МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»
Кафедра дискретной математики и информационных технологий

РАЗРАБОТКА ВЕБ-ПРИЛОЖЕНИЯ ДЛЯ АНАЛИЗА ТРАНСПОРТНЫХ СРЕДСТВ ПО ИЗОБРАЖЕНИЯМ С ИСПОЛЬЗОВАНИЕМ НЕЙРОННЫХ СЕТЕЙ

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 421 группы
направления 09.03.01 — Информатика и вычислительная техника
факультета КНиИТ
Щелкина Ильи Сергеевича

Научный руководитель
старший преподаватель _____ А. А. Трунов

Заведующий кафедрой
доцент, к. ф.-м. н. _____ Л. Б. Тяпаев

Саратов 2025

ВВЕДЕНИЕ

Классификация транспортных средств является важной задачей в области автоматизации процессов управления дорожным движением, обеспечения безопасности на дорогах и мониторинга транспортных потоков. Современные технологии позволяют автоматизировать этот процесс, используя методы компьютерного зрения и искусственного интеллекта. Одним из наиболее перспективных направлений в этой сфере является применение нейронных сетей, способных эффективно распознавать и классифицировать объекты на изображениях.

Использование нейронных сетей делает возможным создание автоматизированных систем, которые могут работать в реальном времени и обрабатывать большие объёмы данных. В частности, задача классификации транспортных средств актуальна для фильтрации данных, что важно для выявления скрытых государственных регистрационных знаков.

Таким образом, разработка веб-приложения для классификации типов транспортных средств с использованием нейронных сетей представляет собой актуальную задачу, имеющую практическое значение в различных областях, включая транспортную инфраструктуру, безопасность дорожного движения и интеллектуальные транспортные системы.

Основная цель выпускной квалификационной работы – разработка веб-приложения, использующего нейронные сети для анализа изображений транспортных средств и предоставляющего результаты анализа, в частности, категорию и модель транспортного средства.

Для достижения цели были поставлены следующие задачи:

1. Провести анализ существующих методов детектирования и классификации транспортных средств, включая традиционные подходы и современные нейронные сети.
2. Разработать архитектуру веб-приложения, включающую клиентскую и серверную части, обеспечивающую взаимодействие пользователя с системой.
3. Выбрать архитектуры и обучить нейронные сети, подходящие для задачи детектирования и классификации транспортных средств.
4. Интегрировать нейронные сети в веб-приложение и обеспечить их корректное функционирование в рамках системы.

В работе представлены 3 главы:

1. Методы и подходы к классификации транспортных средств.
2. Обучение нейросетевых моделей.
3. Разработка и реализация веб-приложения.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

В первом разделе содержится обзор основных техник, применяемых для решения задачи распознавания и классификации автомобилей, мотоциклов и других ТС на изображениях. Раздел начинается с анализа классических подходов, в котором подробно рассматриваются этапы предобработки изображений (удаление шумов, выравнивание контраста), детектирование краёв с помощью операторов Канни и Собеля, а также алгоритмы сегментации на основе пороговой фильтрации и метода водораздела. Выделение контуров и геометрических признаков (площадь, соотношение сторон, форма) описано как традиционный путь к распознаванию ТС, однако отмечена слабая устойчивость таких методов к вариациям освещённости и положению камер.

Далее обсуждаются методы, опирающиеся на ручное извлечение признаков: гистограммы ориентированных градиентов, ключевые точки SIFT и SURF, локальные бинарные шаблоны LBP и цветовые гистограммы. Эти дескрипторы позволяют формировать векторные представления объектов, после чего задействуются классические классификаторы (SVM, деревья решений, KNN). Несмотря на относительную простоту и интерпретируемость, такие подходы ограничены необходимостью тонкой настройки для каждого нового сценария и плохо справляются с шумами и сложным фоном.

Отдельное внимание уделяется алгоритмам обработки изображений: фильтрации и сегментации. Описаны методы линейной и адаптивной фильтрации, пороговые техники (Оцу, Саувола) и алгоритмы Watershed и k-means, позволяющие разбивать кадр на однородные области. Показано, что классические фильтры и морфологические операции работают быстро, но не учитывают контекст сцены и требуют ручной подгонки порогов, что снижает их надёжность в условиях реального видеопотока.

Следом рассматривается машинное обучение с ручными признаками: текстурных и контурных дескрипторах, применяемых для обучения моделей классификации. Подчёркнута интерпретируемость таких систем и возможность работы с небольшими выборками, однако отмечена их низкая адаптивность к изменениям условий и ограниченная способность захватывать сложные нелинейные зависимости.

Существенный прогресс в области распознавания объектов обеспечили нейросетевые подходы, где свёрточные архитектуры (CNN) представлены как

средство автоматического извлечения иерархических признаков. Описаны основные принципы работы свёрток и слоёв подвыборки, подчёркнута роль больших размеченных датасетов и настройки гиперпараметров. Показано, что CNN значительно превосходят классические методы по точности и устойчивости, но требуют значительных вычислительных ресурсов и объёма данных для обучения.

Следующим обсуждается глубокое обучение в целом: многослойные нейросети позволяют автоматически выделять сложные нелинейные зависимости, обеспечивают масштабируемость на больших массивах данных и демонстрируют высокую обобщающую способность. Отмечены ключевые преимущества — автоматическое извлечение признаков, устойчивость к шумам и способность учитывать контекст изображения — и обозначены основные ограничения, связанные с риском переобучения и необходимостью регуляризации.

В завершение рассматриваются архитектурные решения, наиболее часто используемые в задачах детектирования и классификации транспортных средств. Рассмотрены ResNet с резидуальными блоками, позволяющими обучать очень глубокие сети без затухания градиентов, и семейство YOLO, объединяющее локализацию и классификацию в единой модели для работы в режиме реального времени. Приведено сравнение их производительности и точности на задачах обнаружения автомобилей, грузовиков и других объектов дорожной инфраструктуры.

Таким образом, раздел даёт полное методологическое основание для дальнейшей части обучения модели, формируя понимание сильных и слабых сторон каждого подхода и определяя базовые принципы, которыми будет руководствоваться автор при разработке собственной системы классификации транспортных средств.

Во втором разделе подробно излагаются этапы построения, настройки и проверки эффективности двух взаимосвязанных подсистем: одностадийного детектора категорий транспортных средств на основе архитектуры YOLO и сверточного классификатора моделей ТС на базе ResNet-50.

Сначала обосновывается выбор архитектур. Для задачи детектирования категорий транспортных средств была применена модель YOLO версии 11s из библиотеки Ultralytics. Одностадийная природа этого детектора позволяет одновременно предсказывать ограничивающие рамки и классы объектов за один

проход по кадру, что обеспечивает требуемую для систем мониторинга скорость обработки. На основании сопоставления со второстадийными подходами и классическими каскадными методами Хаара демонстрируется, что YOLO достигает более высокой mAP при существенно меньшей задержке инференса. Вторая подсистема, предназначенная для тонкой классификации моделей автомобилей, реализована на предобученной ResNet-50, в которой конечный полносвязный слой заменён таким образом, чтобы число выходных нейронов соответствовало числу классов в датасете Stanford Cars. Выбор ResNet-50 обоснован оптимальным соотношением глубины сети и числа параметров, что позволяет достичь высокой точности при приемлемой скорости инференса на аппаратуре со средними вычислительными ресурсами.

Далее раздел посвящён подготовке и структуре обучающих выборок. Для детектора категорий использовался открытый набор Vehicle Dataset for YOLO объёмом 3000 изображений, размеченных по шести классам транспортных средств. Изображения и аннотации организованы в директории с разделением на обучающую (70%), валидационную (30%) и тестовую (по 5 образцов на класс из валидации) подвыборки. Для классификатора моделей применялся Stanford Cars с 16185 изображениями по 196 категориям; данные были разбиты почти поровну на обучение и тест. Перед обучением рассчитывались средние и стандартные отклонения каналов RGB, что позволило выполнить нормализацию входных данных согласно общепринятым формулам.

Этап обучения детектора включал настройку входного разрешения 640×640, размера набора изображений 16 и числа эпох 100, а также комплексную систему аугментаций, объединявшую мозаичную сборку, случайное отражение и цветовые преобразования. В процессе обучения сохранялись весовые коэффициенты модели с наилучшими значениями критерия mAP на валидации. На этапе обучения классификатора ResNet-50 использовался оптимизатор Adam, размер набора изображений 64 и 50 эпох. Аугментации включали небольшие повороты, отражения и масштабирование с сохранением пропорций; для контроля переобучения применялись регуляризация и ранняя остановка по качеству на валидационной выборке.

В заключительной части раздела приводятся ключевые результаты. Для детектора продемонстрирована устойчивая сходимость всех компонентов функции потерь (box_loss, cls_loss, dfl_loss) и высокие значения Precision ($\approx 0{,}98$) и

Recall (≈0,95), а средняя точность mAP@0.5 достигла 0,98 и для mAP@[0.5:0.95] – 0,92. Классификатор ResNet-50 показал снижение функции потерь на обучении до уровня ≈0,05 и стабилизацию на валидации в диапазоне 0,85–0,95, что свидетельствует о начальных признаках переобучения; точность на обучении превысила 98%, тогда как на валидации составила около 80%. В качестве итоговой модели были выбраны весовые параметры 10-й эпохи, обеспечившие оптимальный компромисс между качеством обобщения и степенью соответствия тренировочным данным.

В третьем разделе описывается разработка и реализация веб-приложения, первая её подглава раскрывает архитектурные принципы построения системы. В основе лежит классическая трёхуровневая клиент-серверная модель, в которой клиентский слой (Presentation Tier) реализован на React и отвечает за отображение данных и первичную валидацию пользовательского ввода. Серверный уровень (Application Tier) построен с использованием FastAPI и содержит всю логику приложения, включая модули нейросетевого анализа изображений и REST-контроллеры для приёма и обработки запросов. Уровень хранения данных (Data Tier) реализован на PostgreSQL и обеспечивает надёжное сохранение информации о пользователях, результатах анализа и фильтраций изображений транспортных средств.

Взаимодействие между клиентом и сервером осуществляется по протоколам HTTP/HTTPS в стиле REST. Каждый ресурс приложения имеет свой URI и управляется стандартными методами (GET, POST, PUT, DELETE), что обеспечивает предсказуемость интерфейса, независимость от платформы и простоту интеграции. Сервер не сохраняет состояние сессий – вся необходимая для обработки информация передаётся в каждом запросе, что упрощает горизонтальное масштабирование и балансировку нагрузки.

При поступлении на сервер JSON-запроса FastAPI разбирает и валидирует поступившие данные, при необходимости обращается к базе данных и передаёт на вход нейросетевому конвейеру. По завершении обработки формируется ответ в формате JSON, возвращаемый клиенту для визуализации. Такая организация потока данных минимизирует объём передаваемой информации и позволяет гибко масштабировать каждый из трёх уровней приложения без влияния на остальные.

Второй подраздел третьего раздела посвящён реализации клиентской ча-

сти веб-приложения. Для обеспечения надёжности и удобства поддержки был выбран TypeScript, который, сохраняя полную совместимость с экосистемой JavaScript, вводит статическую типизацию и позволяет выявлять типовые и логические ошибки ещё на этапе компиляции. Пользовательский интерфейс построен на React: компонентный подход и виртуальное DOM обеспечивают декларативное описание представления, оптимизируют перерисовки при обновлении состояния и упрощают масштабирование приложения за счёт переиспользования независимых компонентов.

Маршрутизация внутри одностраничного приложения организована с помощью «react-router-dom», что даёт возможность определять иерархию маршрутов, динамически подставлять параметры в URL и реализовывать защищённые страницы (например, административную панель) на основании состояния авторизации. Для ускорения верстки и унификации визуального стиля используется Ant Design: готовый набор адаптивных компонентов (формы, таблицы, кнопки, меню) с поддержкой темизации и единообразным дизайном позволил сократить объём кода и сосредоточиться на логике приложения.

Обмен данными с сервером организован через Axios. В отличие от встроенного fetch, Axios автоматически сериализует JSON, поддерживает перехватчики запросов и ответов, отмену запросов и централизованную обработку ошибок, что упрощает реализацию надёжного слоя коммуникации. Все запросы к API обёрнуты в модели AnalyseModel и FilterModel, настроенные на базовые пути «/api/analyse» и «/api/filter» соответственно, и добавляют к заголовкам JWT-токен для авторизации.

Структура клиентской части разделена на общие компоненты – Layout (обёртка страницы с Header и Footer), Header (динамическое меню на основе состояния пользователя и Drawer для мобильной версии) и Footer (контактные данные и копирайт) – и на страницы основных сценариев: начальную (MainPage), анализ изображений (AnalysePage), фильтрацию (FilterPage), аутентификацию/регистрацию (AuthPage) и административную панель (AdminPage). Каждая страница реализована как самостоятельный React-компонент с использованием хуков для управления состоянием и эффектами и обеспечивает полный цикл взаимодействия: от загрузки изображений через компоненты Dragger и Upload до визуализации результатов анализа или фильтрации с поддержкой вставки из буфера обмена, отображением галереи или таблиц и

обработкой ошибок.

Серверная часть веб-приложения реализована с использованием фреймворка FastAPI, обеспечивающего высокую производительность и удобство разработки. Она выполняет роль связующего звена между пользовательским интерфейсом и модулями обработки изображений, в том числе нейронными сетями для детекции и классификации транспортных средств.

Ключевой задачей серверной части является приём изображений от клиента, их предобработка, а также маршрутизация к соответствующим модулям анализа. Сервер обрабатывает входные данные, делегирует задачи нейронным сетям и возвращает пользователю структурированные результаты. Таким образом, обеспечивается централизованная логика взаимодействия между клиентом, хранилищем данных и средствами интеллектуального анализа.

Архитектура построена модульно, что упрощает масштабирование и сопровождение системы. Каждый компонент реализован в виде отдельного контроллера, выполняющего строго определённые функции. В частности, реализованы контроллеры для загрузки изображений, запуска процессов детекции и классификации, а также для получения результатов анализа.

Модель обработки изображений разделена на два этапа. На первом этапе используется свёрточная нейросеть YOLO, предназначенная для локализации и первичной классификации транспортных средств по категориям (например, легковой автомобиль, грузовик, автобус и т.п.). Полученные прямоугольные области передаются на второй этап, где применяется другая нейронная сеть — модифицированная ResNet, обученная на задаче определения точной модели транспортного средства. Этот этап анализа осуществляется по обрезанному изображению из прямоугольной области, выделенной первой моделью.

Кроме нейросетевых компонентов, серверная часть использует сторонние библиотеки, такие как Pillow для обработки изображений, а также SQLAlchemy для взаимодействия с базой данных. Для сериализации и валидации входных и выходных данных применяется библиотека Pydantic, позволяющая точно определять схемы запросов и ответов.

Дополнительно реализована система логирования, обеспечивающая отслеживание ключевых этапов обработки изображений и выявление возможных сбоев. Это особенно важно в условиях параллельной обработки большого числа запросов.

Таким образом, серверная часть обеспечивает устойчивую и масштабируемую основу для функционирования всей системы анализа транспортных средств, объединяя в себе традиционные подходы веб-разработки и современные методы машинного зрения.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы было разработано и реализовано веб-приложение для детектирования и классификации транспортных средств на изображениях, использующее нейронные сети. В качестве нейросетевых моделей были выбраны и обучены свёрточные нейронные сети: Ultralytics YOLO11s для детектирования и классификации категории транспортного средства и ResNet-50 для точного распознавания конкретных моделей транспортных средств. Внедрение такого комбинированного подхода позволило добиться высокой точности распознавания: mAP@0.5 примерно равен 0.98 для детектирования и классификации категории транспортного средства и до 0.95 по валидационной выборке для классификатора модели транспортного средства.

Разработанная архитектура веб-приложения обеспечивает интуитивно понятный пользовательский интерфейс, где каждая страница и компонент логически связаны и реализуют чёткую навигацию по функциональности. Приложение предоставляет возможность загрузки и анализа изображения для предоставления результатов анализа, в частности, категории и модели транспортного средства. А также возможность фильтрации набора изображений по заданным настройкам параметров фильтра, а именно по набору категорий и моделей транспортных средств. Также пользователи с правами администратора могут просматривать таблицы с пользователями веб-приложения, результатами анализов и фильтраций изображений.

Модели нейросетей продемонстрировали устойчивую сходимость функций потерь и стабильность метрик качества на валидационных наборах. Сочетание методов детектирования и классификации показало высокую надёжность в разных условиях съёмки и обеспечило корректную обработку как крупных, так и мелких объектов.

Основные источники информации:

1. Хусаинов Р. М., Талипов Н. Г., Катасёв А. С., Шалаева Д. В. Интеллектуальная система анализа транспортных потоков в автоматизированных системах управления дорожным движением // Программные продукты и системы. 2024. №1. – 69–76 с.
2. Сацюк А. В., Белый Р. В., Ищенко А. Е. Оценка эффективности алгоритмов YOLO для обнаружения объектов в реальном времени во встраиваемых

системах беспилотных транспортных средств // Сборник научных трудов ДОНИЖТ. 2024. №4 (75). – 73–82 с.
3. Ван Ц., Чэнь С., Чжан Ю., Ши Д., Чэнь Ц. Алгоритм определения городских транспортных целей с применением метода мультимодального слияния на БПЛА // ИВД. 2024. №9 (117). – 51 с.
4. Дели А. Г. Классификация видео контента с помощью сверточных нейронных сетей // E-Scio. 2022. №5 (68). – 120–130 с.
5. Бычков И. В., Ружников Г. М., Федоров Р. К., Попова А. К., Авраменко Ю. В. О классификации космических снимков Sentinel-2 нейронной сетью ResNet-50 // КО. 2023. №3. – 474–481 с.
6. Архитектура информационных систем: учебное пособие / сост. И. В. Беляева. – Ульяновск : УлГТУ, 2019. – 192 с.
7. Амундсен, М. RESTful Web API: паттерны и практики / М. Амундсен. – СПб: Sprint Book, 2025. – 464 с.