МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Факультет компьютерных технологий и прикладной математики

УТВЕРЖДАЮ:
Проректор по учебной работе,
качеству образования – первый
проректор
_____Хагуров Т.А.
*подпись*
« 29 » августа 2025 г.

# РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

Б1. В.14 Интеллектуальные методы оптимизации

Направление подготовки 02.03.02 Фундаментальная информатика и информационные технологии

Профиль Современные методы машинного обучения и компьютерного зрения

Форма обучения очная

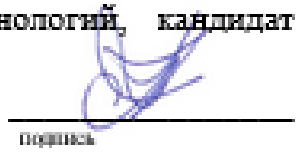
Квалификация бакалавр

Краснодар 2025

Рабочая программа дисциплины «Интеллектуальные методы оптимизации» составлена в соответствии с федеральным государственным образовательным стандартом высшего образования (ФГОС ВО) по направлению подготовки 02.03.02 Фундаментальные информатика и информационные технологии

Программу составила:

Е.Е. Полупанова, доцент кафедры вычислительных технологий, кандидат технических наук, доцент



подпись

Рабочая программа дисциплины «Интеллектуальные методы оптимизации» утверждена на заседании центра искусственного интеллекта протокол № 1 «28» августа 2025 г.

Руководитель центра ИИ Коваленко А.В.

фамилия, инициалы

подпись

Утверждена на заседании учебно-методической комиссии факультета компьютерных технологий и прикладной математики протокол №1 «28» августа 2025 г.

Председатель УМК факультета Коваленко А.В.

фамилия, инициалы

подпись

Рецензенты:

Мостовой Евгений Викторович, генеральный директор ООО «Портал-Юг», e-mail: mostovoy@portal-yug.ru

Луценко Евгений Вениаминович, доктор экономических наук, кандидат технических наук, профессор кафедры компьютерных технологий и систем Федерального государственного бюджетное образовательное учреждение высшего образования «Кубанский государственный аграрный университет имени И.Т. Трубилина», e-mail: prof.lutsenko@gmail.com

# 1. Цели и задачи освоения дисциплины (модуля)

## 1.1 Цель освоения дисциплины

Целью преподавания и изучения дисциплины «Интеллектуальные методы оптимизации» является формирование у бакалавров знаний, умений и навыков в области классических методов решения оптимизационных задач, основанных на использовании дифференциального исчисления для нахождения точек экстремумов функции, методов одномерной минимизации, методов условной и безусловной оптимизации, алгоритмов случайного поиска, биоинспирированных алгоритмы, методов метаоптимизации, многоцелевой оптимизации, а также нейроэволюционных методов.

## 1.2 **Задачи дисциплины**

Основная задача освоения дисциплины: анализ и построение эффективных интеллектуальных алгоритмов для решения типовых задач поисковой оптимизации с применением современных языков программирования и инструментальных сред.

## 1.3 **Место дисциплины (модуля) образовательной программе**

Дисциплина «Интеллектуальные методы оптимизации» относится к вариативной части, формируемой участниками образовательных отношений, Блока 1 учебного плана. Для изучения дисциплины необходимо знание дисциплин «Дифференциальные уравнения», «Математический анализ», «Программирование», «Алгебра и аналитическая геометрия». Знания, получаемые при изучении дисциплины, используются при изучении таких дисциплин учебного плана бакалавра как «Нейросетевые технологии», «Генеративные нейронные сети».

## 1.4 Профессиональные роли в структуре образовательной программы

Роль 1: Data Engineer (Инженер по данным)
Задачи:

- Проектирование и построение ETL-процессов
- Создание и оптимизация хранилищ данных
- Обеспечение качества и доступности данных
- Настройка инфраструктуры для обработки больших данных
- Интеграция разрозненных источников данных
- Работа с данными в области природопользования, медицины, связи и телекоммуникаций

Роль 2: ML Engineer (Инженер МО)
Задачи:

- Реализация ML-моделей в продуктивных системах
- Оптимизация производительности и масштабирование моделей
- Разработка ML-пайплайнов и автоматизация процессов
- Мониторинг качества моделей в продуктиве
- Интеграция ML-решений с бизнес-приложениями

Роль 3: MLOps (Специалист по эксплуатации ИИ)
Задачи:

- Автоматизация процессов обучения и развертывания моделей
- Мониторинг производительности ML-систем
- Управление версиями моделей и данных
- Обеспечение CI/CD для ML-проектов

- Оптимизация вычислительных ресурсов

**1.4 Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с планируемыми результатами освоения образовательной программы.**

Изучение данной учебной дисциплины направлено на формирование у обучающихся следующих компетенций:

| Код и наименование индикатора* | Результаты обучения по дисциплине *(знает, умеет, владеет (навыки и/или опыт деятельности))* |
|---|---|
| **MF-3. Способен применять современные методы оптимизации для обучения моделей машинного обучения, настройки гиперпараметров и решения задач искусственного интеллекта** | |
| MF-3.1. Применяет методы оптимизации для разработки и исследования обучающих алгоритмов. | Знает основные теоретические аспекты градиентных алгоритмов, их классификацию и области применения.<br>Умеет применять типовые градиентные алгоритмы для решения типовых задач оптимизации и обучения.<br>Умеет применять типовые метаэвристические алгоритмы для решения типовых задач оптимизации, понимает основные теоретические аспекты метаэвристических алгоритмов, их классификацию и области применения.<br>Умеет анализировать сходимость и эффективность алгоритмов, выбирает и обосновывает применение наиболее подходящих методов в зависимости от характеристик данных и модели.<br>Выбирает наиболее подходящие к поставленной задаче алгоритмы метаэвристической оптимизации и сравнивает с аналогами на основе постановки вычислительного эксперимента, знает тренды в области, способы и примеры применения. |
| MF-3.2. Применяет методы оптимизации для настройки гиперпараметров моделей машинного обучения, включая использование методов поиска (grid search, random search) и байесовской оптимизации. | Знает и использует стандартные методы поиска гиперпараметров, такие как grid search и random search, для настройки моделей машинного обучения в стандартных задачах.<br>Умеет настраивать гиперпараметры с использованием более сложных методов, таких как байесовская оптимизация, для улучшения производительности моделей и минимизации времени обучения. |
| **ML-3. Способен применять классические алгоритмы машинного обучения с пониманием их математических основ и областей применения** | |
| ML-3.1. Обосновывает способы и варианты применения классических методов и моделей машинного обучения в задачах ИИ, включая их математическое (алгоритмическое) преобразование и адаптацию к специфике задачи | Знает базовые модели МО с учителем (метод ближайших соседей, деревья принятия решений, линейная регрессия, логистическая регрессия, метод опорных векторов) с пониманием их математической сущности<br>Умеет обосновывать выбор конкретных алгоритмов и их параметров в зависимости от задачи и данных<br>Владеет навыками разрабаотки и адаптации собственных алгоритмических решений на основе классических методов, обоснования математически сложных решений. |
| **ML-6. Способен применять алгоритмы обучения с подкреплением** | |
| ML-6.1. Обосновывает способы и варианты применения алгоритмов обучения с подкреплением в задачах ИИ, включая их преобразование и адаптацию к специфике задачи | Знает основные принципы обучения с подкреплением (агент, среда, награда) и обосновывает выбор простейших алгоритмов (Q-Learning, SARSA) для решения типовых задач<br>Умеет разрабатывать адаптивного агента; проводить аппроксимацию функции ценности агента, в том числе с помощью стратегии; применять TD-методы и методы Монте-Карло для обучения агента; задавать цель агента |

| Код и наименование индикатора* | Результаты обучения по дисциплине *(знает, умеет, владеет (навыки и/или опыт деятельности))* |
|---|---|
| | с помощью полного вознаграждения, вознаграждения с обесценением, лямбда-дохода |
| **О-3. Способен применять и (или) разрабатывать интеллектуальные методы оптимизации** | |
| О-3.2. Обосновывает способы и варианты применения интеллектуальных методов в задачах оптимизации | Умеет обосновывать методы оптимизации на основе статических данных о параметрах и характеристиках продуктов компании и статических алгоритмов<br>Умеет обосновывать методы оптимизации на основе анализа динамики функционирования объектов оптимизации и использования статических алгоритмов |
| О-3.3. Эффективно применяет интеллектуальные методы оптимизации для обеспечения достижимости функциональных характеристик продуктов компании | Умеет применять типовые решения для оптимизации функционирования наблюдаемой системы<br>Умеет выбирать методы оптимизации с учетом специфики наблюдаемой системы, определяемой требованиями по назначению |

# 2. Структура и содержание дисциплины

## 2.1 Распределение трудоёмкости дисциплины по видам работ

Общая трудоёмкость дисциплины составляет 4 зач.ед. (144 часа), их распределение по видам работ представлено в таблице.

| Вид учебной работы | | Всего часов | Очная форма обучения |
|---|---|---|---|
| | | | 5 семестр (часы) |
| **Контактная работа, в том числе:** | | | |
| **Аудиторные занятия (всего):** | | **72,3** | **68,3** |
| Занятия лекционного типа | | 34 | 32 |
| Лабораторные занятия | | 34 | 32 |
| Занятия семинарского типа (семинары, практические занятия) | | – | – |
| **Иная контактная работа:** | | | |
| Контроль самостоятельной работы (КСР) | | 4 | 4 |
| Промежуточная аттестация (ИКР) | | 0,3 | 0,3 |
| **Самостоятельная работа, в том числе:** | | **36** | **36** |
| Курсовая работа/проект (КР/КП) (подготовка) | | – | – |
| Проработка учебного (теоретического) материала | | 36 | 36 |
| Выполнение индивидуальных заданий (подготовка сообщений, презентаций) | | – | – |
| Реферат | | – | – |
| Подготовка к текущему контролю | | – | – |
| **Контроль:** | | **экзамен** | **экзамен** |
| Подготовка к экзамену | | 35,7 | 35,7 |
| **Общая трудоёмкость** | **час.** | **144** | **144** |
| | **в том числе контактная работа** | **72,3** | **68,3** |
| | **зач. ед.** | **4** | **4** |

## 2.2 Структура дисциплины

Распределение видов учебной работы и их трудоемкости по разделам дисциплины.
Разделы дисциплины, изучаемые в _5_ семестре (очная форма).

| № | Наименование разделов | Количество часов | | | | |
|---|---|---|---|---|---|---|
| | | Всего | Аудиторная работа | | | Внеаудиторная работа |
| | | | Л | ПЗ | ЛР | СРС |

| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|
| 1 | Постановка и классификация алгоритмов решения задачи поисковой оптимизации | 6 | 2 | – | 2 | 2 |
| 2 | Локальная оптимизация. Алгоритмы градиентного спуска. | 14 | 4 | – | 6 | 4 |
| 3 | Глобальная оптимизация. Алгоритмы случайного поиска. | 12 | 4 | – | 4 | 6 |
| 4 | Биоинспирированные алгоритмы. | 18 | 6 | – | 6 | 6 |
| 5 | Метаоптимизация. | 18 | 6 | – | 6 | 6 |
| 6 | Алгоритмы многоцелевой оптимизации. | 18 | 6 | – | 6 | 6 |
| 7 | Нейроэволюция. | 18 | 6 | – | 6 | 6 |
| | *ИТОГО по разделам дисциплины* | 104 | 34 | – | 34 | 36 |
| 8 | Подготовка к текущему контролю | 35,7 | | | | |
| 9 | Промежуточная аттестация (ИКР) | 0,3 | | | | |
| 10 | Контроль самостоятельной работы (КСР) | 4 | | | | |
| | Общая трудоемкость по дисциплине | **144** | **34** | – | **34** | **36** |

Примечание: Л – лекционные занятия, ПЗ – практические занятия / семинары, ЛР – лабораторные занятия, СРС – самостоятельная работа студента

**2.3 Содержание разделов дисциплины**

**2.3.1 Занятия лекционного типа**

| № раз-дела | Наименование раздела | Содержание раздела | Форма текущего контроля |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| 1 | Постановка и классификация алгоритмов решения задачи поисковой оптимизации. | Постановка задачи поисковой оптимизации. Классификация задач оптимизации. Классификация алгоритмов оптимизации. | ЛР |
| 2 | Локальная оптимизация. | Локальная безусловная оптимизация. Одношаговые алгоритмы. Многошаговые алгоритмы. Алгоритмы градиентного спуска: метод градиентного спуска с постоянным шагом, метод наискорейшего спуска, метод покоординатного спуска. Локальная условная оптимизация. | ЛР |
| 3 | Глобальная оптимизация. Алгоритмы случайного поиска. | Алгоритмы случайного поиска. Алгоритм Монте-Карло. Алгоритм имитации отжига. Двухфазные алгоритмы случайного поиска. Алгоритм мультистарта. Жадный адаптивный алгоритм случайного поиска. Поиск с запретами. | ЛР |

| | | | |
|---|---|---|---|
| 4 | Биоинспирированные алгоритмы. | Эволюционные алгоритмы. Общая схема эволюционных алгоритмов: основные термины и определения. Способы кодирования особей в эволюционных алгоритмах. Генетические алгоритмы. Операторы отбора и селекции. Операторы кроссинговера и мутации.<br>Алгоритма оптимизации роем частиц. Оптимизация пчелиным роем. Пчелиный алгоритм. Искусственные иммунные системы. Канонический алгоритм бактериальной оптимизации. | ЛР |
| 5 | Метаоптимизация. | Классификация методов метаоптимизации. Статистическая параметрическая метаоптимизация. Динамическая параметрическая метаоптимизация. Неадаптивная динамическая метаоптимизация. Адаптивная динамическая метаоптимизация. Структурная метаоптимизация. Структурно-параметрическая метаоптимизация. | ЛР |
| 6 | Алгоритмы многоцелевой оптимизации | Задача многоцелевой оптимизации (МЦО-задача) и алгоритмы ее решения. Алгоритмы Парето-аппроксимации. | ЛР |
| 7 | Нейроэволюция | Эволюционные алгоритмы и нейроэволюционные методы. Генетические операторы. Схемы кодирования генома. Коэволюция. Алгоритм NEAT: схема кодирования, структурные мутации, видообразование. Использование NEAT для оптимизации решения задачи XOR. NEAT на основе гиперкуба. Зрительное различение с NEAT на основе гиперкуба. | ЛР |

**2.3.2 Занятия семинарского типа (практические / семинарские занятия/ лабораторные работы)**

| № | Наименование раздела (темы) | Наименование лабораторных работ | Форма текущего контроля |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| 1. | Постановка и классификация алгоритмов решения задачи поисковой оптимизации | Разработка базовой архитектуры программного комплекса для решения задачи оптимизации | ЛР |
| 2. | Локальная оптимизация. | Разработка алгоритма градиентного спуска для оптимизации функции Розенброка | ЛР |
| 3. | Глобальная оптимизация. Алгоритмы случайного поиска. | Разработка комбинированного алгоритма глобальной оптимизации для поиска минимума функции Химмельблау | ЛР |
| 4. | Биоинспирированные алгоритмы | Эволюционный алгоритм глобальной условной оптимизации целевой и фитнес-функции | ЛР |

| | | | |
|---|---|---|---|
| 5. | | Разработка генетического алгоритма оптимизации с вещественным кодированием особей | |
| 6. | | Решение задачи глобальной безусловной оптимизации для функций Шекеля, Розенброка, Растригина с помощью алгоритма PSO | ЛР |
| 7. | | Миксимизация функций Розенброка, Химмельблау, Растригина B-алгоритмом. | ЛР |
| 8. | | Разработка алгоритма искусственной иммунной сети для задачи глобальной безусловной минимизации функций Розенброка. | ЛР |
| 9. | | Канонический алгоритм бактериальной оптимизации. | |
| 10. | Метаоптимизация | Разработка алгоритма динамической параметрической метаоптимизации. | ЛР |
| 11. | Алгоритмы многоцелевой оптимизации | Разработка непопуляционного алгоритма Парето-аппроксимации. | ЛР |
| 12. | | Разработка популяционного алгоритма Парето-аппроксимации. | ЛР |
| 13. | Нейроэволюция | Использование алгоритма NEAT на основе гиперкуба для решения задачи зрительного различения | ЛР |

Защита лабораторной работы (ЛР), выполнение курсового проекта (КП), курсовой работы (КР), расчетно-графического задания (РГЗ), написание реферата (Р), эссе (Э), коллоквиум (К), тестирование (Т) и т.д.

### 2.3.3 Примерная тематика курсовых работ (проектов)

Учебным планом не предусмотрены.

## 2.4 Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине

| № | Вид СРС | Перечень учебно-методического обеспечения дисциплины по выполнению самостоятельной работы |
|---|---|---|
| 1 | 2 | 3 |
| 1 | Проработка и повторение лекционного материала, материала учебной и научной литетаруры, подготовка к семинарским занятиям | Методические указания по выполнению самостоятельной работы, утвержденные на заседании кафедры вычислительных технологий ФКТиПМ ФГБОУ ВО «КубГУ», протокол №7 от 07.05.2025 |

Учебно-методические материалы для самостоятельной работы обучающихся и числа инвалидов и лиц с ограниченными возможностями здоровья (ОВЗ) предоставляются в формах, адаптированных к ограничениям их здоровья и восприятия информации:
Для лиц с нарушениями зрения:
– в печатной форме увеличенным шрифтом,
– в форме электронного документа.
Для лиц с нарушениями слуха:

– в печатной форме,
– в форме электронного документа.
Для лиц с нарушениями опорно-двигательного аппарата:
– в печатной форме,
– в форме электронного документа.
Данный перечень может быть конкретизирован в зависимости от контингента обучающихся.

**3. Образовательные технологии**

Программа дисциплины предусматривает использование в учебном процессе следующих образовательные технологии: чтение лекций с использованием мультимедийных технологий; метод малых групп, разбор практических задач и кейсов.

При обучении используются следующие образовательные технологии:

- Технология коммуникативного обучения – направлена на формирование коммуникативной компетентности студентов, которая является базовой, необходимой для адаптации к современным условиям межкультурной коммуникации.
- Технология разноуровневого (дифференцированного) обучения – предполагает осуществление познавательной деятельности студентов с учётом их индивидуальных способностей, возможностей и интересов, поощряя их реализовывать свой творческий потенциал. Создание и использование диагностических тестов является неотъемлемой частью данной технологии.
- Технология модульного обучения – предусматривает деление содержания дисциплины на достаточно автономные разделы (модули), интегрированные в общий курс.
- Информационно-коммуникационные технологии (ИКТ) - расширяют рамки образовательного процесса, повышая его практическую направленность, способствуют интенсификации самостоятельной работы учащихся и повышению познавательной активности.

В рамках ИКТ выделяются 2 вида технологий:

- Технология использования компьютерных программ – позволяет эффективно дополнить процесс обучения языку на всех уровнях.
- Интернет-технологии – предоставляют широкие возможности для поиска информации, разработки научных проектов, ведения научных исследований.
- Технология индивидуализации обучения – помогает реализовывать личностно-ориентированный подход, учитывая индивидуальные особенности и потребности учащихся.
- Проектная технология – ориентирована на моделирование социального взаимодействия учащихся с целью решения задачи, которая определяется в рамках профессиональной подготовки, выделяя ту или иную предметную область.
- Технология обучения в сотрудничестве – реализует идею взаимного обучения, осуществляя как индивидуальную, так и коллективную ответственность за решение учебных задач.
- Игровая технология – позволяет развивать навыки рассмотрения ряда возможных способов решения проблем, активизируя мышление студентов и раскрывая личностный потенциал каждого учащегося.
- Технология развития критического мышления – способствует формированию разносторонней личности, способной критически относиться к информации, умению отбирать информацию для решения поставленной задачи.

Комплексное использование в учебном процессе всех вышеназванных технологий стимулируют личностную, интеллектуальную активность, развивают познавательные процессы, способствуют формированию компетенций, которыми должен обладать будущий специалист.

Основные виды интерактивных образовательных технологий включают в себя:

- работа в малых группах (команде) - совместная деятельность студентов в группе под руководством лидера, направленная на решение общей задачи путём творческого сложения результатов индивидуальной работы членов команды с делением полномочий и ответственности;
- проектная технология - индивидуальная или коллективная деятельность по отбору, распределению и систематизации материала по определенной теме, в результате которой составляется проект;
- анализ конкретных ситуаций - анализ реальных проблемных ситуаций, имевших место в соответствующей области профессиональной деятельности, и поиск вариантов лучших решений;
- развитие критического мышления – образовательная деятельность, направленная на развитие у студентов разумного, рефлексивного мышления, способного выдвинуть новые идеи и увидеть новые возможности.

Подход разбора конкретных задач и ситуаций широко используется как преподавателем, так и студентами во время лекций, лабораторных занятий и анализа результатов самостоятельной работы. Это обусловлено тем, что при исследовании и решении каждой конкретной задачи имеется, как правило, несколько методов, а это требует разбора и оценки целой совокупности конкретных ситуаций.

Темы, задания и вопросы для самостоятельной работы призваны сформировать навыки поиска информации, умения самостоятельно расширять и углублять знания, полученные в ходе лекционных и практических занятий.

Подход разбора конкретных ситуаций широко используется как преподавателем, так и студентами при проведении анализа результатов самостоятельной работы.

Для лиц с ограниченными возможностями здоровья предусмотрена организация консультаций с использованием электронной почты.

Для лиц с нарушениями зрения:
– в печатной форме увеличенным шрифтом,
– в форме электронного документа.

Для лиц с нарушениями слуха:
– в печатной форме,
– в форме электронного документа.

Для лиц с нарушениями опорно-двигательного аппарата:
– в печатной форме,
– в форме электронного документа.

Для лиц с ограниченными возможностями здоровья предусмотрена организация консультаций с использованием электронной почты.

Данный перечень может быть конкретизирован в зависимости от контингента обучающихся.

## 4. Оценочные средства для текущего контроля успеваемости и промежуточной аттестации

### 4.1 Оценочные средства для текущего контроля успеваемости и промежуточной аттестации

Оценочные средства предназначены для контроля и оценки образовательных достижений обучающихся, освоивших программу учебной дисциплины «Интеллектуальные методы оптимизации».

Оценочные средства включает контрольные материалы для проведения текущего контроля выполнения заданий, лабораторных работ, средств итоговой аттестации (экзамен в 5 семестре).

Оценка успеваемости осуществляется по результатам:
- выполнения лабораторных работ;
- ответов на теоретические вопросы при сдаче лабораторных работ;
- ответа на экзамене (для выявления знания и понимания теоретического материала дисциплины).

Оценочные средства для инвалидов и лиц с ограниченными возможностями здоровья выбираются с учетом их индивидуальных психофизических особенностей.
– при необходимости инвалидам и лицам с ограниченными возможностями здоровья предоставляется дополнительное время для подготовки ответа на экзамене;
– при проведении процедуры оценивания результатов обучения инвалидов и лиц с ограниченными возможностями здоровья предусматривается использование технических средств, необходимых им в связи с их индивидуальными особенностями;
– при необходимости для обучающихся с ограниченными возможностями здоровья и инвалидов процедура оценивания результатов обучения по дисциплине может проводиться в несколько этапов.
Процедура оценивания результатов обучения инвалидов и лиц с ограниченными возможностями здоровья по дисциплине (модулю) предусматривает предоставление информации в формах, адаптированных к ограничениям их здоровья и восприятия информации:
Для лиц с нарушениями зрения:
– в печатной форме увеличенным шрифтом,
– в форме электронного документа.
Для лиц с нарушениями слуха:
– в печатной форме,
– в форме электронного документа.
Для лиц с нарушениями опорно-двигательного аппарата:
– в печатной форме,
– в форме электронного документа.
Данный перечень может быть конкретизирован в зависимости от контингента обучающихся.

**Структура оценочных средств для текущей и промежуточной аттестации**

| № п/п | Код и наименование индикатора | Результаты обучения | Наименование оценочного средства | |
|---|---|---|---|---|
| | | | Текущий контроль | Промежу-точная аттестация |
| **<u>на базовом уровне</u>**: | | | | |
| 1 | MF-3.1. Применяет методы оптимизации для разработки и исследования обучающих алгоритмов. | Знает типовые градиентные алгоритмы для решения типовых задач оптимизации, понимает основные теоретические аспекты градиентных алгоритмов, их классификацию и области применения.<br>Умеет применять типовые метаэвристические алгоритмы для решения типовых задач оптимизации, понимает основные теоретические аспекты метаэвристических алгоритмов, их классификацию и области применения. | Опрос по теме, лабораторная работа | Вопросы на экзамен 1-28 |
| 2 | MF-3.2. Применяет методы оптимизации для настройки гиперпараметров моделей машинного обучения, включая использование методов поиска (grid search, random search) и байесовской оптимизации | Знает и использует стандартные методы поиска гиперпараметров, такие как grid search и random search, для настройки моделей машинного обучения в стандартных задачах. | Опрос по теме, лабораторная работа | Вопросы на экзамен 31-36 |

| | | | | |
|---|---|---|---|---|
| 3 | ML-3.1. Обосновывает способы и варианты применения классических методов и моделей машинного обучения в задачах ИИ, включая их математическое (алгоритмическое) преобразование и адаптацию к специфике задачи | Знает базовые модели МО с учителем с пониманием их математической сущности | Опрос по теме, лабораторная работа | Вопросы на экзамен 31-36 |
| 4 | ML-6.1. Обосновывает способы и варианты применения алгоритмов обучения с подкреплением в задачах ИИ, включая их преобразование и адаптацию к специфике задачи | Знает основные принципы обучения с подкреплением и обосновывает выбор простейших алгоритмов для решения типовых задач | Опрос по теме, лабораторная работа | Вопросы на экзамен 31-36 |
| 5 | О-3.2. Обосновывает способы и варианты применения интеллектуальных методов в задачах оптимизации | Умеет обосновывать методы оптимизации на основе статических данных о параметрах и характеристиках продуктов компании и статических алгоритмов | Опрос по теме, лабораторная работа | Вопросы на экзамен 14-21 |
| 6 | О-3.3. Эффективно применяет интеллектуальные методы оптимизации для обеспечения достижимости функциональных характеристик продуктов компании | Умеет применять типовые решения для оптимизации функционирования наблюдаемой системы | Опрос по теме, лабораторная работа | Вопросы на экзамен 29-30 |
| **<u>на продвинутом уровне:</u>** | | | | |
| 7 | MF-3.1. Применяет методы оптимизации для разработки и исследования обучающих алгоритмов. | Умеет анализировать сходимость и эффективность алгоритмов, выбирает и обосновывает применение наиболее подходящих методов в зависимости от характеристик данных и модели.<br>Выбирает наиболее подходящие к поставленной задаче алгоритмы метаэвристической оптимизации и сравнивает с аналогами на основе постановки вычислительного эксперимента, знает тренды в области, способы и примеры применения. | Опрос по теме, лабораторная работа | Вопросы на экзамен 1-28 |
| 8 | MF-3.2. Применяет методы оптимизации для настройки гиперпараметров моделей машинного обучения, включая использование методов поиска (grid search, random search) и байесовской оптимизации | Умеет настраивать гиперпараметры с использованием более сложных методов, таких как байесовская оптимизация, для улучшения производительности моделей и минимизации времени обучения. | Опрос по теме, лабораторная работа | Вопросы на экзамен 31-36 |
| 9 | ML-3.1. Обосновывает способы и варианты применения классических методов и моделей машинного обучения в задачах ИИ, включая их математическое (алгоритмическое) | Умеет обосновывать выбор конкретных алгоритмов и их параметров в зависимости от задачи и данных | Опрос по теме, лабораторная работа | Вопросы на экзамен 31-36 |

| | | | | |
|---|---|---|---|---|
| | преобразование и адаптацию к специфике задачи | | | |
| 10 | ML-6.1. Обосновывает способы и варианты применения алгоритмов обучения с подкреплением в задачах ИИ, включая их преобразование и адаптацию к специфике задачи | Умеет разрабатывать адаптивного агента; проводить аппроксимацию функции ценности агента, в том числе с помощью стратегии; применять TD-методы и методы Монте-Карло для обучения агента | Опрос по теме, лабораторная работа | Вопросы на экзамен 31-36 |
| 11 | О-3.2. Обосновывает способы и варианты применения интеллектуальных методов в задачах оптимизации | Умеет обосновывать методы оптимизации на основе анализа динамики функционирования объектов оптимизации и использования статических алгоритмов | Опрос по теме, лабораторная работа | Вопросы на экзамен 14-21 |
| 12 | О-3.3. Эффективно применяет интеллектуальные методы оптимизации для обеспечения достижимости функциональных характеристик продуктов компании | Умеет выбирать методы оптимизации с учетом специфики наблюдаемой системы, определяемой требованиями по назначению | Опрос по теме, лабораторная работа | Вопросы на экзамен 29-30 |
| **<u>на экспертном уровне:</u>** | | | | |
| 13 | ML-3.1. Обосновывает способы и варианты применения классических методов и моделей машинного обучения в задачах ИИ, включая их математическое (алгоритмическое) преобразование и адаптацию к специфике задачи | Владеет навыками разработки и адаптации собственных алгоритмических решений на основе классических методов, обоснования математически сложных решений. | Опрос по теме, лабораторная работа | Вопросы на экзамен 31-36 |

**Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций в процессе освоения образовательной программы**

***Лабораторная работа №13 «Использование HyperNEAT для решения задачи зрительного различения»***

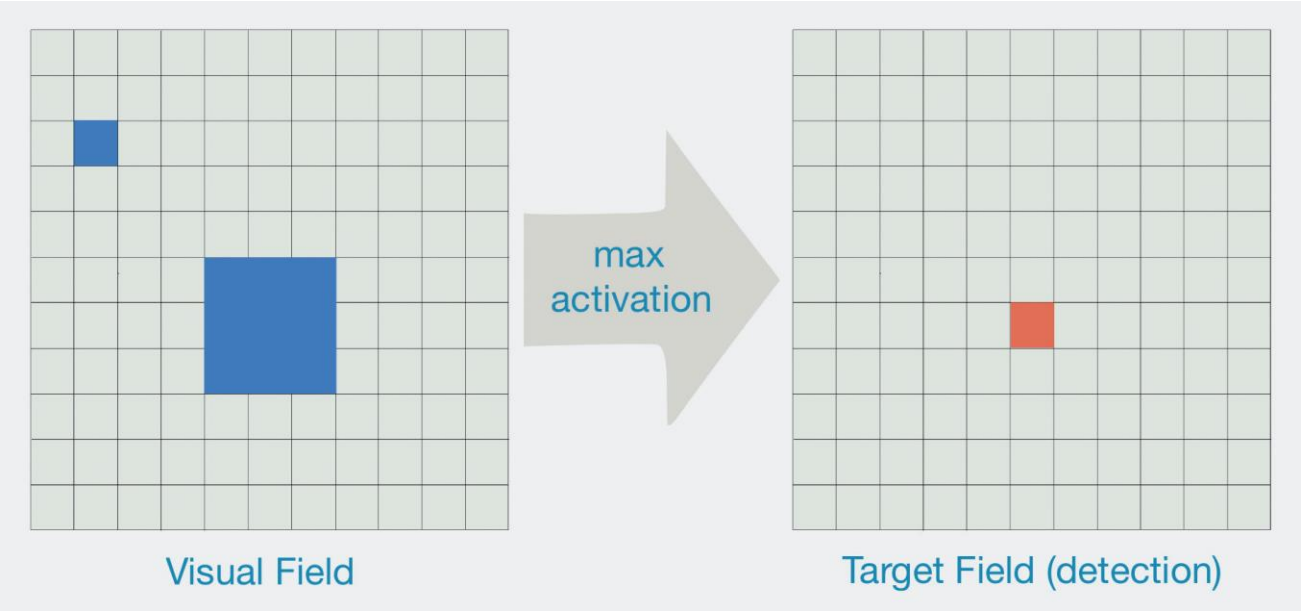
***Постановка задачи***

Задача зрительного различения состоит в том, чтобы отличить крупный объект от маленького объекта в двумерном зрительном пространстве независимо от их расположения в поле зрения и положения относительно друг друга. Задача зрительного различения решается специализированной различающей нейросетью – зрительным дискриминатором, – которая развивается на субстрате, сконфигурированном как сэндвич пространства состояний с двумя слоями:

– зрительное поле представляет собой двумерный массив датчиков, которые могут находиться в двух состояниях: включено или выключено (черно-белое изображение);

– целевое поле – это двумерный массив выходов со значениями активации в диапазоне [0,1].

Схема задачи зрительного различения показана на рисунке:




*Задача зрительного различения*

Различаемые объекты представлены в виде двух квадратов, разделенных пустым пространством. Большой объект в три раза превосходит маленький по каждому измерению. Алгоритм, который необходимо построить, должен найти центр более крупного объекта. Обнаружение основано на измерении значений активации узлов нейросети в целевом поле.

Положение узла с наивысшим значением активации указывает на центр обнаруженного объекта. Цель – найти правильный паттерн связей между зрительным полем и целевым полем, который сопоставляет выходной узел с наибольшей активацией в целевом поле с центром большого объекта в зрительном поле. Кроме того, результат работы нейросети не должен зависеть от взаимного расположения объектов.

Алгоритм для задачи зрительного различения должен оценивать большое количество входных данных – значений, представляющих ячейки в зрительном поле. Кроме того, успешный алгоритм должен обнаружить стратегию, способную обрабатывать входные данные из нескольких ячеек одновременно. Такая стратегия должна основываться на общем принципе, позволяющем определять относительные размеры объектов в зрительном поле, которое представлено в виде двумерной сетки. Следовательно, общим геометрическим принципом, который должен быть обнаружен, является локальность.

Можно использовать принцип локальности в конфигурации целевой (различающей) нейросети, применяя определенный паттерн в схеме соединений между узлами зрительного поля и целевого поля. В этой схеме соединений отдельные узлы зрительного поля соединены со множеством смежных узлов вывода вокруг определенного местоположения в целевом поле. В результате активация выходного узла зависит от того, как много сигналов поступает в него через соединения с отдельными входными узлами.

Чтобы эффективно использовать принцип локальности, представление связей должно учитывать геометрию субстрата различающей нейросети и тот факт, что правильный паттерн связей повторяется по всей сети. Наилучшим кандидатом для такого представления является CPPN, которая может один раз обнаружить локальный паттерн связей и повторить его по сетке субстрата для сколь угодно большого разрешения.

### *Определение целевой функции*

Основная задача зрительного дискриминатора – правильно определить положение более крупного объекта независимо от взаимного расположения обоих объектов. Отсюда можно определить целевую функцию для управления процессом нейроэволюции. Целевая функция

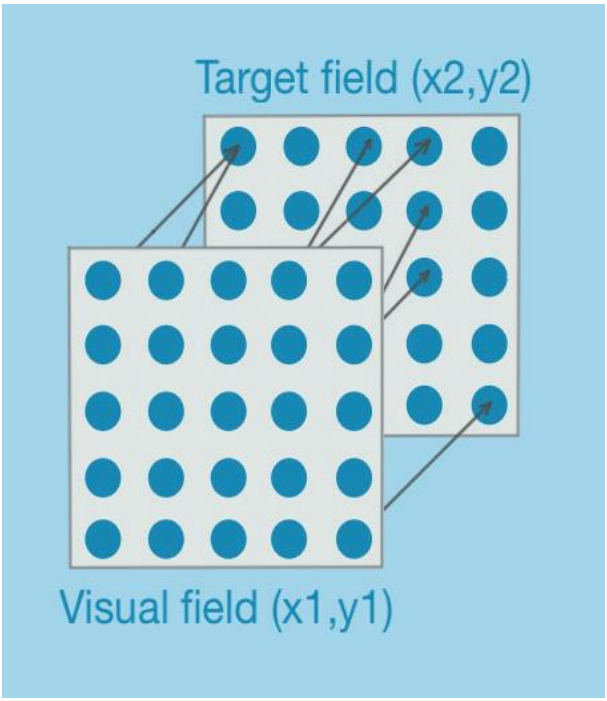
должна основываться на евклидовом расстоянии между точным положением более крупного объекта в поле зрения и его прогнозируемым положением в целевом поле.
Функция ошибки может быть представлена непосредственно как евклидово расстояние между фактическим и прогнозируемым положениями следующим образом:

$$\mathcal{L} = \sqrt{\sum_{i=1}^{2} (G_i - P_i)^2},$$

где $L$ – функция ошибки, $G_i$ – истинные координаты большого объекта, а $P_i$ –координаты, предсказанные нейросетью.
Используя функцию ошибки, определенную ранее, мы можем представить целевую функцию следующим образом:

$$\mathcal{F}_n = 1.0 - \frac{\mathcal{L}}{D_{max}}.$$

Здесь $D_{max}$ – максимально возможное расстояние между двумя точками в пределах целевого пространства поля. Формула целевой функции гарантирует, что рассчитанный показатель приспособленности ($F_n$) всегда находится в интервале [0,1].
Во время обучения нейросети зрительного дискриминатора, используется фиксированное разрешение зрительного и целевого полей, с размерностью 11×11. Следовательно, соединительная CPPN должна связать между собой 121 вход зрительного поля и 121 выход целевого поля, что в итоге потенциально может дать 14 641 значение веса связи.
На рисунке показана схема субстрата для нейросети зрительного дискриминатора.




*Субстрат пространства состояний для нейросети зрительного дискриминатора*

Нейросеть дискриминатора имеет два слоя с узлами, образующими по одной двумерной плоской сетке на слой. Соединительная CPPN «рисует» паттерны связей, проводя связи от узлов одного уровня к узлам другого уровня.
На каждом поколении эволюции каждый индивидуум в популяции (геном, кодирующий CPPN) оценивается на его способность создавать паттерны связей нейросети дискриминатора. Затем зрительный дискриминатор проверяется, чтобы определить, может ли он найти центр большого объекта в поле зрения. Всего для текущей нейросети предусмотрено 75 оценочных испытаний, в которых два объекта расположены в разных местах. В каждом испытании необходимо поместить маленький объект в одну из 25 позиций, равномерно распределенных в поле зрения.

Центр большого объекта находится в пяти шагах от маленького объекта вправо, вниз или по диагонали. Если большой объект не полностью вписывается в поле зрения, он переходит на другую сторону. Таким образом, учитывая логику размещения объектов относительно друг друга и сетки, необходимо оценить все возможные конфигурации в 75 испытаниях.

### *Тестовая среда зрительного дискриминатора*

Сначала нужно определить тестовую среду и предоставить доступ к набору данных, который содержит все возможные конфигурации зрительных полей. Набор данных, создается во время инициализации тестовой среды.
Тестовая среда состоит из двух основных компонентов:
– структура данных для хранения определений зрительного поля;
– менеджер тестовой среды, который хранит набор данных и предоставляет средства для оценки нейросети дискриминатора.

### *Определение зрительного поля*

Конфигурация зрительного поля для каждого из упомянутых ранее 75 испытаний сохраняется в классе Python VisualField. Он имеет следующий конструктор:

```
def __init__(self, big_pos, small_pos, field_size):
self.big_pos = big_pos
self.small_pos = small_pos
self.field_size = field_size
self.data = np.zeros((field_size, field_size))
# Позиция маленького объекта.
self._set_point(small_pos[0], small_pos[1])
# Позиция большого объекта.
offsets = [-1, 0, 1]
for xo in offsets:
for yo in offsets:
self._set_point(big_pos[0] + xo, big_pos[1] + yo)
```

Конструктор VisualField принимает в качестве параметров кортеж с координатами (x,y) большого и маленького объектов, а также размер зрительного поля. Мы рассматриваем квадратное поле, поэтому размер зрительного поля одинаковый вдоль каждой оси. Внутренним представлением зрительного поля является двумерный двоичный массив, где единицы представляют позиции, занятые объектами, а нули – это пустые пространства.
Он хранится в поле self.data, которое представляет собой массив NumPy с размерностью (2, 2). Маленький объект имеет размер 1×1, а размерность большого объекта в три раза больше. Следующий фрагмент из исходного кода конструктора создает представление большого объекта в массиве данных:

```
offsets = [-1, 0, 1]
for xo in offsets:
for yo in offsets:
self._set_point(big_pos[0] + xo, big_pos[1] + yo)
```

Конструктор класса VisualField получает координаты центра большого объекта в виде кортежа (x,y). Предыдущий фрагмент кода рисует большой объект, начиная с верхнего левого угла (x-1, y-1) и заканчивая нижним правым углом (x+1, y+1).
Функция _set_point(self,x,y), упомянутая в предыдущем фрагменте, устанавливает значение 1.0 в определенной позиции в поле self.data:

```
def _set_point(self, x, y):
```

```
px, py = x, y
if px < 0:
px = self.field_size + px
elif px >= self.field_size:
px = px - self.field_size
if py < 0:
py = self.field_size + py
elif py >= self.field_size:
py = py - self.field_size
self.data[py, px] = 1 # В Numpy индекс имеет вид [строка, столбец]
```

Функция _set_point(self, x, y) выполняет перенос координат, когда значение координат превышает допустимое количество измерений на ось. Например, для оси x исходный код для переноса значений координат выглядит следующим образом:

```
if px < 0:
px = self.field_size + px
elif px >= self.field_size:
px = px - self.field_size
```

Исходный код для переноса координат вдоль оси y аналогичен. После переноса координат, указанных в качестве параметров функции (при необходимости), мы присваиваем соответствующим позициям в поле self.data значение 1.0.
NumPy выполняет индексацию в виде [строка, столбец]. Поэтому мы должны поставить y в первую позицию и x во вторую позицию индекса.

### *Рабочая среда зрительного дискриминатора*

Рабочая среда зрительного дискриминатора содержит сгенерированный набор данных с описаниями зрительного поля. Кроме того, она предоставляет методы для создания набора данных и оценки нейросети дискриминатора на конкретном наборе данных. Класс Python VDEnvironment содержит определения всех упомянутых методов, а также соответствующих структур данных.
Рассмотрим все важные компоненты класса VDEnvironment.
Конструктор класса определяется следующим образом:

```
def __init__(self, small_object_positions, big_object_offset,
field_size):
self.s_object_pos = small_object_positions
self.data_set = []
self.b_object_offset = big_object_offset
self.field_size = field_size
self.max_dist = self._distance((0, 0),
(field_size - 1, field_size - 1))
# Создание тестового набора данных
self._create_data_set()
```

Первый параметр конструктора VDEnvironment представляет собой массив с определениями всех возможных положений малых объектов, определенных как последовательность значений координат для каждой оси. Второй параметр определяет смещение координат центра большого объекта от координат малого объекта. В качестве значения этого параметра используем 5. Наконец, третий параметр – это размер зрительного поля в двух измерениях.
Сохранив все полученные параметры в полях объекта, можно вычислить максимально возможное расстояние между двумя точками в зрительном поле следующим образом:

```
self.max_dist = self._distance((0, 0),
(field_size - 1, field_size - 1))
```

Евклидово расстояние между верхним левым и нижним правым углами зрительного поля сохраняется в поле self.max_dist. Это значение будет использовано позже для нормализации расстояний между точками зрительного поля, чтобы сохранить их в интервале [0,1].
Функция _create_data_set() создает все возможные наборы данных с учетом заданных параметров среды. Исходный код этой функции выглядит следующим образом:

```
def _create_data_set(self):
for x in self.s_object_pos:
for y in self.s_object_pos:
# Диагональ
vf = self._create_visual_field(x, y,
x_off=self.b_object_offset,
y_off=self.b_object_offset)
self.data_set.append(vf)
# Вправо
vf = self._create_visual_field(x, y,
x_off=self.b_object_offset,
y_off=0)
self.data_set.append(vf)
# Вниз
vf = self._create_visual_field(x, y,
x_off=0,
y_off=self.b_object_offset)
self.data_set.append(vf)
```

Функция выполняет итерацию по позициям маленького объекта вдоль двух осей и пытается создать большой объект по координатам, расположенным справа, снизу или по диагонали от координат маленького объекта.
Функция _create_visual_field создает соответствующую конфигурацию зрительного поля, используя координаты маленького объекта (sx, sy) и смещение центра большого объекта (x_off, y_off). Следующий исходный код показывает, как это реализовано:

```
def _create_visual_field(self, sx, sy, x_off, y_off):
bx = (sx + x_off) % self.field_size # Перенос по координате X
by = (sy + y_off) % self.field_size # Перенос по координате Y
# Создаем зрительное поле.
return VisualField(big_pos=(bx, by), small_pos=(sx, sy),
field_size=self.field_size)
```

Если координаты большого объекта, вычисленные предыдущей функцией, находятся вне пространства зрительного поля, применяем перенос следующим образом:

```
if bx >= self.field_size:
bx = bx - self.field_size # Перенос
```

Предыдущий фрагмент показывает перенос по оси x. Перенос по оси y выполняется аналогично. Создаем объект VisualField и возвращаем его для добавления в набор данных.
Тем не менее наиболее интересная часть описания VDEnvironment связана с оценкой нейросети дискриминатора, которая выполняется в функции evaluate_net(self, net) следующим образом:

```
def evaluate_net(self, net):
avg_dist = 0
# Вычисление предсказанной позиции
```

```
for ds in self.data_set:
# Вычисляем выходные значения
outputs, x, y = self.evaluate_net_vf(net, ds)
# Находим расстояние до большого объекта
dist = self._distance((x, y), ds.big_pos)
avg_dist = avg_dist + dist
avg_dist /= float(len(self.data_set))
# Нормализация ошибки предсказания позиции
error = avg_dist / self.max_dist
# Приспособленность
fitness = 1.0 - error
return fitness, avg_dist
```

Эта функция получает нейросеть зрительного дискриминатора в качестве параметра и возвращает вычисленную оценку приспособленности и среднее расстояние между предсказанными и истинными координатами большого объекта, рассчитанное для всех рассмотренных зрительных полей. Среднее расстояние рассчитывается следующим образом:

```
for ds in self.data_set:
# Вычисляем выходные значения
_, x, y = self.evaluate_net_vf(net, ds)
# Находим расстояние до большого объекта
dist = self._distance((x, y), ds.big_pos)
avg_dist = avg_dist + dist
avg_dist /= float(len(self.data_set))
```

Предыдущий исходный код перебирает все объекты VisualField в наборе данных и использует нейросеть зрительного дискриминатора для предсказания координат большого объекта. После этого необходимо вычислить расстояние (ошибку предсказания) между истинным и предсказанным положениями большого объекта. Далее, находим среднее значение ошибки предсказания и нормализуем его следующим образом:

```
# Нормализованная ошибка предсказания
error = avg_dist / self.max_dist
```

В соответствии с предыдущим кодом максимально возможное значение ошибки составляет 1.0. Значение показателя приспособленности вычисляется как разность между 1.0 и значением ошибки, поскольку приспособленность возрастает по мере уменьшения ошибки:

```
# fitness
fitness = 1.0 - error
```

Функция evaluate_net возвращает рассчитанный показатель приспособленности вместе с ненормализованной ошибкой обнаружения.

Функция evaluate_net_vf(self, net, vf) предоставляет средства для оценки нейросети дискриминатора на конкретном объекте VisualField и реализована в следующем блоке кода:

```
def evaluate_net_vf(self, net, vf):
depth = 1 # У нас только 2 слоя
net.Flush()
# Подготовка входа
inputs = vf.get_data()
net.Input(inputs)
# Активация
[net.Activate() for _ in range(depth)]
# Получаем выходы
```

```
outputs = net.Output()
# Находим координаты большого объекта
x, y = self._big_object_coordinates(outputs)
return outputs, x, y
```

Предыдущая функция получает нейросеть дискриминатора в качестве первого параметра и объект VisualField в качестве второго параметра.
После этого она формирует развернутый входной массив из объекта VisualField и использует его в качестве входных данных для нейросети дискриминатора:

```
inputs = vf.get_data()
net.Input(inputs)
```

После определения входов дискриминатора, он должен быть активирован для распространения входных значений по всем сетевым узлам. Нейросеть дискриминатора имеет только два слоя, что определяется пространственной конфигурацией субстрата. Следовательно, нужно активировать ее дважды – по одному разу для каждого слоя. После распространения сигнала активации через оба уровня нейросети дискриминатора можно определить положение большого объекта в целевом поле через индекс максимального значения в выходном массиве. Используя функцию _big_object_coordinates(self, outputs), можно извлечь декартовы координаты (x, y) большого объекта в целевом поле.
Наконец, функция evaluate_net_vf возвращает необработанный выходной массив вместе с извлеченными декартовыми координатами (x, y) большого объекта в пространстве целевого поля.
Функция _big_object_coordinates(self, outputs) извлекает декартовы координаты большого объекта в пространстве целевого поля из необработанных выходных данных, полученных из нейросети дискриминатора. Исходный код функции выглядит следующим образом:

```
def _big_object_coordinates(self, outputs):
max_activation = -100.0
max_index = -1
for i, out in enumerate(outputs):
if out > max_activation:
max_activation = out
max_index = i
# Получение координат точки максимальной активации
x = max_index % self.field_size
y = int(max_index / self.field_size)
return (x, y)
```

Сначала функция перебирает выходной массив и находит индекс максимального значения:

```
max_activation = -100.0
max_index = -1
for i, out in enumerate(outputs):
if out > max_activation:
max_activation = out
max_index = I
```

После этого она использует найденный индекс для вычисления декартовых координат с учетом размера целевого поля:

```
x = max_index % self.field_size
y = int(max_index / self.field_size)
```

Наконец, функция возвращает кортеж (x, y) с декартовыми координатами большого объекта в целевом поле.

## *HyperNEAT*

Задачу зрительного различения можно решить с помощью метода HyperNEAT. Для этого нужно воспользоваться библиотекой, которая обеспечивает реализацию алгоритма HyperNEAT. Воспульзуемся библиотекой MultiNEAT Python.
Функция движка позволяет осуществлять разработку, используя доступные гиперпараметры и инициализированную тестовую среду зрительного дискриминатора. Код функции состоит из нескольких важных частей.

*Инициализация первой популяции геномов CPPN.*

В представленном далее блоке кода сначала происходит инициализация начального числа генератора случайных чисел текущим системным временем. После этого создается подходящая конфигурация субстрата для нейросети дискриминатора, способная работать со зрительным полем заданной размерности. Затем на основе конфигурации субстрата необходимо создать геном CPPN:

```
# Начальное значение генератора случайных чисел.
seed = int(time.time())
# Создаем субстрат.
substrate = create_substrate(num_dimensions)
# Создаем геном CPPN и популяцию.
g = NEAT.Genome(0,
substrate.GetMinCPPNInputs(),
0,
substrate.GetMinCPPNOutputs(),
False,
NEAT.ActivationFunction.UNSIGNED_SIGMOID,
NEAT.ActivationFunction.UNSIGNED_SIGMOID,
0,
params, 0)
pop = NEAT.Population(g, params, True, 1.0, seed)
pop.RNG.Seed(seed)
```

Геном CPPN имеет необходимое количество входных и выходных узлов, предоставляемых субстратом. Сначала в качестве функции активации узла он использует беззнаковый сигмоид. Позже, в ходе эволюции тип функции активации каждого узла CPPN будет изменен в соответствии с процедурами алгоритма HyperNEAT. Наконец, на основе инициализированного генома CPPN и гиперпараметров HyperNEAT создается исходная популяция.

*Запуск нейроэволюции в течение нужного числа поколений*

Сначала создаем промежуточные переменные для хранения результатов выполнения и сборщик статистики (Statistics). После этого выполняем цикл эволюции для числа поколений, указанного в параметре n_generations:

```
start_time = time.time()
best_genome_ser = None
best_ever_goal_fitness = 0
best_id = -1
solution_found = False
stats = Statistics()
for generation in range(n_generations):
```

В рамках цикла эволюции получаем список геномов, принадлежащих к популяции в текущем поколении, и оцениваем все геномы из списка на приспособленность к условиям тестовой среды следующим образом:

```
genomes = NEAT.GetGenomeList(pop)
# Получаем геномы.
genome, fitness, distances = eval_genomes(genomes,
vd_environment=vd_environment,
substrate=substrate,
generation=generation)
stats.post_evaluate(max_fitness=fitness, distances=distances)
solution_found = fitness >= FITNESS_THRESHOLD
```

Сохраняем в сборщик статистики значения, возвращенные функцией eval_genomes(genomes, substrate, vd_environment, generation) для текущего поколения. Кроме того, используем возвращаемую функцией оценку приспособленности, чтобы оценить, было найдено успешное решение или нет. Если показатель приспособленности превышает значение FITNESS_THRESHOLD, мы считаем, что было найдено успешное решение.
Далее, если было найдено успешное решение или текущий показатель приспособленности является максимальным из когда-либо достигнутых показателей приспособленности, сохраняем геном CPPN и текущий показатель приспособленности:

```
if solution_found or best_ever_goal_fitness < fitness:
best_genome_ser = pickle.dumps(genome)
best_ever_goal_fitness = fitness
best_id = genome.GetID()
```

Кроме того, если будет найдено успешное решение, прерываем цикл эволюции и переходим к этапу вывода отчетов:

```
if solution_found:
print('Solution found at generation: %d, best fitness: %f,
species count: %d' % (generation, fitness, len(pop.Species) ))
break
```

Если удачное решение не было найдено, в консоль необходимо вывести статистику для текущего поколения и переходим к следующему поколению:_

```
# Переходим к следующему поколению
pop.Epoch()
# Выводим статистику в консоль
gen_elapsed_time = time.time() - gen_time
print("Best fitness: %f, genome ID: %d" % (fitness, best_id))
print("Species count: %d" % len(pop.Species))
print("Generation elapsed time: %.3f sec" % (gen_elapsed_time))
print("Best fitness ever: %f, genome ID: %d"
% (best_ever_goal_fitness, best_id))
```

После завершения основного цикла эволюции в консоль выводятся результаты эксперимента, основанные на статистике, собранной в цикле.

*Сохранение результатов эксперимента*

Результаты эксперимента собираются и сохраняются в текстовом и графическом представлениях (файлы SVG). Начнем с вывода на печать общей статистики производительности алгоритма:

```
print("\nBest ever fitness: %f, genome ID: %d"
% (best_ever_goal_fitness, best_id))
print("\nTrial elapsed time: %.3f sec" % (elapsed_time))
print("Random seed:", seed)
```

Первые три строки данного кода выводят на консоль лучшую оценку приспособленности, полученную среди всех поколений эволюции. После этого выводим на печать продолжительность эксперимента и используемое случайное начальное значение.
Если необходимо сохранить или показать визуализацию, то будут вызваны соответствующие функции:

```
# Визуализация результатов эксперимента
show_results = not silent
if save_results or show_results:
net = NEAT.NeuralNetwork()
best_genome.BuildPhenotype(net)
visualize.draw_net(net, view=show_results, node_names=None,
directory=trial_out_dir, fmt='svg')
```

Этот фрагмент кода рисует сетевой граф CPPN и выводит на печать статистику графа. Далее переходим к визуализации вывода нейросети дискриминатора:

```
# Визуализация активаций от наилучшего генома
net = NEAT.NeuralNetwork()
best_genome.BuildHyperNEATPhenotype(net, substrate)
# Выбор случайного зрительного поля
index = random.randint(0, len(vd_environment.data_set) - 1)
vf = vd_environment.data_set[index]
# Отображение активаций
outputs, x, y = vd_environment.evaluate_net_vf(net, vf)
visualize.draw_activations(outputs, found_object=(x, y), vf=vf,
dimns=num_dimensions, view=show_results,_
filename=os.path.join(trial_out_dir,
"best_activations.svg"))
```

Представляем в графическом виде выходные данные активации, полученные путем запуска нейросети дискриминатора в тестовой среде. Ииспользуем зрительное поле, которое выбирается случайным образом из набора данных эксперимента.
Наконец, отображаем общую статистику, собранную во время эксперимента:

```
# Visualize statistics
visualize.plot_stats(stats, ylog=False, view=show_results,
filename=os.path.join(trial_out_dir, 'avg_fitness.svg'))
```

Графическое отображение статистики включает в себя лучшие оценки приспособленности и средние расстояния ошибок, собранные за поколения эволюции.

*Функция конструктора субстрата*

Метод HyperNEAT построен вокруг понятия субстрата, определяющего структуру нейросети дискриминатора. Поэтому крайне важно создать правильную конфигурацию субстрата,

которая будет задействована во время выполнения эксперимента. Процедуры создания субстрата определены в следующих двух функциях.

1) Функция конструктора субстрата create_substrate создает объект субстрата:

```
def create_substrate(dim):
# Строим конфигурации входных и выходных декартовых листов
inputs = create_sheet_space(-1, 1, dim, -1)
outputs = create_sheet_space(-1, 1, dim, 0)
substrate = NEAT.Substrate( inputs, [], # hidden outputs)
substrate.m_allow_input_output_links = True
...
substrate.m_hidden_nodes_activation = \
NEAT.ActivationFunction.SIGNED_SIGMOID
substrate.m_output_nodes_activation = \
NEAT.ActivationFunction.UNSIGNED_SIGMOID
substrate.m_with_distance = True
substrate.m_max_weight_and_bias = 3.0
return substrate
```

Данная функция сначала создает два разбитых на сетку декартовых листа, которые представляют входные данные (зрительное поле) и выходные данные (целевое поле) в конфигурации субстрата. Как вы помните, для этого эксперимента мы выбрали конфигурацию слоев типа «сэндвич». Затем экземпляр субстрата инициализируется в соответствии с созданными конфигурациями полей:

```
inputs = create_sheet_space(-1, 1, dim, -1)
outputs = create_sheet_space(-1, 1, dim, 0)
substrate = NEAT.Substrate( inputs, [], # hidden outputs)
```

Субстрат не использует никаких скрытых узлов; вместо них используется пустой список.
Далее нужно настроить субстрат, чтобы разрешить только соединения, направленные от входных к выходным узлам, и использовать сигмоидную функцию активации со знаком на выходных узлах. Устанавливаем максимальные значения для смещения и веса соединения.

2) Функция create_sheet_space, вызываемая конструктором субстрата, определяется следующим образом:

```
def create_sheet_space(start, stop, dim, z):
lin_sp = np.linspace(start, stop, num=dim)
space = []
for x in range(dim):
for y in range(dim):
space.append((lin_sp[x], lin_sp[y], z))
return space
```

Функция create_sheet_space получает начальную и конечную координаты сетки в одном измерении вместе с количеством измерений сетки. Также указана координата z листа. Используя указанные параметры, предыдущий код создает равномерное линейное пространство с координатами, начинающимися в диапазоне [start, stop], с шагом dim:

```
lin_sp = np.linspace(start, stop, num = dim)
```

Затем используем это линейное пространство, чтобы заполнить двумерный массив координатами узлов сетки:

```
space = []
for x in range(dim):
```

```
for y in range(dim):
space.append((lin_sp[x], lin_sp[y], z))
```

Функция ____create_sheet_space возвращает конфигурацию сетки в виде двумерного массива.

*Оценка приспособленности*

Оценка приспособленности генома является важной частью любого алгоритма нейроэволюции, включая метод HyperNEAT. Как вы видели, главный цикл эксперимента вызывает функцию eval_genomes для оценки приспособленности всех геномов в популяции для каждого поколения. Давайте рассмотрим детали реализации оценки приспособленности, которая состоит из двух основных функций.

1) Функция eval_genomes оценивает все геномы в популяции:

```
def eval_genomes(genomes, substrate, vd_environment, generation):
best_genome = None
max_fitness = 0
distances = []
for genome in genomes:
fitness, dist = eval_individual(genome, substrate,
vd_environment)
genome.SetFitness(fitness)
distances.append(dist)
if fitness > max_fitness:
max_fitness = fitness
best_genome = genome
return best_genome, max_fitness, distances
```

Функция eval_genomes в качестве параметров принимает список геномов, конфигурацию субстрата нейросети дискриминатора, инициализированную тестовую среду и идентификатор текущего поколения. Первые строки функции создают промежуточные переменные для хранения результатов оценки:

```
best_genome = None
max_fitness = 0
distances = []
```

После этого перебираем все геномы популяции и собираем соответствующую статистику:

```
for genome in genomes:
fitness, dist = eval_individual(genome, substrate,
vd_environment)
genome.SetFitness(fitness)
distances.append(dist)
if fitness > max_fitness:
max_fitness = fitness
best_genome = genome
```

Наконец, функция eval_genomes возвращает собранную статистику в виде кортежа (best_genome, max_fitness, distance).

2) Функция eval_individual позволяет нам оценить приспособленность отдельного генома:

```
def eval_individual(genome, substrate, vd_environment):
# Создает нейросеть из генома CPPN и субстрата.
net = NEAT.NeuralNetwork()
genome.BuildHyperNEATPhenotype(net, substrate)
```

```
fitness, dist = vd_environment.evaluate_net(net)
return fitness, dist
```

Код создает фенотип нейросети дискриминатора с использованием генома CPPN, предоставленного в качестве параметра. После этого фенотип нейросети дискриминатора оценивается на приспособленность к тестовой среде.
Функция eval_individual возвращает оценку приспособленности и величину ошибки, полученные из тестовой среды во время оценки фенотипа. Теперь, когда мы завершили настройку, давайте приступим к эксперименту по зрительному различению.

### *Зрительное различение объектов*

Используем следующую конфигурацию зрительного поля:
*Размер видимого поля 11x11*
*Положение мелких объектов в поле зрения вдоль каждой оси [1, 3, 5, 7, 9]*
*Размер маленького объекта 1x1*
*Размер большого объекта 3x3*
*Смещение центра большого объекта от маленького объекта 5*
Далее нужно выбрать подходящие значения гиперпараметров HyperNEAT, что позволит нам найти успешное решение задачи зрительного различения.
Гиперпараметр пределяет, как развивать соединительную CPPN, используя процесс нейроэволюции. Нейросеть зрительного дискриминатора создается путем наложения соединительной CPPN на субстрат.

*Выбор гиперпараметра*

Библиотека MultiNEAT использует класс Parameters для хранения всех необходимых гиперпараметров. Чтобы установить соответствующие значения гиперпараметров, определяем функцию create_hyperparameters в скрипте Python для запуска эксперимента. Опишем основные гиперпараметры, которые оказывают существенное влияние на производительность алгоритма HyperNEAT в этом эксперименте.
1) Функция create_hyperparameters начинается с создания объекта Parameters для хранения параметров HyperNEAT:

```
params = NEAT.Parameters()
```

2) Используем популяции геномов среднего размера, чтобы ускорить вычисления. В то же время необходимо сохранить достаточное количество индивидов в популяции для эволюционного разнообразия. Численность популяции определяется следующим образом:

```
params.PopulationSize = 150
```

3) Цель - создание компактных геномов CPPN, которые имеют как можно меньше узлов, повышения эффективности кодирования. Задаем очень маленькую вероятность добавления нового узла во время эволюции, а также оставляем довольно низкую вероятность создания новой связи:

```
params.MutateAddLinkProb = 0.1
params.MutateAddNeuronProb = 0.03
```

4) Метод HyperNEAT создает геномы CPPN с различными типами функций активации в скрытых и выходных узлах. Следовательно, нкжно определить вероятность мутации, которая меняет тип активации узла. Кроме того, предпочтительно использование одной четырех

функций активации: гауссовой со знаком, сигмоиды со знаком, синусоиды со знаком и линейной функции.
Устанавливаем вероятности выбора любой из функций активации равными 1.0:

```
params.MutateNeuronActivationTypeProb = 0.3
params.ActivationFunction_SignedGauss_Prob = 1.0
params.ActivationFunction_SignedSigmoid_Prob = 1.0
params.ActivationFunction_SignedSine_Prob = 1.0
params.ActivationFunction_Linear_Prob = 1.0
```

5) Определяем количество видов в популяции, которое должно оставаться в интервале [5,10], и устанавливаем допустимую величину стагнации вида в течение 100 поколений. Эта конфигурация поддерживает умеренное видовое разнообразие, но сохраняет виды достаточно долго, чтобы позволить им развиваться и создавать полезные конфигурации генома CPPN:

```
params.SpeciesMaxStagnation = 100
params.MinSpecies = 5
params.MaxSpecies = 10
```

*Настройка рабочей среды*

Библиотека MultiNEAT обеспечивает реализацию алгоритма HyperNEAT. Необходимо создать соответствующую среду Python, которая включает в себя библиотеку MultiNEAT и все необходимые зависимости. Это можно сделать с помощью Anaconda, выполнив следующие команды в командной строке:

```
$ conda create --name vd_multineat python=3.5
$ conda activate vd_multineat
$ conda install -c conda-forge multineat
$ conda install matplotlib
$ conda install -c anaconda seaborn
$ conda install graphviz
$ conda install python-graphviz
```

Эти команды создают и активируют виртуальную среду vd_multineat на основе Python 3.5. После этого устанавливатся последняя версия библиотеки MultiNEAT вместе с зависимостями, которые используются для визуализации результатов.

*Запуск эксперимента по зрительному различению*

Чтобы запустить эксперимент, нужно перейти в локальный каталог, содержащий скрипт vd_experiment_multineat.py, и выполнить следующую команду:

```
$ python vd_experiment_multineat.py
```

Также нужно активировать соответствующую виртуальную среду с помощью команды:

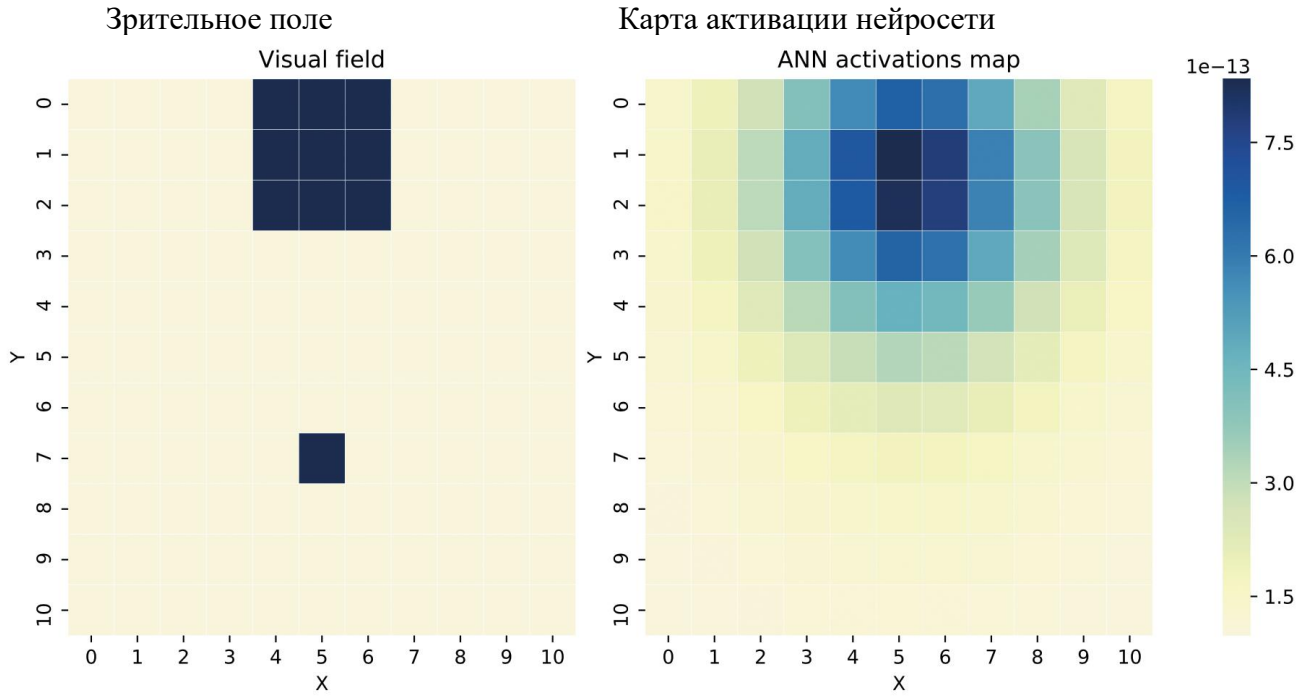
```
$ conda activ vd_multineat.
```

Спустя определенное число поколений будет найдено успешное решение:

```
****** Generation: 16 ******
Best fitness: 0.995286, genome ID: 2410
Species count: 11
Generation elapsed time: 3.328 sec
Best fitness ever: 0.995286, genome ID: 2410
```

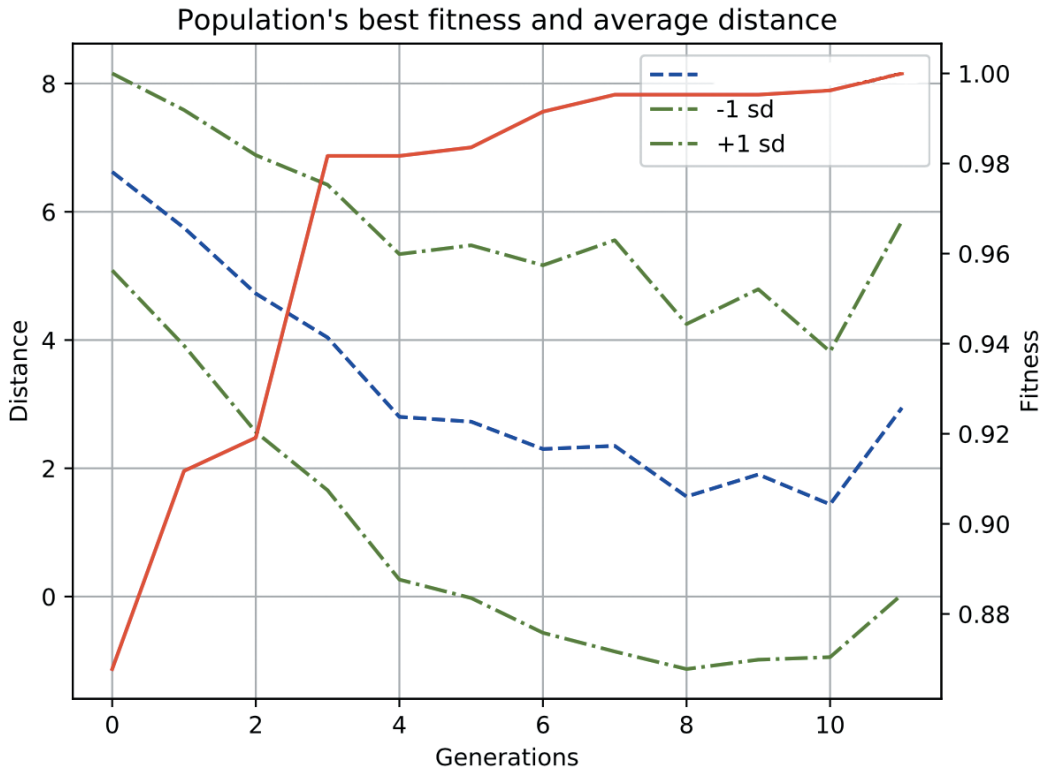
****** Generation: 17 *******
*Solution found at generation: 17, best fitness: 1.000000, species count: 11*
*Best ever fitness: 1.000000, genome ID: 2565*
*Trial elapsed time: 57.753 sec*
*Random seed: 1568629572*
*CPPN nodes: 10, connections: 16*
*Running test evaluation against random visual field: 41*
*Substrate nodes: 242, connections: 14641*
*found (5, 1)*
*target (5, 1)*

Решение было найдено в 16-м поколении. Идентификатор успешного генома CPPN – 2565, и этот геном имеет 10 узлов и 16 связей между ними. Кроме того, здесь представлены результаты оценки нейросети дискриминатора, полученного с помощью лучшего генома CPPN в отношении случайно выбранного зрительного поля.
Найденные декартовы координаты большого объекта в целевом поле и фактические координаты в зрительном поле совпадают (5, 1), что свидетельствует о способности найденного решения различать объекты с высокой точностью.
На рисунке приведено визуальное представление уровней активации выходов нейросети зрительного дискриминатора, полученных во время тестового прогона.




*Активации целевого поля зрительного дискриминатора*

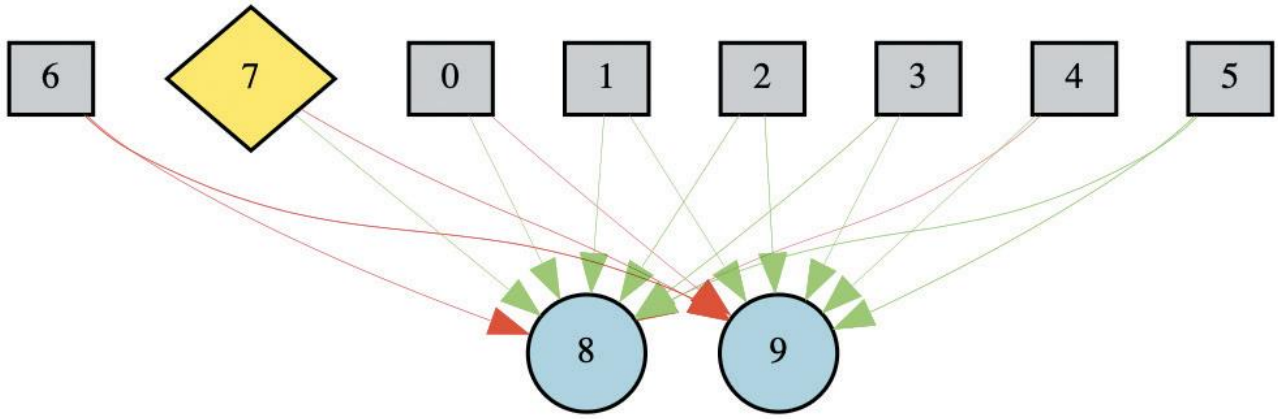
Правая часть рисунка отображает значения активации целевого поля (выходного слоя) нейросети дискриминатора, которые было получено во время оценки на случайном зрительном поле. В левой части графика можно увидеть фактическую конфигурацию зрительного поля. Максимальное значение активации целевого поля (самая темная ячейка) находится точно в той же позиции, что и центр большого объекта с координатами в зрительном поле (5,1).
Значениям активации нейросети очень малы: минимальное значение составляет $\sim 1\times10^{-13}$, а максимальное – только $\sim 9\times10^{-13}$.
Следующий график позволяет проанализировать ход процесса эволюции в течение нескольких поколений и сделать вывод о том, насколько хорошо полученные соединительные CPPN справляются с задачей построения успешной нейросети зрительного дискриминатора.




*Наилучшие оценки приспособленности и средние ошибки для нейросети зрительного дискриминатора*

На рисунке показано изменение оценок приспособленности (восходящая линия) и средних ошибок (нисходящая линия) для каждого поколения эволюции. Показатели приспособленности почти достигли максимального значения уже в третьем поколении эволюции, но потребовалось еще семь поколений, чтобы проработать конфигурации генома CPPN и найти победителя. Кроме того, вы можете видеть, что среднее расстояние между предсказанным и истинным положениями большого объекта постепенно уменьшается в процессе эволюции.

На следующем рисунке представлен граф фенотипа CPPN, который использовался для построения связей в нейросети успешного зрительного дискриминатора.




*Граф фенотипа CPPN лучшего генома*

На графе фенотипа CPPN входные узлы помечены квадратами, выходные узлы – закрашенные кружки, а узел смещения – ромб.

Два выходных узла CPPN имеют следующее значение:

– первый узел (8) предоставляет вес связи;

– второй узел (9) определяет, экспрессирована ли связь.

Назначение входных узлов CPPN определено следующим образом:

– первые два узла (0 и 1) задают координаты точки (x, y) во входном слое субстрата;

– следующие два узла (2 и 3) задают координаты точки (x, y) в скрытом
– слое субстрата (не использовались в нашем эксперименте);
– следующие два узла (4 и 5) задают координаты точки (x, y) в выходном
– слое субстрата;
– последний узел (6) задает евклидово расстояние от точки на входном слое до начала координат.

Также можно заметить, что фенотип CPPN не содержит скрытых узлов. В процессе нейроэволюции удалось найти подходящие типы функций активации для выходных узлов CPPN в задаче зрительного различения. Имея всего 16 соединений между 10 узлами, фенотип CPPN смог закодировать паттерн связей, который способен покрыть субстрат зрительного поля разрешением 11×11, потенциально имеющего 14 641 соединение между узлами зрительного и целевого полей. Таким образом, мы достигли степени сжатия информации около 0,11 %, что весьма впечатляет.

Такая высокая степень сжатия стала возможной благодаря обнаружению геометрических закономерностей в связях субстрата соединительной CPPN.

Используя обнаруженные закономерности, CPPN может обойтись только несколькими паттернами (мотивами локальной связности) для всего пространства связей субстрата. После этого CPPN может неоднократно применять эти локальные паттерны в разных позициях субстрата, чтобы нарисовать полную схему связей между слоями субстрата – в нашем случае чтобы нарисовать связи между входным слоем (зрительное поле) и выходным слоем (целевое поле).

**Зачетно-экзаменационные материалы для промежуточной аттестации (экзамен)**

1. Постановка задачи поисковой оптимизации.
2. Классификация детерминированных задач поисковой оптимизации.
3. Классификация алгоритмов решения детерминированной задачи поисковой оптимизации.
4. Одношаговые алгоритмы. Правила выбора направления поиска. Правила выбора величины шага. Возврат при неудачном шаге.
5. Многошаговые алгоритмы.
6. Локальная безусловная оптимизация. Алгоритм градиентного спуска.
7. Алгоритмы градиентного спуска: с постоянным шагом, наискорейшего спуска, покоординатного спуска.
8. Локальная условная оптимизация. Алгоритм штрафных функций.
9. Глобальная оптимизация. Алгоритмы случайного поиска. Основные термины и определения.
10. Алгоритмы случайного поиска. Алгоритм Монте-Карло.
11. Алгоритмы случайного поиска. Алгоритм имитации отжига.
12. Двухфазные алгоритмы случайного поиска: алгоритм мультистарта, жадный адаптивный алгоритм случайного поиска.
13. Алгоритм поиска с запретами (Tabu Search algorithm, TS algorithm).
14. Общая схема эволюционных алгоритмов: основные термины и определения.
15. Способы кодирования особей в эволюционных алгоритмах.
16. Типовые генетические алгоритмы (ГА).
17. ГА: операторы отбора и селекции, операторы кроссинговера и мутации.
18. Оптимизация роем частиц. Канонический роевой алгоритм.
19. Пчелиный алгоритм оптимизации. Биологические предпосылки, схема и принцип работы пчелиного алгоритма.
20. Искусственные иммунные системы. Биологические основы. Оптимизация с помощью модели иммунной сети.
21. Бактериальная оптимизация. Биологические предпосылки. Канонический алгоритм бактериальной оптимизации.

22. Классификация методов метаоптимизации.
23. Статистическая параметрическая метаоптимизация.
24. Динамическая параметрическая метаоптимизация.
25. Неадаптивная динамическая метаоптимизация.
26. Адаптивная динамическая метаоптимизация.
27. Структурная метаоптимизация.
28. Структурно-параметрическая метаоптимизация.
29. Задача многоцелевой оптимизации (МЦО-задача) и алгоритмы ее решения.
30. Алгоритмы Парето-аппроксимации.
31. Эволюционные алгоритмы и нейроэволюционные методы.
32. Генетические операторы. Схемы кодирования генома. Коэволюция.
33. Алгоритм NEAT: схема кодирования, структурные мутации, видообразование.
34. Использование NEAT для оптимизации решения задачи XOR.
35. NEAT на основе гиперкуба.
36. Зрительное различение с NEAT на основе гиперкуба.

***Перечень компетенций (части компетенции), проверяемых оценочным средством:*** MF-3.1; MF-3.2; ML-3.1; ML-6.1; O-3.2; O-3.3.

**4.2 Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций**

**Методические рекомендации, определяющие процедуры оценивания выполнения лабораторных работ**

Задание считается выполненным при выполнении следующих условий:

- предоставлен исходный код;
- продемонстрирована работоспособность программы;
- студент понимает исходный код и отвечает на вопросы по его организации.

**Методические рекомендации, определяющие процедуры оценивания на экзамене**

Оценка «отлично»: точные формулировки алгоритмов, теорем и правильные доказательства; точные определения математических объектов и ясные и правильные определения объектов, характеризующихся неформализованными понятиями.

Оценка «хорошо»: при ответе на один вопрос даны точные формулировки алгоритмов, теорем и правильные доказательства; точные определения математических объектов и ясные и правильные определения объектов, характеризующихся неформализованными понятиями; при ответе на второй вопрос имеются неточности формулировки алгоритмов, теорем или пробелы в правильных доказательствах; недостаточно точные определения математических объектов или неясные и не совсем правильные определения объектов, характеризующихся неформализованными понятиями.

Оценка «удовлетворительно»: при ответе на оба вопроса имеются неточности формулировки алгоритмов, теорем или пробелы в правильных доказательствах; недостаточно точные определения математических объектов или неясные и не совсем правильные определения объектов, характеризующихся неформализованными понятиями.

Оценка «неудовлетворительно»: отсутствует ответ хотя бы на один из вопросов или имеются существенные неточности в формулировках алгоритмов, теорем, приведены неправильные доказательства; неверные определения математических объектов и неправильные определения объектов, характеризующихся неформализованными понятиями.

**4.3. Методические указания по организации вычислительной инфраструктуры**
**Условия применения:**

- Курс рассчитан на студентов 3-го года обучения.
- Наличие доступа к вычислительным ресурсам (GitLab, Google Colab или Yandex DataSphere, JupyterHub, Hugging Face).

Разработаны лабораторные работы;

Инфраструктура для приёма задач (gitlab, CI/CD) согласована с лабораторными работами.

**Цели, задачи и ожидаемые результаты**

Цели организации вычислительной инфраструктуры:

- дать начальное представление о работе в IT инфраструктуре (приучить пользоваться гитом, jupyter-ноутбуками).

Задачи преподавателя:

Организация регистрации студентов в Google Colab и Yandex DataSphere

Создание учетных записей студентов в gitlab вуза;

Настройка GitLab Runner для автоматического тестирования кода.

Разработка шаблонного репозитория для лабораторных работ с предустановленными зависимостями (PyTorch, Keras, TensorFlow, Hugging Face Transformers).

Написание автотестов для проверки корректности выполнения заданий (например, fine-tuning моделей).

Визуализация результатов тестирования через HTML-отчеты.

Подготовка инструкций по работе с Git и облачными ресурсами.

Ожидаемые результаты студентов:

- начальное представление о работе в IT инфраструктуре (гит, нейминг).
- Навыки запуска и тестирования глубоких нейросетей в облачных средах.
- Понимание CI/CD-процессов в контексте разработки глубоких нейросетей.

**Порядок реализации**

**Задача №1:** Организация регистрации студентов в Google Colab и Yandex DataSphere

**Задача №2:** Создание учетных записей студентов в gitlab вуза

**Задача №3:** Настройка GitLab Runner:

Для автоматического тестирования кода используется Docker-образ с предустановленными библиотеками (PyTorch, Keras, TensorFlow, Hugging Face Transformers).

Для выполнения CI/CD пайплайна был настроен GitLab Runner на удалённой виртуальной машине с ОС Ubuntu 24.04.

Последовательность настройки включала следующие шаги:

- Настройка системы – установка необходимых компонентов, таких, как Docker.
- Установка GitLab Runner по официальной инструкции.
- Регистрация Runner для частного сервера GitLab.

**Задача №4: Шаблонный репозиторий:**

Включает:

.gitlab-ci.yml для CI/CD.

Скрипты для предобработки текста и обучения моделей.

Примеры кода для работы с BERT, GPT и другими архитектурами.

**Задача №5: Автотесты:**

Проверяют корректность fine-tuning моделей (например, accuracy на тестовом датасете).

**Задача №6: Визуализация результатов:**

Генерация HTML-отчетов с результатами тестирования, включая метрики качества моделей.

**Порядок проверки корректности:**

- Наличие Git-репозиториев у всех студентов.
- Шаблонный репозиторий с подключенными автотестами.
- Инструкция по работе с Git и CI/CD в формате README.md.

Вся структура максимально адаптирована для копирования студентами и минимизации порога входа при выполнении лабораторных

### 4.4. Методические указания по организации лабораторных работ

**Условия применения:**

Курс рассчитан на студентов 3-го года обучения.

Наличие доступа к вычислительным ресурсам (GitLab) и к GPU/CPU (Kaggle, локальные серверы).

Разработана инфраструктура для приёма задач (Gitlab, CI/CD) и согласована с лабораторными работами и настроена на всех студентов образовательной программы;

Использование открытых датасетов и библиотек.

**Цели, задачи и ожидаемые результаты**

Цели организации лабораторных работ:

Закрепление теоретических знаний о классических и интеллектуальных методах оптимизации на практике.

Развитие навыков алгоритмического мышления и реализации оптимизационных алгоритмов для решения прикладных задач.

Подготовка к решению реальных задач в таких областях, как финансы, логистика, машинное обучение и проектирование.

Задачи преподавателя:

Обеспечить студентов структурированными лабораторными работами, охватывающими спектр методов от классических до современных интеллектуальных.

Предоставить доступ к необходимым программным средствам и вычислительным ресурсам (Python, библиотеки для оптимизации, среды для визуализации).

Организовать проверку и обратную связь по выполненным работам, акцентируя внимание на корректности реализации, анализе результатов и качестве выводов.

Ожидаемые результаты студентов:

Умение применять и сравнивать различные классы оптимизационных алгоритмов (градиентные, методы второго порядка, эволюционные, swarm-интеллект) на практике.

Владение библиотеками и фреймворками для научных вычислений и оптимизации (NumPy, SciPy, Matplotlib, DEAP, Scikit-learn).

Опыт решения прикладных задач оптимизации из разных предметных областей (поиск экстремума функций, условная оптимизация, комбинаторные задачи).

Навыки анализа поведения алгоритмов, включая сходимость, устойчивость и вычислительную сложность, а также визуализации процесса оптимизации.

### 4.5 Методические указания по организации проектной деятельности студентов

**Условия применения:**

Курс рассчитан на студентов 3-го года обучения,

Общее время на проект – не более 16 часов на каждого студента.

Имеется доступ к кейсам индустриальных партнеров.

**Цели, задачи и ожидаемые результаты**

**Цели организации вычислительной инфраструктуры:**

дать начальное представление о реальных задачах, решаемых с помощью глубоких нейронных сетей и возникающих проблемах.

**Задачи преподавателя:**

- сбор кейсов индустриальных партнеров;
- сбор кейсов преподавателей практиков и лабораторий в вузе;
- формирование ТЗ на зачетный проект на основе кейсов;
- разработка системы учёта результатов проекта в итоговой оценке зачета

**Ожидаемые результаты студентов:**

начальное представление о реальных задачах, решаемых с помощью глубоких нейросетей и возникающих проблемах.

Рассмотрим примеры кейсов.

**Кейсы ПАО «Сбербанк»**

**1. Оптимизация сети отделений банка (Branch Network Optimization)**
**Описание:** Используя данные о клиентском потоке, арендных ставках, демографии и транспортной доступности, необходимо разработать модель оптимизации сети отделений Сбербанка. Учесть затраты на содержание, потенциальную выручку и стратегические цели охвата регионов.

**Цель:** Определить оптимальную конфигурацию сети отделений (открытие, закрытие, переезд) для максимизации прибыли при сохранении качества обслуживания.

**Ожидаемые результаты:**

- Модель оптимизации размещения отделений с учетом пространственных данных
- Алгоритм оценки экономической эффективности каждого отделения
- План реструктуризации сети с расчетом ROI

**2. Оптимизация капитала банка (Capital Allocation Optimization)**
**Описание:** Разработать систему оптимального распределения капитала между различными направлениями бизнеса (кредитование, инвестиции, лизинг) с учетом риск-аппетита, регуляторных требований и рыночных условий.
**Цель:** Максимизировать доходность использования капитала при соблюдении нормативов ЦБ и ограничений по рискам.

**Ожидаемые результаты:**

- Многокритериальная модель оптимизации распределения капитала
- Алгоритм динамической корректировки allocation в зависимости от изменения рынка
- Инструмент для сценарного анализа и стресс-тестирования

**3. Оптимизация ликвидности банка (Liquidity Management Optimization)**
**Описание:** Создать систему управления ликвидностью, которая оптимально распределяет денежные средства между различными инструментами (межбанк, ОФЗ, корпоративные облигации) с учетом сроков, доходности и рисков.
**Цель:** Обеспечить выполнение нормативов ликвидности при минимизации альтернативных издержек.

**Ожидаемые результаты:**

- Модель оптимизации портфеля ликвидности
- Алгоритм прогнозирования cash flow и стресс-сценариев
- Система рекомендаций по размещению временно свободных средств

**4. Оптимизация кредитного портфеля (Loan Portfolio Optimization)**
**Описание:** Разработать модель оптимизации структуры кредитного портфеля по отраслям, регионам и типам заемщиков с учетом риск-профиля, доходности и корреляции дефолтов.
**Цель:** Максимизировать риск-скорректированную доходность кредитного портфеля в рамках установленных лимитов.

**Ожидаемые результаты:**

- Модель оптимизации Markowitz для кредитного портфеля
- Алгоритм расчета оптимальных кредитных лимитов
- Инструмент для анализа концентрации рисков

**5. Оптимизация процессов колл-центра (Call Center Workforce Optimization)**
**Описание:** Создать систему оптимального распределения операторов колл-центра по сменам и навыкам на основе прогноза входящих звонков, сложности запросов и SLA.
**Цель:** Минимизировать затраты на персонал при обеспечении целевых показателей обслуживания.

**Ожидаемые результаты:**

- Модель оптимизации расписания операторов
- Алгоритм прогнозирования нагрузки в разрезе типов запросов
- Расчет экономии фонда оплаты труда

**6. Оптимизация кампаний кросс-продаж (Cross-Selling Campaign Optimization)**
**Описание:** Разработать систему оптимального планирования маркетинговых кампаний по кросс-продажам банковских продуктов с учетом клиентского профиля, истории взаимодействий и каналов коммуникации.
**Цель:** Максимизировать конверсию кросс-продаж при ограниченном маркетинговом бюджете.
**Ожидаемые результаты:**

- Модель оптимизации маркетингового микса
- Алгоритм персонализированных предложений
- Инструмент оценки эффективности кампаний

**7. Оптимизация процессинговой инфраструктуры (Payment Processing Optimization)**
**Описание:** Создать модель оптимального распределения транзакционной нагрузки между процессинговыми центрами с учетом их мощностей, задержек и стоимости обработки.
**Цель:** Минимизировать операционные затраты на процессинг при обеспечении SLA по времени обработки транзакций.

**Ожидаемые результаты:**

- Модель балансировки нагрузки между дата-центрами
- Алгоритм прогнозирования пиковых нагрузок
- Расчет оптимальной конфигурации процессинговой инфраструктуры

**8. Оптимизация валютной позиции банка (Foreign Exchange Position Optimization)**
**Описание:** Разработать систему управления валютной позицией банка, оптимально распределяющей активы и пассивы в различных валютах с учетом курсовых рисков и стоимости хеджирования.

**Цель:** Минимизировать валютные риски при заданных ограничениях по капиталу.

**Ожидаемые результаты:**

- Модель оптимизации валютной позиции
- Алгоритм определения оптимальных объемов хеджирования
- Система мониторинга валютных рисков в реальном времени

**9. Оптимизация залогового портфеля (Collateral Portfolio Optimization)**
**Описание:** Создать систему оптимального управления залоговым портфелем ипотечных и корпоративных кредитов с учетом ликвидности залогов, их стоимости и рисков обесценения.
**Цель:** Максимизировать качество залогового покрытия при минимизации затрат на управление.

**Ожидаемые результаты:**

- Модель оптимизации структуры залогового портфеля
- Алгоритм приоритезации залогов для регулярной переоценки
- Инструмент для стресс-тестирования залоговой базы

**10. Оптимизация IT-инфраструктуры банка (IT Infrastructure Optimization)**
**Описание:** Разработать модель оптимального распределения вычислительных ресурсов между бизнес-приложениями банка с учетом их критичности, нагрузки и требований к производительности.

**Цель:** Минимизировать затраты на IT-инфраструктуру при обеспечении бесперебойности бизнес-процессов.

**Ожидаемые результаты:**

- Модель оптимизации распределения вычислительных ресурсов
- Алгоритм автоматического масштабирования мощностей
- Расчет экономии на капитальных и операционных затратах

**Кейсы от «АВАЛАБ»**

**1. Оптимизация графика строительства жилого комплекса (Construction Schedule Optimization)**
**Описание:** Используя данные о последовательности работ, ресурсах, погодных условиях и логистике, необходимо разработать оптимальный календарный план строительства жилого комплекса. Учесть взаимозависимости работ, ограничения по подрядчикам и сезонные факторы.
**Цель:** Минимизировать сроки строительства при соблюдении бюджетных ограничений и качества работ.
**Ожидаемые результаты:**

- Модель оптимизации критического пути с учетом ресурсных ограничений
- Алгоритм динамической корректировки графика при сбоях
- Расчет экономии от сокращения сроков строительства

**2. Оптимизация закупки строительных материалов (Procurement Optimization)**
**Описание:** Разработать систему оптимального планирования закупок материалов с учетом колебания цен, логистики, условий хранения и графика строительства. Учесть риски срыва поставок и альтернативных поставщиков.
**Цель:** Минимизировать затраты на материалы при обеспечении бесперебойности строительства.
**Ожидаемые результаты:**

- Модель оптимизации объемов и сроков закупок
- Алгоритм выбора оптимальных поставщиков
- Система управления складскими запасами

**3. Оптимизация использования строительной техники (Equipment Utilization Optimization)**
**Описание:** Создать систему оптимального распределения строительной техники между объектами компании с учетом их характеристик, стоимости аренды/эксплуатации и требований проектов.
**Цель:** Максимизировать коэффициент использования техники при минимизации затрат.
**Ожидаемые результаты:**

- Модель оптимизации парка техники
- Алгоритм планирования ремонтов и обслуживания
- Расчет оптимального состава собственного и арендуемого парка

**4. Оптимизация проектных решений (Design Solutions Optimization)**

**Описание:** Разработать систему многокритериальной оптимизации проектных решений для жилых комплексов с учетом стоимости материалов, энергоэффективности, сроков реализации и рыночной привлекательности.
**Цель:** Найти оптимальные проектные решения, балансирующие между затратами и качеством.
**Ожидаемые результаты:**
- Модель оптимизации конструктивных решений
- Алгоритм выбора отделочных материалов
- Инструмент для оценки trade-off между различными параметрами

**5. Оптимизация продаж квартир (Apartment Sales Optimization)**
**Описание:** Создать систему оптимального управления продажами квартир в строящемся доме с учетом динамики спроса, ценовой эластичности и конкурентной среды.
**Цель:** Максимизировать выручку от продажи квартир в течение жизненного цикла проекта.
**Ожидаемые результаты:**
- Модель динамического ценообразования
- Алгоритм определения оптимального момента для запуска продаж
- Система управления скидками и акциями

**6. Оптимизация финансирования строительства (Construction Financing Optimization)**
**Описание:** Разработать модель оптимального структурирования финансирования строительных проектов с учетом стоимости капитала, графиков освоения средств и рисков.
**Цель:** Минимизировать стоимость финансирования при обеспечении ликвидности проекта.
**Ожидаемые результаты:**
- Модель оптимизации структуры финансирования
- Алгоритм управления денежными потоками
- Инструмент для сценарного анализа финансовых рисков

**7. Оптимизация управления подрядчиками (Contractor Management Optimization)**
**Описание:** Создать систему оптимального отбора и распределения подрядчиков по объектам с учетом их специализации, загрузки, репутации и стоимости услуг.
**Цель:** Минимизировать риски срывов сроков и качества работ при оптимизации затрат на подрядчиков.
**Ожидаемые результаты:**
- Модель оптимизации пула подрядчиков
- Алгоритм оценки надежности подрядчиков
- Система мониторинга выполнения обязательств

**8. Оптимизация земельного банка (Land Bank Optimization)**
**Описание:** Разработать модель оптимального управления земельным банком компании с учетом перспектив развития территорий, инфраструктурных проектов и рыночного спроса.
**Цель:** Максимизировать стоимость земельного банка и эффективность его использования.
**Ожидаемые результаты:**
- Модель оптимизации портфеля земельных участков
- Алгоритм определения приоритетов освоения
- Инструмент для оценки инвестиционной привлекательности участков

**9. Оптимизация эксплуатационных расходов (Operational Expenses Optimization)**
**Описание:** Создать систему оптимального управления эксплуатационными расходами сданных объектов с учетом энергоэффективности, сервисных затрат и требований жителей.
**Цель:** Минимизировать эксплуатационные расходы при сохранении качества обслуживания.
**Ожидаемые результаты:**
- Модель оптимизации затрат на содержание ЖК
- Алгоритм планирования ремонтов и обновлений
- Система мониторинга эффективности управляющей компании

**10. Оптимизация маркетингового бюджета (Marketing Budget Optimization)**
**Описание:** Разработать модель оптимального распределения маркетингового бюджета между каналами продвижения, проектами и этапами продаж с учетом их эффективности и целевой аудитории.

**Цель:** Максимизировать ROI маркетинговых инвестиций.
**Ожидаемые результаты:**
- Модель оптимизации медиамикса
- Алгоритм определения оптимального бюджета для каждого проекта
- Инструмент для атрибуции маркетинговых результатов

Каждый кейс содержит конкретную бизнес-задачу девелоперской компании, требующую применения интеллектуальных методов оптимизации, и четкие измеримые результаты для оценки эффективности решений.

## 5. Перечень учебной литературы, информационных ресурсов и технологий

### 5.1 Учебная литература


1. Поляков, В. М. Методы оптимизации : учебное пособие / В. М. Поляков, З. С. Агаларов. – 2-е изд. – Москва : Дашков и К°, 2022. – 86 с. : ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=697026 (дата обращения: 29.05.2024). – Библиогр. в кн. – ISBN 978-5-394-05003-9. – Текст : электронный.
2. Заозерская, Л. А. Методы оптимизации: целочисленное линейное программирование : учебное пособие : [16+] / Л. А. Заозерская, В. П. Ильев, Т. В. Леванова. – Омск : Омский государственный университет им. Ф.М. Достоевского (ОмГУ), 2020. – 40 с. : табл., схем. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=614055. – Библиогр.: с. 39. – ISBN 978-5-7779-2484-1. – Текст : электронный.
3. Сахарова, Л. В. Методы оптимизации для машинного обучения : учебное пособие : [16+] / Л. В. Сахарова, Г. В. Лукьянова ; Ростовский государственный экономический университет (РИНХ). – Ростов-на-Дону : Издательско-полиграфический комплекс РГЭУ (РИНХ), 2023. – 87 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=711220. – Библиогр. в кн. – ISBN 978-5-7972-3139-4. – Текст : электронный.
4. Омельяненко, Я. Эволюционные нейросети на языке Python : учебное пособие : [16+] / Я. Омельяненко ; пер. с англ. В. С. Яценкова. – Москва : ДМК Пресс, 2020. – 310 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=602179. – ISBN 978-5-97060-854-8. – Текст : электронный.
5. Карпенко, А. П. Современные алгоритмы поисковой оптимизации : алгоритмы, вдохновленные природой : учебное пособие / А. П. Карпенко. – 2-е изд. – Москва : МГТУ им. Н.Э. Баумана, 2017. – 448 с. : табл., граф., схем., ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=560390. – Библиогр.: с. 420-421. – ISBN 978-5-7038-4634-6. – Текст : электронный.
6. Аверченков, В. И. Основы математического моделирования технических систем : учебное пособие : [16+] / В. И. Аверченков, В. П. Федоров, М. Л. Хейфец. – 4-е изд., стер. – Москва : ФЛИНТА, 2021. – 271 с. : схем., ил., табл. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=93344. – Библиогр. в кн. – ISBN 978-5-9765-1278-8. – Текст : электронный.


### 5.2. Периодические издания:

1. Базы данных компании «Ист Вью» http://dlib.eastview.com
2. Электронная библиотека GREBENNIKON.RU https://grebennikon.ru/

### 5.3. Интернет-ресурсы, в том числе современные профессиональные базы данных и информационные справочные системы

**Электронно-библиотечные системы (ЭБС):**

1. ЭБС «ЮРАЙТ» https://urait.ru/

2. ЭБС «УНИВЕРСИТЕТСКАЯ БИБЛИОТЕКА ОНЛАЙН» www.biblioclub.ru
3. ЭБС «BOOK.ru» https://www.book.ru
4. ЭБС «ZNANIUM.COM» www.znanium.com
5. ЭБС «ЛАНЬ» https://e.lanbook.com

**Профессиональные базы данных:**

1. Scopus http://www.scopus.com/
2. ScienceDirect https://www.sciencedirect.com/
3. Журналы издательства Wiley https://onlinelibrary.wiley.com/
4. Научная электронная библиотека (НЭБ) http://www.elibrary.ru/
5. Полнотекстовые архивы ведущих западных научных журналов на Российской платформе научных журналов НЭИКОН http://archive.neicon.ru
6. Национальная электронная библиотека (доступ к Электронной библиотеке диссертаций Российской государственной библиотеки (РГБ) https://rusneb.ru/
7. Президентская библиотека им. Б.Н. Ельцина https://www.prlib.ru/
8. База данных CSD Кембриджского центра кристаллографических данных (CCDC) https://www.ccdc.cam.ac.uk/structures/
9. Springer Journals: https://link.springer.com/
10. Springer Journals Archive: https://link.springer.com/
11. Nature Journals: https://www.nature.com/
12. Springer Nature Protocols and Methods: https://experiments.springernature.com/sources/springer-protocols
13. Springer Materials: http://materials.springer.com/
14. Nano Database: https://nano.nature.com/
15. Springer eBooks (i.e. 2020 eBook collections): https://link.springer.com/
16. "Лекториум ТВ" http://www.lektorium.tv/
17. Университетская информационная система РОССИЯ http://uisrussia.msu.ru

**Информационные справочные системы:**

1. Консультант Плюс - справочная правовая система (доступ по локальной сети с компьютеров библиотеки)

**Ресурсы свободного доступа:**

1. КиберЛенинка http://cyberleninka.ru/;
2. Американская патентная база данных http://www.uspto.gov/patft/
3. Министерство науки и высшего образования Российской Федерации https://www.minobrnauki.gov.ru/;
4. Федеральный портал "Российское образование" http://www.edu.ru/;
5. Информационная система "Единое окно доступа к образовательным ресурсам" http://window.edu.ru/;
6. Единая коллекция цифровых образовательных ресурсов http://school-collection.edu.ru/ .
7. Проект Государственного института русского языка имени А.С. Пушкина "Образование на русском" https://pushkininstitute.ru/;
8. Справочно-информационный портал "Русский язык" http://gramota.ru/;
9. Служба тематических толковых словарей http://www.glossary.ru/;
10. Словари и энциклопедии http://dic.academic.ru/;
11. Образовательный портал "Учеба" http://www.ucheba.com/;
12. Законопроект "Об образовании в Российской Федерации". Вопросы и ответы http://xn--273--84d1f.xn--p1ai/voprosy_i_otvety

**Собственные электронные образовательные и информационные ресурсы КубГУ:**

1. Электронный каталог Научной библиотеки КубГУ http://megapro.kubsu.ru/MegaPro/Web

2. Электронная библиотека трудов ученых КубГУ http://megapro.kubsu.ru/MegaPro/UserEntry?Action=ToDb&idb=6
3. Среда модульного динамического обучения http://moodle.kubsu.ru
4. База учебных планов, учебно-методических комплексов, публикаций и конференций http://infoneeds.kubsu.ru/
5. Библиотека информационных ресурсов кафедры информационных образовательных технологий http://mschool.kubsu.ru;
6. Электронный архив документов КубГУ http://docspace.kubsu.ru/
7. Электронные образовательные ресурсы кафедры информационных систем и технологий в образовании КубГУ и научно-методического журнала "ШКОЛЬНЫЕ ГОДЫ" http://icdau.kubsu.ru/

### 5.4 **Публикации конференций A***

1. Junhyuk Oh, Matteo Hessel, Wojciech M. Czarnecki, Zhongwen Xu, Hado P van Hasselt, Satinder P. Singh, David Silver. Discovering Reinforcement Learning Algorithms. Advances in Neural Information Processing Systems 33 (NeurIPS 2020) URL: https://papers.nips.cc/paper_files/paper/2020/file/0b96d81f0494fde5428c7aea243c9157-Paper.pdf

## 6. Методические указания для обучающихся по освоению дисциплины (модуля)

По курсу предусмотрено проведение лекционных занятий, на которых дается основной систематизированный материал, лабораторных работ, экзамена.
Оценка успеваемости осуществляется по результатам:
- выполнения лабораторных работ;
- ответов на теоретические вопросы при сдаче лабораторных работ;
- ответа на экзамене (для выявления знания и понимания теоретического материала дисциплины).

Одним из этапов курса является самостоятельная работа по дисциплине с использованием указанных литературных источников и методических указаний автора курса.
Виды и формы СР, сроки выполнения, формы контроля приведены выше в данном документе.
Для лучшего освоения дисциплины при защите ЛР студент должен ответить на несколько вопросов из лекционной части курса.
Процедура промежуточной аттестации проходит в соответствии с Положением о текущем контроле и промежуточной аттестации обучающихся ФГБОУ ВО «КубГУ».
Итоговой формой контроля сформированности компетенций у обучающихся по дисциплине является экзамен. Студенты обязаны сдать экзамен в соответствии с расписанием и учебным планом.
ФОС промежуточной аттестации состоит из вопросов к экзамену, задач по дисциплине и результатов текущего контроля.
Экзамен по дисциплине преследует цель оценить работу студента за курс, получение теоретических знаний, их прочность, развитие творческого мышления, приобретение навыков самостоятельной работы, умение применять полученные знания для решения практических задач.
Форма проведения экзамена: устно.
Экзаменатору предоставляется право задавать студентам дополнительные вопросы по всей учебной программе дисциплины.
Результат сдачи экзамена заноситься преподавателем в экзаменационную ведомость и зачетную книжку.

Оценивание уровня освоения дисциплины основывается на качестве выполнения студентом заданий текущего контроля и ответов на вопросы экзамена.
В освоении дисциплины инвалидами и лицами с ограниченными возможностями здоровья большое значение имеет индивидуальная учебная работа (консультации) – дополнительное разъяснение учебного материала.
Индивидуальные консультации по предмету являются важным фактором, способствующим индивидуализации обучения и установлению воспитательного контакта между преподавателем и обучающимся инвалидом или лицом с ограниченными возможностями здоровья.

## 7. Материально-техническое обеспечение по дисциплине (модулю)

| Наименование специальных помещений | Оснащенность специальных помещений | Перечень лицензионного программного обеспечения |
|---|---|---|
| Учебная аудитория для проведения лекционных занятий, аудитория текущего контроля и промежуточной аттестации (ауд. 129, А305, А307). | Мебель: учебная мебель<br>Технические средства обучения: проектор, экран/телевизор, компьютер/ноутбук с соответствующим программным обеспечением (ПО) | Программы для демонстрации и создания презентаций |
| Учебная аудитория для проведения занятий семинарского типа, учебная аудитория для проведения лабораторных занятий, аудитория текущего контроля и промежуточной аттестации, учебная аудитория для проведения индивидуальных и групповых консультаций (ауд. 147, 148, 149). | Мебель: учебная мебель<br>Технические средства обучения: экран/телевизор, компьютер/ноутбук | Программы для демонстрации и создания презентаций |
| Компьютерный класс, учебная аудитория для проведения лабораторных занятий, учебная аудитория для проведения курсового проектирования (выполнения курсовых работ) (ауд. 101-107). | Мебель: учебная мебель<br>Технические средства обучения: экран/телевизор, проектор, компьютер/ноутбук.<br>Оборудование: компьютерная техника с выходом в Интернет, доступом в электронную информационно-образовательную среду | системы программирования на языках высокого уровня, сетевой доступ к ресурсам, в частности C++, Python и пр. с возможностью многопользовательской работы, виртуальные машины* |

Для самостоятельной работы обучающихся предусмотрены помещения, укомплектованные специализированной мебелью, оснащенные компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду университета.

| Наименование помещений для самостоятельной работы обучающихся | Оснащенность помещений для самостоятельной работы обучающихся | Перечень лицензионного программного обеспечения |
|---|---|---|
| Помещение для самостоятельной работы обучающихся (читальный зал Научной библиотеки) | Мебель: учебная мебель<br>Комплект специализированной мебели: компьютерные столы<br>Оборудование: компьютерная техника с подключением к информационно-коммуникационной сети «Интернет» и доступом в электронную информационно-образовательную среду образовательной организации, веб-камеры, коммуникационное оборудование, обеспечивающее доступ к сети интернет (проводное | Доступ к печатным и электронным информационным ресурсам |

| | | |
|---|---|---|
| | соединение и беспроводное соединение по технологии Wi-Fi) | |
| Помещение для самостоятельной работы обучающихся (ауд. 146 ) | Мебель: учебная мебель<br>Комплект специализированной мебели: компьютерные столы<br>Оборудование: компьютерная техника с подключением к информационно-коммуникационной сети «Интернет» и доступом в электронную информационно-образовательную среду образовательной организации, веб-камеры, коммуникационное оборудование, обеспечивающее доступ к сети интернет (проводное соединение и беспроводное соединение по технологии Wi-Fi) | Системы программирования на языках C++, Python с возможностью многопользовательской работы, виртуальные машины* |

*Виртуальные машины и ресурсы GPU в облаке предоставляется индустриальным партнером ПАО «Сбербанк»:

| № | Продукт | Параметры продукта | Кол-во | Кол-во конфигураций | Ед. изм. |
|---|---|---|---|---|---|
| 1 | Виртуальная машина | Виртуальная машина 10% vCPU 2 vCPU 4 RAM | 1 | 60 | Шт |
| | | ОС Ubuntu 22.04 | 1 | | Шт |
| | | Системный диск SSD | 1 | | Шт |
| | | | 10 | | Гб |
| | | Аренда публичного IP | 1 | | Шт |
| 2 | Виртуальная машина с GPU | Виртуальная машина с GPU NVIDIA® Tesla® V100 2 GPU 8 vCPU 128 ГБ RAM | 1 | 1 | Шт |
| | | ОС Ubuntu_24.04 | 1 | | Шт |
| | | Системный диск SSD | 1 | | Шт |
| | | | 2000 | | Гб |
| | | Диск SSD | 1 | | Шт |
| | | | 4096 | | Гб |
| | | Диск SSD | 1 | | Шт |
| | | | 4096 | | Гб |
| | | Аренда публичного IP | 1 | | Шт |

Дополнительные облачные ресурсы предоставляются технологическим партнером Yandex Cloud.