МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Вятский государственный университет» Факультет автоматики и вычислительной техники Кафедра электронных вычислительных машин

Отчет
по производственной практике
на ООО «Синаптик», г. Киров

Выполнил студент группы ИВТб-4301_____/Жеребцов К. А./

Руководитель практики от ВятГУ_____/Клюкин В. Л/

Руководитель практики от предприятия_____/Клюкин И. А./

Киров 2024

График прохождения практики

Номер п/п	Перечень заданий, которые подлежат выполнению в ходе практики	Сроки выполнения
1.	Пройти инструктаж по ознакомлению с правилами внутреннего трудового распорядка, охраны труда, техники безопасности, противопожарной безопасности, санитарно-эпидемиологическими правилами и гигиеническими нормативами, а также вводный инструктаж и инструктаж на рабочем месте	29.004.2024
2.	Анализ предметной области и разработка ТЗ	30.04.2024 - 10.05.2024
3.	Разработка структуры	11.05.2024 – 19.05.2024
4.	Программная реализация	20.05.2024 – 25.05.2024
5.	Подготовить отчёт по производственной практике	26.05.2024 – 27.05.2024

Введение

Производственная практика является неотъемлемой и важной частью учебного процесса. В ходе неё не только закрепляются теоретические знания, полученные во время обучения, но и отрабатываются необходимые практические умения для эффективной работы в профессиональной сфере, осваивается инструментарий для работы и взаимодействия в команде, оттачиваются коммуникативные навыки для общения с коллегами.

1 Общие сведения о предприятии

Предприятие ООО «Синаптик» действует с 14 февраля 2013 года.

Физический адрес: Российская Федерация, Кировская область, город Киров, улица Карла Маркса, 18А.

Организационно правовая форма предприятия – Общество с ограниченной ответственностью.

Основным видом деятельности компании является разработка компьютерного программного обеспечения, консультационные услуги в данной области и другие сопутствующие услуги.

2 Описание выполненной работы

2.1 Анализ предметной области

2.1.1 Распознавание номеров автомобилей

Система распознавания номерных знаков на парковке состоит из нескольких основных компонентов, каждый из которых играет важную роль в обеспечении точности и эффективности процесса распознавания.

Камеры видеонаблюдения: Ключевым компонентом являются камеры, установленные на парковочных местах или вокруг парковки. Они захватывают видеопоток, содержащий изображения автомобилей и их номерных знаков. Камеры должны быть размещены таким образом, чтобы обеспечить максимальное покрытие парковочной зоны и достаточное разрешение для четкого изображения номерных знаков.

Алгоритмы обработки изображений: Полученные с камер изображения подвергаются обработке с использованием специальных алгоритмов компьютерного зрения. Эти алгоритмы предназначены для выделения и

извлечения номерных знаков из изображений, фильтрации шумов и искажений, а также подготовки данных для дальнейшего анализа и распознавания.

Методы машинного обучения: Для обучения системы распознавания номерных знаков используются методы машинного обучения, включая классические алгоритмы обучения с учителем и глубокие нейронные сети. Модели обучаются на больших объемах данных, содержащих изображения номерных знаков, чтобы достичь высокой точности и надежности распознавания.

Системы хранения и обработки данных: Полученная информация о номерах автомобилей и времени прибытия сохраняется и обрабатывается с использованием специализированных систем хранения и обработки данных. Это позволяет эффективно управлять информацией о парковочных местах, предоставлять отчеты о использовании парковки и обеспечивать необходимую функциональность для пользователей системы.

2.1.2 Сбор данных

Сбор данных - это процесс измерения и сбора информации о желаемых переменных таким образом, чтобы можно было находить вопросы, связанные с данными, и использовать их в исследованиях различных типов. Сбор данных является общей чертой обучения по различным дисциплинам, таким как маркетинг, статистика, экономика, естественные науки и т.д. Методы сбора данных могут варьироваться в зависимости от предмета, но конечная цель исследования и честность при сборе данных имеют одинаковое значение во всех вопросах обучения.

В зависимости от характера сбора данных их можно разделить на два основных типа, а именно:

- Первичный метод сбора данных
- Вторичный метод сбора данных.

Первичные данные собираются исследователями самостоятельно и впервые в рамках исследования. Существуют различные способы сбора первичных данных, некоторые из которых следующие:

- Интервью: Интервью являются наиболее часто используемым методом первичного сбора данных. При проведении интервью для сбора данных используется анкета или исследователь может задавать вопросы непосредственно интервьюируемому. Идея заключается в поиске информации по волнующим темам из ответов респондента. Используемые анкеты можно отправить по электронной почте или уточнить подробности в ходе телефонного интервью.
- Метод Дельфи: В этом методе исследователь запрашивает информацию у группы экспертов. Исследователь может выбрать очное исследование или отправить анкеты по электронной почте. В конце применения метода Дельфи все данные собираются в соответствии с потребностями исследования.
- Проективные методы: Проективные методы используются в исследованиях, которые являются частными или конфиденциальными таким образом, что исследователь считает, что респонденты не раскроют информацию, если будут заданы прямые вопросы. Существует множество типов проективных методик, таких как тематические тесты оценки (TAT), ролевые игры, завершение мультфильма, словесная ассоциация и завершение предложения.
- Интервью в фокус-группе: Здесь собираются несколько человек, чтобы обсудить насущную проблему. В таких интервью обычно участвует от шести до двенадцати человек. Каждый участник выражает свое собственное мнение, и принимается коллективное единогласное решение.
- Метод анкетирования: Здесь используется вопросник для сбора данных от различных групп населения. Для соответствующего исследования используется набор вопросов, и респонденты отвечают на вопросы, прямо или

косвенно связанные с вопросником. Этот метод может быть, как открытым, так и закрытым.

2.1.3 Определение интересов

Понимание текущих тенденций на рынке автомобилей и изменений в потребительских предпочтениях является достаточно важным.

В современном автомобильном секторе наблюдается постоянное развитие и изменение, под влиянием различных факторов, включая технологические инновации, экологические требования и социокультурные изменения. Одним из ключевых аспектов этого развития является постоянное изменение потребительских предпочтений в отношении моделей автомобилей и их характеристик.

В последние годы отмечается растущий интерес к экологически чистым технологиям, таким как электромобили и гибридные автомобили, вследствие повышенного внимания к экологической устойчивости и сокращению выбросов вредных веществ. Помимо этого, наблюдается увеличение спроса на автомобили с продвинутыми технологиями безопасности и удобства, такими как системы автопилота, умные системы информации и развлечений, а также функции связи и дистанционного управления.

Важно также отметить, что предпочтения в выборе автомобиля могут сильно различаться в зависимости от возрастной группы потребителей. Например, молодые водители могут проявлять больший интерес к стильным и технологичным моделям, в то время как старшие покупатели могут уделять большее внимание комфорту, безопасности и эффективности использования.

Таким образом, понимание текущих тенденций рынка автомобилей и предпочтений потребителей в различных возрастных группах является важным аспектом проекта, поскольку это позволяет точнее выявить интересы

владельцев автомобилей и предложить им более релевантные услуги и продукты.

2.2 Постановка расширенного технического задания.

2.2.1 Основание для разработки

Программа разрабатывается на основе учебного плана кафедры “Электронные вычислительные машины” по направлению 09.03.01

2.2.2 Цель и задача

Название темы: «Разработка системы оценки и прогнозирования интересов клиентов автостоянки».

Целью дипломного проекта является автоматизация процесса сбора данных об интересах посетителей парковки и их прогнозирования.

Задача – проектирование и разработка системы для сбора информации об интересах и их прогнозирования на базе искусственной нейронной сети.

2.2.3 Наименование и область применения

Наименование разрабатываемого программного обеспечения: системы оценки и прогнозирования интересов клиентов автостоянки.

Область применения: прогнозирование интересов посетителей парковки для улучшения сервиса.

2.2.4 Требование к программе

Программа должна обеспечивать возможность выполнения следующих функций:

- определять регистрационный номер автомобиля;

- определять модель автомобиля
- анализировать бланки опросов определенного формата;
- формирование набора данных в файл типа CSV;
- обучение нейронной сети по сформированному датасету;
- предсказывать интересы посетителей с использование обученной модели.

Требования к нейронной сети

Нейронная сеть должна быть реализована на базе языка программирования Python с использованием библиотеки Tensorflow, Keras.

2.2.5 Требование к программной документации

Состав программной документации должен включать в себя:

- техническое задание;
- пояснительная записка, содержащая описание разработки;
- руководство пользователя.

Разрабатываемые программные модули должны быть самодокументированы, т.е. тексты программ должны содержать все необходимые комментарии.

2.2.6 Технико-экономические характеристики

Реализованное программное обеспечение использует бесплатную модель распространения и разрабатывается с помощью бесплатных решений.

2.3 Разработка структуры системы

Разрабатываемая система состоит из 3 модулей, каждый из которых может работать обособленно, при наличии необходимых ресурсов:

- модуль для сбора информации;
- модуль для обучения нейросети и предсказания интересов.

Необходимо сформировать наиболее точное описание разрабатываемого программного обеспечения. Для этого было принято решение о рассмотрении функциональной диаграммы верхнего уровня.

В данном случае в качестве отображения взаимосвязей была выбрана нотация IDEF0. В качестве входных данные для сбора информации (бланк опроса, изображение автомобиля с номером) и данные для предсказания интересов (возраст владельца и модель авто). В качестве субъекта выступает пользователь и вычислительная машина. Управление задается алгоритмами для сбора данных, алгоритм обучения нейросети, алгоритм работы с обученной моделью. К выходным данным относятся датасет, обученная модель и предсказанные интересы.

Контекстная диаграмма IDEF0 представлена на рисунке 1.

Рисунок 1 – Контекстная диаграмма IDEF0

На детализирующей функциональной диаграмме показаны следующие этапы:

- сбор информации;
- предсказание интересов.

Детализированная контекстная диаграмма представлена на рисунке 2.

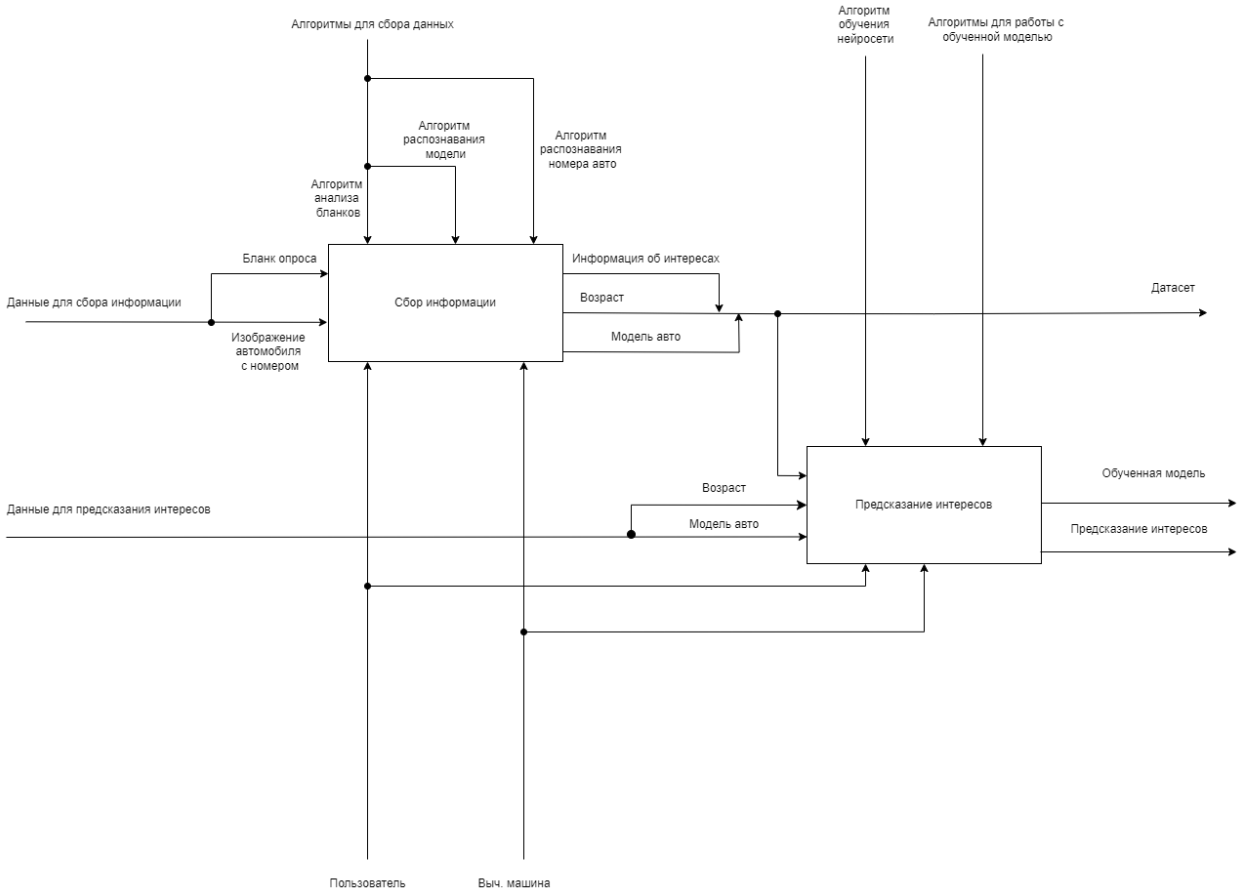

Рисунок 2 – Детализированная контекстная диаграмма IDEF0

2.4 Разработка структуры компонента для сбора информации

2.4.1 Общая структура

Разрабатываемый модуль будет состоять из трех основных компонентов: «Анализ бланков», «Определение модели», «Распознавание номера». Структура представлена на рисунке 3.

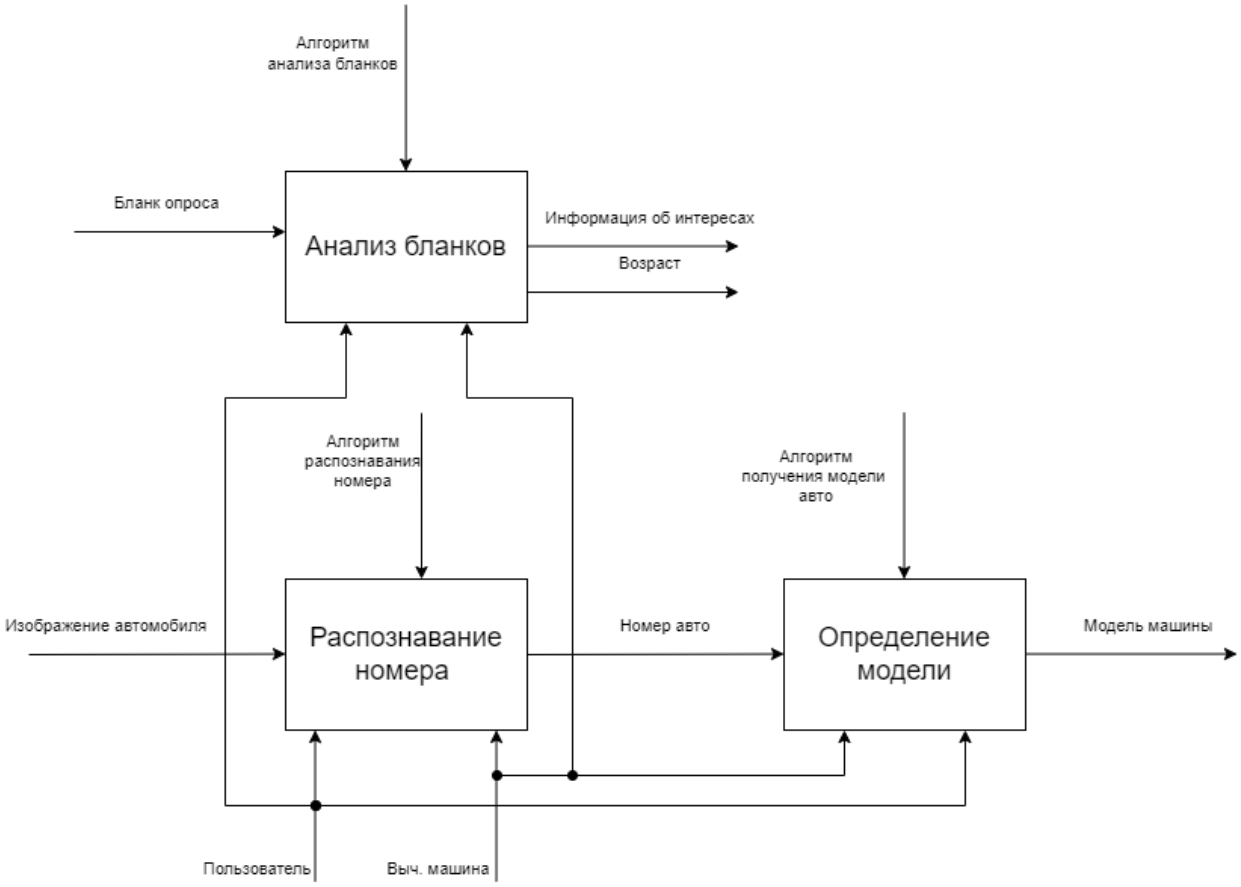

Рисунок 3 – Структура модуля

Так же стоит определить формат создаваемого датасета. В него будет входить вся получаемая информация. Следовательно, датасет будет иметь следующие поля:

- model: модель автомобиля;
- age: возраст;
- art: интерес к искусству;
- sport: интерес к спорту;
- book/film: интерес к фильмам и книгам;
- science: интерес к науке и технологиям;
- travel: интерес к путешествиям;
- cooking: интерес к готовке;
- politics: интерес к политике.

2.4.2 Анализатор бланков

Основными функциями компонента являются функции для анализа анкет, сбора информации об ответах и возрасте, указанном на бланке.

Задача компонента заключается в анализе анкет. Он должен автоматически обрабатывать изображения анкет, извлекать информацию об ответах на вопросы и возрасте, указанном на бланке.

Для выполнения задачи выбраны следующие инструменты:

- OpenCV: для обработки изображений, выделения контуров и преобразований перспективы.
- imutils: для удобной работы с изображениями, сортировки контуров и других операций.
- pytesseract: для распознавания текста на изображениях с использованием технологии OCR.

На вход компоненту подается изображение анкеты. Это может быть фотография сверху или скан копия бланка. После своей работы результаты будут сохраняются в CSV-файл.

Описание алгоритма:

- Начало: Вход в компонент.
- Загрузка изображения: Компонент получает на вход изображение анкеты для анализа.
- Препроцессинг изображения:
  - Преобразование в оттенки серого.
  - Сглаживание с помощью фильтра Гаусса.
  - Применение оператора Canny для обнаружения границ.
- Поиск области анкеты:
  - Поиск контуров на изображении.
  - Определение контура анкеты.
- Выделение области анкеты:

- Применение преобразования перспективы для выделения области анкеты.
- Получение и сохранение выделенной области.

- Обработка анкеты:
  - Применение пороговой обработки к выделенной области для получения четкого изображения.
  - Обнаружение контуров на обработанном изображении.
  - Определение контуров блоков с ответами на анкете.
- Анализ ответов:
  - Идентификация блоков с ответами на анкете.
  - Определение заполненных ответов в каждом блоке.
  - Оценка правильности ответов согласно ключу ответов.
  - Формирование списка результатов анализа.
- Извлечение возраста:
  - Определение области, содержащей возраст на анкете.
  - Применение преобразования перспективы для выделения области с возрастом.
  - Применение пороговой обработки и распознавание текста для извлечения возраста.
- Запись результатов:
  - Сохранение результатов анализа и возраста в CSV файл.
- Конец: Завершение работы компонента.

2.4.3 Поиск модели авто

Функциональные требования: компонент должен обеспечивать ввод номера, отправку запроса на сайт Autoteka, получение результатов и извлечение необходимой информации.

Выбор инструментов и технологий:

- Для автоматизации взаимодействия с веб-страницами можно использовать библиотеку Selenium для Python, так как она обладает мощными возможностями веб-автоматизации.
- Для парсинга полученных данных удобно применять инструменты для работы с текстом и регулярные выражения, например, встроенные функции Python.

Составление алгоритма работы:

- Компонент должен открывать веб-браузер, переходить на сайт Autoteka и вводить указанный регистрационный номер.
- После получения результата, необходимо обработать страницу с результатами и извлечь нужные данные, такие как модель автомобиля.
- Важно учесть возможные сценарии ошибок и способы их обработки, например, отсутствие данных по указанному номеру или проблемы с доступом к сайту Autoteka.

2.4.4 Распознавание номера

Для обнаружения положения номерного знака используется предобученная модель model_resnet.tflite с архитектурой ResNet-50. Она предназначена для выявления границ номерного знака и выделение его области. ResNet-50 выбрана благодаря ее способности эффективно извлекать признаки из изображений и обучаться на больших объемах данных.

50-уровневая архитектура ResNet включает в себя следующие элементы:

- **Свертка ядра 7 × 7** наряду с 64 другими ядрами с шагом в 2 размера.
- **Максимальный уровень объединения** с шагом в 2 размера.
- **9 слоев**- свертка ядра размером 3 × 3,64, другой с 1 × 1,64 ядрами и третий с 1 × 1256 ядрами. Эти 3 слоя повторяются 3 раза.
- **12 слоев** с 1 × 1128 ядрами, 3 × 3128 ядрами и 1 × 1512 ядрами, повторяется 4 раза.

- **18 слоев** с ядрами 1 × 1256 и 2 ядрами 3 × 3,256 и 1 × 1,1024, повторяется 6 раз.
- **9 слоев** с ядрами 1 × 1512, 3 × 3512 и 1 × 12048 повторяются 3 раза.
- **Создание среднего пула**, за которым следует полностью подключенный уровень с 1000 узлами, с использованием функции активации softmax.

Для определения номера автомобиля используется предобученная модель model_nomer.tflite. Данная модель предназначена для определения текста на номерном знаке. фрагмент изображения с номерным знаком поступает от выше описанной модели. Модель представляет собой комбинацю сверточной нейронной сети (CNN), рекуррентной нейронной сети долгой краткосрочной памяти (LSTM) и функции свертки по времени (CTC) для распознавания текста на изображениях. CNN используется для извлечения признаков из изображения номерного знака, LSTM преобразует эти признаки в последовательность символов, а CTC используется для выравнивания последовательности символов с фактическим текстом номера.

К функциям компонента относятся следующие пункты:

- Загрузка и разбиение входного видео на кадры
- Выбор отдельного кадра
- Распознавание номера автомобиля на получившимся изображении

2.5 Разработка структуры компонента для обучения нейросети и предсказания интересов

2.5.1 Общая структура

разрабатываемый модуль будет состоять из двух основных компонентов: «Обучения нейросети», «Определение интересов». структура представлена на рисунке 4.

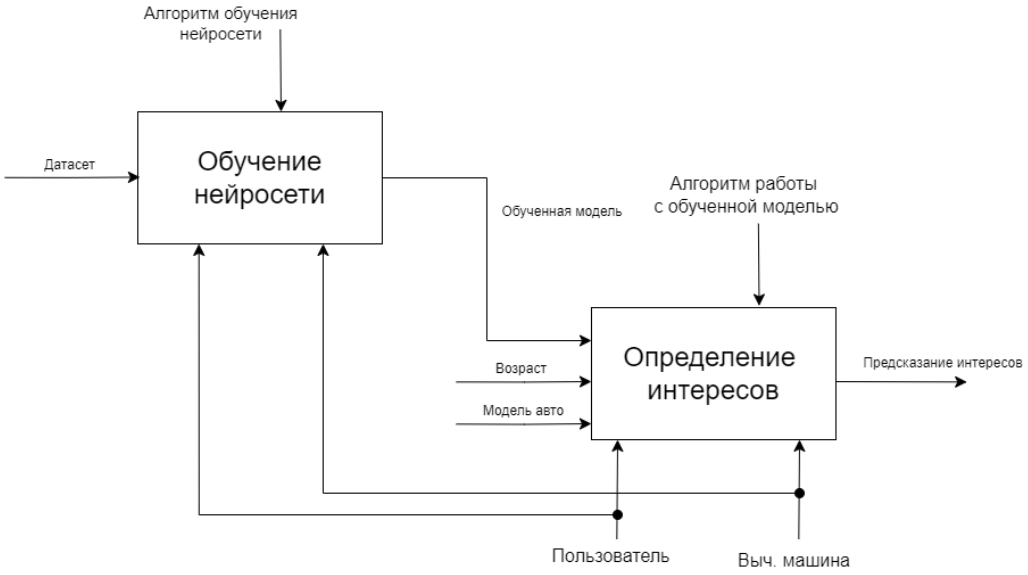

Рисунок 4 – Структура модуля

2.5.2 Алгоритм обучения нейросети

Обобщенный алгоритм обучения нейронной сети:

1) Загрузка данных: Начинаем с загрузки данных из источника, чаще всего из файлов CSV, баз данных или других источников данных.
2) Предварительная обработка данных: Перед тем как данные попадут в модель, их необходимо подготовить. Это может включать в себя удаление или заполнение отсутствующих значений, преобразование категориальных переменных в числовые (например, с помощью кодирования One-Hot), нормализацию числовых данных и т.д.
3) Разделение данных: Для оценки модели данные обычно разделяются на обучающий и тестовый наборы. Обучающий набор используется для обучения модели, а тестовый - для оценки ее производительности.
4) Определение архитектуры модели: Решается, какая будет архитектура нейронной сети: сколько слоев и скрытых нейронов в каждом слое, какие функции активации использовать, будет ли применяться метод регуляризации (например, Dropout), и т.д.

5) Определение функции потерь и оптимизатора: Выбирается функция потерь (например, кросс-энтропия для классификации) и оптимизатор (например, стохастический градиентный спуск, Adam и т.д.), которые будут использоваться в процессе обучения.
6) Обучение модели: Модель обучается на обучающем наборе данных. Обычно это включает в себя несколько эпох обучения, где каждая эпоха представляет собой один проход по всем обучающим данным.
7) Оценка производительности модели: После завершения обучения модель оценивается на тестовом наборе данных, чтобы определить ее производительность и обобщающую способность.
8) Настройка гиперпараметров: Для улучшения производительности модели может быть выполнен подбор оптимальных гиперпараметров, таких как количество слоев, количество нейронов, скорость обучения и т.д.
9) Выбор лучшей модели: Выбирается модель с лучшей производительностью на основе метрик оценки, таких как точность (accuracy), F1-мера, и т.д.
10) Интеграция с обратной связью: Для улучшения обучения и производительности модели могут быть применены различные методы обратной связи, такие как ранняя остановка (Early Stopping), адаптивная скорость обучения и т.д.

2.5.3 Описание структуры нейронной сети

Для получения множества моделей с различной архитектурой используют специальные библиотеки-тюнеры, которые по заданным гиперпараметрам для каждого варианта обучают получившуюся сеть, после чего можно выбрать архитектуру сети, показавшую наибольшую точность.

Были выделены следующие параметры, которые учитывались при создании структуры нейронной сети:

- num_hidden_layers – число скрытых слоев
- num_neurons – число нейронов в слоях
- activation – функция активации
- dropout_rate – значение слоя Dropout
- optimizer – определяет оптимизатор
- learning_rate – скорость обучения
- epochs – количество поколений

Разработка нейронной сети и аугментация данных производилась на языке Python с использованием библиотеки Keras. Для автоматического подбора параметров использовалась библиотека GridSearchCV.

В таблице 1 представлен результат лучшей модели.

Таблица 1 – Параметры модели

Точность	Ошибка
0,896	0,117

На рисунке 5 представлена структура лучшей модели.

Рисунок 5 – Структура нейронной сети

2.6 Программная реализация компонента для сбора информации

2.6.1 Анализ бланков

В коде компонента используются библиотеки «cv2» (OpenCV) для обработки изображений, «numpy» для работы с массивами, «argparse» для обработки аргументов командной строки, «imutils» для удобных функций обработки изображений, «csv» для записи результатов в CSV-файл и «pytesseract» для распознавания текста на изображении.

Сам компонент имеет следующие основные функции:

- process_exam(image_path):

Функция принимает путь к изображению экзаменационного листа.

Используя библиотеку «cv2», выполняет предварительную обработку изображения

- get_age(image_path):

Функция принимает путь к изображению документа, содержащего возраст. Выполняет распознавание возраста на бланке.

- «write_to_csv(filename, results)»:

Функция выполняет запись одной строки в датасет.

- «blank(path_in, path_out, model)»:
  - Функция принимает путь к входному изображению, путь к выходному CSV-файлу и модель (номер экзаменационного листа).
  - Вызывает функции «process_exam» и «get_age» для обработки изображения и определения возраста.
  - Добавляет модель и возраст в начало списка результатов.
  - Вызывает функцию «write_to_csv» для записи результатов в CSV-файл.
- «main()»:
  - Основная функция, которая принимает путь к входному изображению и путь к выходному CSV-файлу с помощью аргументов командной строки.
  - Вызывает функции «process_exam» и «get_age» для обработки изображения и определения возраста.
  - Вызывает функцию «write_to_csv» для записи результатов в CSV-файл.

2.6.2 Поиск модели авто

При программной реализации данного компонента использовалась библиотеку Selenium для автоматизации веб-браузера. Она включает в себя следующие функции и шаги:

- Импортируются необходимые модули: selenium, time, и другие модули, необходимые для работы с Selenium.
- Указывается путь к драйверу браузера (chromedriver.exe) в переменной «driver_path».
- Указывается URL-адрес веб-сайта, на котором будет выполняться поиск, в переменной «website_url».
- Указывается номер, который будет использоваться в качестве вводного параметра, в переменной «num».
- Создается экземпляр браузера Chrome с использованием указанного драйвера в переменной «driver».
- Определяется функция «site», которая принимает в качестве аргумента номер и выполняет следующие действия:
    - Загружается веб-страница с указанным URL-адресом.
    - На странице вводится номер в поле с именем "identifier".
    - Номер отправляется в поле ввода с помощью метода «send_keys» и клавиши Enter.
    - Ожидается изменение URL-адреса на веб-странице с использованием «WebDriverWait».
    - Ожидается 3 секунды для загрузки страницы.
    - На странице находится элемент с классом "pit4K", который содержит текст, связанный с номером.
    - Текст из элемента извлекается и разбивается на строки с помощью «split('\n')».

- Результирующий текст объединяется в две строки с помощью «join(lines[:2])».
- Возвращается первая строка текста, разделенная запятой и приводится к нижнему регистру.
- Завершается работа браузера с помощью «driver.quit()».

- В блоке «if __name__ == "__main__":» вызывается функция «site» с передачей номера в качестве аргумента и выводится результат в консоль.

2.6.3 Распознавание номера

Функции компонента:

- `load()`:
  - Функция загружает видеофайл из диалогового окна выбора файлов.
  - Использует метод `QFileDialog.getOpenFileName()` для открытия диалогового окна выбора файлов.
  - Путь к выбранному видеофайлу сохраняется в переменной `video_file` в формате `Path(video_file[0])`.
- `split()`:
  - Функция разделяет видеофайл на отдельные кадры.
  - Создает папку с именем, соответствующим имени исходного видеофайла, если такой папки еще нет.
  - Использует библиотеку `moviepy` для разделения видео на кадры с заданной частотой кадров (SAVING_FRAMES_PER_SECOND).
  - Каждый кадр сохраняется в папке с именем, соответствующим времени его появления в видео.
- `choose()`:
  - Функция загружает изображение с номерным знаком из диалогового окна выбора файлов.

- Использует метод `QFileDialog.getOpenFileName()` для открытия диалогового окна выбора файлов.
- Путь к выбранному изображению сохраняется в переменной `image_file`.
- Изображение отображается на метке `label_6` с помощью метода `setPixmap()`.

- `carplate_text()`:
  - Функция распознает номер автомобиля на изображении.
  - Использует модели машинного обучения для обнаружения номерного знака и его распознавания.
  - Производит предварительную обработку изображения, включая изменение размера, преобразование в оттенки серого, применение фильтра Canny для обнаружения границ и преобразование изображения в бинарное.
  - Использует модель TensorFlow Lite для распознавания символов на номерном знаке.
  - Возвращает распознанный номер автомобиля в виде строки.
- `recognize()`:
  - Функция вызывается при нажатии на кнопку "Распознать".
  - Вызывает функцию `carplate_text()` для распознавания номера автомобиля на выбранном изображении.
  - Отображает результат распознавания на метке `label_7`.
  - Обновляет статистику по количеству распознанных номеров и их вероятности.
  - Проверяет формат распознанного номера и выводит сообщение об ошибке, если формат неверный.
  - Если формат верный, сохраняет результат в массив `arr` и обновляет статистику на метках `label_2` и `label_5`.

- Передает полученный номер в компонент определения модели машины

2.7 Программная реализация компонента для обучения нейросети и работы с обученной моделью

2.7.1 Обучение нейросети

Теперь более подробно рассмотрим процесс реализации:

1) Загрузка данных: Из CSV-файла 'datanew.csv' данные загружаются в объект DataFrame библиотеки pandas.
   Входными параметрами являются поля Model и Age, а остальные поля выходными.
2) Предобработка данных: Категориальный столбец "Model" преобразуется в числовые значения с помощью метода OneHotEncoding из библиотеки sklearn. Нормализация числовых данных о возрасте также выполняется для лучшей работы нейронной сети.
3) Разделение данных: Данные разделяются на обучающий и тестовый наборы с использованием функции train_test_split из библиотеки sklearn.
4) Определение функции создания модели: Функция create_model определяет архитектуру нейронной сети с возможностью настройки различных параметров, таких как количество слоев, количество нейронов, функция активации, dropout rate, оптимизатор и т.д.
5) Создание модели KerasClassifier: Создается модель KerasClassifier, обертка над моделью Keras, для использования с методом кросс-валидации GridSearchCV из библиотеки sklearn.
6) Задание сетки параметров для подбора: Задаются параметры для поиска лучших комбинаций параметров модели с помощью GridSearchCV.

7) Создание объекта GridSearchCV: Создается объект GridSearchCV с указанными параметрами для поиска оптимальных гиперпараметров модели.
8) Определение обратного вызова EarlyStopping: Обратный вызов EarlyStopping определяется для автоматической остановки обучения, если происходит переобучение модели.
9) Подгонка объекта GridSearchCV к данным: Объект GridSearchCV подгоняется к обучающим данным с использованием метода fit, включая обратный вызов EarlyStopping.
10) Вывод лучших параметров: Выводятся лучшие параметры модели, найденные с помощью GridSearchCV.
11) Оценка модели с лучшими параметрами: Лучшая модель из GridSearchCV оценивается на тестовом наборе данных, и выводится точность предсказания.

2.7.2 Реализация интерфейса для работы с моделью

При реализации модуля были написаны 2 скрипта. Один для выполнения предсказаний, другой реализует интерфейс.

Рассмотрим первый скрипт более подробно:

1) Загрузка обученной модели: В начале модуля загружается обученная модель нейронной сети из файла, созданная и сохраненная в этапе обучения.
2) Преобразование новых данных: Функция preprocess_new_data принимает новые данные (модель и возраст) и преобразует их в формат, совместимый с обученной моделью. В частности, она преобразует категориальный столбец "Model" в числовые значения с помощью OneHotEncoding и нормализует числовые данные о возрасте.
3) Получение предсказаний: Функция get_predictions принимает преобразованные данные и использует обученную модель для

выполнения предсказаний. Она возвращает вероятности принадлежности к каждой из категорий интересов.

4) Вывод результатов: Функция print_results выводит результаты предсказаний в консоль для отладки.
5) Главная функция main: Главная функция main принимает модель и возраст, вызывает описанные выше функции для выполнения предсказаний и форматирует результаты для дальнейшего использования. Она возвращает массив вероятностей принадлежности к каждой категории интересов.
6) Тестирование модуля: В блоке if __name__ == "__main__": пример вызова функции main с тестовыми данными и вывод результатов предсказаний в консоль для отладки.

Теперь рассмотрим скрипт, который реализует интерфейс:

1) Определение графического интерфейса: Создается класс MyWindow, который наследуется от QWidget и представляет собой окно приложения. В методе initUI определяется интерфейс окна, включая надписи, поля ввода, таблицу для отображения результатов и кнопку для выполнения предсказаний.
2) Обработка событий: В методе proc происходит обработка события нажатия на кнопку. Данные из полей ввода для модели и возраста извлекаются с помощью метода toPlainText(). Если оба поля не пустые, вызывается функция main(), описанная в предыдущем скрипте, для выполнения предсказаний. Результаты предсказаний отображаются в таблице.
3) Запуск приложения: В блоке if __name__ == '__main__': создается экземпляр приложения QApplication, создается окно MyWindow, показывается на экране и запускается выполнение приложения с помощью sys.exit(app.exec()).