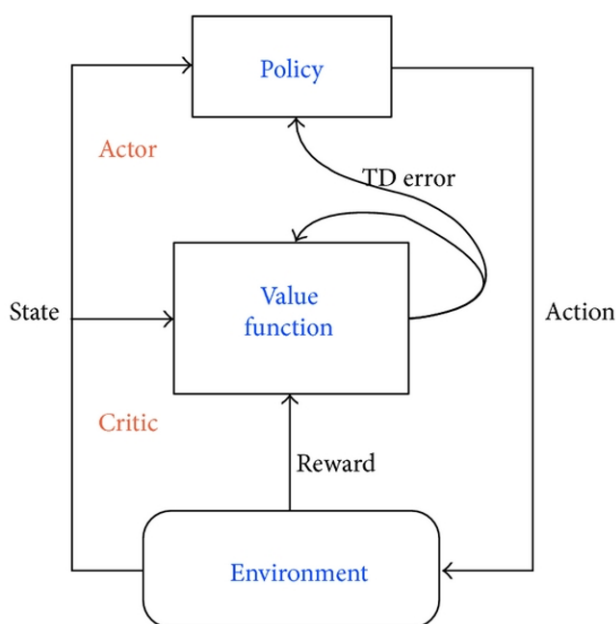


Рис.1.6. Актор получает состояние из окружающей среды и реагирует на действие, критик получает состояние и награду и рассчитывает TD ошибку для обновления себя и актора.

Основано на [8]

1.5.3.3. Глубокий детерминированный градиент политики (DDPG)

DDPG - актор-критик алгоритм без модели и без политики [6]. Он расширяет DPG использованием глубоких нейронных сетей. Так же, он использует хорошо зарекомендовавший себя приём DQN с текущей и целевой Q-сетями (сети критиков)

и experience replay, чтобы стабилизировать обучение. И, наконец, он использует детерминистическую политику (сеть акторов) вместо политики стохастического поведения. В отличие от стохастической политики, которая определяет вероятность распределения через действия, в данном состоянии, детерминированность в DDPG подразумевает, что конкретное действие аппроксимируется в данном состоянии. Соответственно, конкретное состояние для следующего шага, так же детерминировано. Следовательно, вместо использования рекурсивного программирования, как в уравнении *Беллмана*, используется детерминированная политика $\mu : S \leftarrow A$ [6]

$$Q^\mu(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1}} \mathbb{E}[r(s_t, a_t) + \gamma Q^\mu(s_{t+1}, \mu(s_{t+1}))] \quad (1.5)$$

что очень похоже на Q-обучение – алгоритм с жадной политикой. Сеть критика обновляется по функции потерь:

$$L(\theta^Q) = \mathbb{E}[(Q(s_t, a_t | \theta^Q) - y_t)^2] \quad (1.6)$$

где

$$y_t = r(s_t, a_t) + \gamma Q(s_{t+1}, \mu(s_{t+1})) \quad (1.7)$$

Сеть актора обновляется функцией потерь:

$$\nabla_{\theta^\mu} J \approx \mathbb{E}[\nabla_{\theta^\mu} Q(s, a | \theta^Q) |_{s=s_t, a=\mu(s_t | \theta^\mu)}] = \mathbb{E}[\nabla_a Q(s, a | \theta^Q) |_{s=s_t, a=\mu(s_t)} \nabla_{\theta^\mu} \mu(s | \theta^\mu) |_{s=s_t}] \quad (1.8)$$

Как и в DQN, чтобы избежать расхождения, DDPG применяет мягкое обновление для целевых сетей критика и актора. Они обновляются только раз в указанное количество шагов.

ЗАКЛЮЧЕНИЕ

Заключение (2 – 5 страниц) содержит выводы по теме работы, конкретные предложения и рекомендации по исследуемым вопросам. Количество общих выводов должно вытекать из количества задач, сформулированных во введении выпускной квалификационной работы.

Предложения и рекомендации должны быть органически увязаны с выводами и направлены на улучшение функционирования исследуемого объекта. При разработке предложений и рекомендаций обращается внимание на их обоснованность, реальность и практическую приемлемость.

Заключение не должно содержать новой информации, положений, выводов и т. д., которые до этого не рассматривались в выпускной квалификационной работе. Рекомендуются писать заключение в виде тезисов.

Последним абзацем в заключении можно выразить благодарность всем людям, которые помогали автору в написании ВКР.

СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ

- ML** Машинное обучение (Machine Learning).
- DL** Глубокое обучение (Deep Learning).
- RL** Обучение с подкреплением (Reinforcement Learning).
- DPG** Детерминированный градиент политики (Deterministic Policy Gradient).
- DDPG** Глубокий детерминированный градиент политики (Deep Deterministic Policy Gradient).
- ИНС** Искусственная нейронная сеть (Artificial Neural Network).
- МППР** Марковский процесс принятия решений (Markov Decision Process, MDP).

СЛОВАРЬ ТЕРМИНОВ

TeX — язык вёрстки текста и издательская система, разработанные Дональдом Кнутом.

LaTeX — язык вёрстки текста и издательская система, разработанные Лэсли Лампортом как надстройка над TeX.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. AlphaGo. — URL: <https://en.wikipedia.org/wiki/AlphaGo>.
2. An Introduction to Statistical Learning: with Applications in R / G. James [и др.]. — Springer New York, 2014. — (Сер.: Springer Texts in Statistics). — URL: <https://books.google.se/books?id=at1bmAEACAAJ>.
3. *Bengio Y., Courville A., Vincent P.* Representation Learning: A Review and New Perspectives. — 2012. — arXiv: 1206.5538 [cs.LG].
4. *Bertsekas D., Tsitsiklis J. a.* Neuro-dynamic Programming. — Athena Scientific, 1996. — (Сер.: Anthropological Field Studies). — URL: <https://books.google.ru/books?id=WxCCQgAACAAJ>.
5. *Bishop C.* Pattern Recognition and Machine Learning. — Springer, 2006. — 738 с. — URL: <http://research.microsoft.com/en-us/um/people/cmbishop/prml>.
6. Continuous control with deep reinforcement learning / T. P. Lillicrap [и др.]. — 2015. — arXiv: 1509.02971 [cs.LG].
7. *Crevier D.* AI: The Tumultuous History of the Search for Artificial Intelligence. — 1993. — с. 1—386. — URL: https://www.researchgate.net/publication/233820788_AI_The_Tumultuous_History_of_the_Search_for_Artificial_Intelligence.
8. Deep Reinforcement Learning: A Brief Survey / K. Arulkumaran [и др.] // IEEE Signal Processing Magazine. — 2017. — т. 34, № 6. — с. 26—38. — DOI 10.1109/msp.2017.2743240. — URL: <http://dx.doi.org/10.1109/MSP.2017.2743240>.
9. *Edwards C.* Growing Pains for Deep Learning // Commun. ACM. — New York, NY, USA, 2015. — т. 58, № 7. — с. 14—16. — DOI 10.1145/2771283. — URL: <https://doi.org/10.1145/2771283>.
10. *Goodfellow I., Bengio Y., Courville A.* Deep Learning. — MIT Press, 2016. — (<http://www.deeplearningbook.org>).
11. *Hastie T., Tibshirani R., Friedman J.* The Elements of Statistical Learning: Data Mining, Inference, and Prediction. — Springer, 2001. — (Сер.: Springer series in statistics). — URL: <https://books.google.ru/books?id=VRzITwgNV2UC>.
12. *Hornik K.* Approximation capabilities of multilayer feedforward networks // Neural Networks. — 1991. — т. 4, № 2. — с. 251—257. — DOI [https://doi.org/10.1016/0893-6080\(91\)90009-T](https://doi.org/10.1016/0893-6080(91)90009-T). — URL: <http://www.sciencedirect.com/science/article/pii/089360809190009T>.

13. *Huang S. H., Hong-Chao Zhang.* Artificial neural networks in manufacturing: concepts, applications, and perspectives // IEEE Transactions on Components, Packaging, and Manufacturing Technology: Part A. — 1994. — т. 17, № 2. — с. 212—228.
14. Human-level control through deep reinforcement learning / V. Mnih [и др.] // Nature. — 2015. — т. 518. — с. 529—33. — DOI 10.1038/nature14236.
15. *Kanter J. M., Veeramachaneni K.* Deep feature synthesis: Towards automating data science endeavors // 2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA). — 2015. — с. 1—10.
16. *Khajanchi A.* Artificial Neural Networks : The next intelligence //. — 2003. — URL: <https://www.semanticscholar.org/paper/Artificial-Neural-Networks-%3A-The-next-intelligence-Khajanchi/312a65e33ebba4cbff154a79f57e2a0ff386e6f6>.
17. *Mitchell T.* Machine Learning. — McGraw-Hill, 1997. — (Сер.: McGraw-Hill International Editions). — URL: <https://books.google.ru/books?id=EoYBngEACAAJ>.
18. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments / R. Lowe [и др.]. — 2017. — arXiv: 1706.02275 [cs.LG].
19. OpenAI Gym. — URL: <https://github.com/openai/gym>.
20. *Otterlo M. van, Wiering M. A.* Reinforcement Learning and Markov Decision Processes // Reinforcement Learning. — 2012.
21. *Russell S., Norvig P.* Artificial Intelligence: A Modern Approach. — 3rd. — USA: Prentice Hall Press, 2009. — с. 1—5.
22. *Samuel A. L.* Some Studies in Machine Learning Using the Game of Checkers // IBM Journal of Research and Development. — 1959. — т. 3, № 3. — с. 210—229. — URL: <http://dx.doi.org/10.1147/rd.33.0210>.
23. *Sultan K., Ali H., Zhang Z.* Big Data Perspective and Challenges in Next Generation Networks // Future Internet. — 2018. — т. 10, № 7. — с. 56. — DOI 10.3390/fi10070056. — URL: <http://dx.doi.org/10.3390/fi10070056>.
24. *Sutton R. S., Barto A. G.* Reinforcement Learning: An Introduction. — 2-е изд. — The MIT Press, 2008. — с. 2. — URL: <https://books.google.ru/books?id=VRzITwgNV2UC>.
25. *Sutton R. S., Barto A. G.* Reinforcement Learning: An Introduction. — 2-е изд. — The MIT Press, 2008. — с. 50. — URL: <https://books.google.ru/books?id=VRzITwgNV2UC>.
26. *Sutton R. S., Barto A. G.* Reinforcement Learning: An Introduction. — 2-е изд. — The MIT Press, 2008. — с. 107—108. — URL: <https://books.google.ru/books?id=VRzITwgNV2UC>.

27. The Arcade Learning Environment: An Evaluation Platform for General Agents / M. G. Bellemare [и др.] // Journal of Artificial Intelligence Research. — 2013. — т. 47. — с. 253—279. — DOI 10.1613/jair.3912.

28. Watson (computer). — URL: [https://en.wikipedia.org/wiki/Watson_\(computer\)](https://en.wikipedia.org/wiki/Watson_(computer)).