МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

**«Вятский государственный университет»**

**(ФГБОУ ВПО «ВятГУ»)**

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

Допущено к защите

Руководитель проекта

\_\_\_\_\_\_\_\_\_\_\_\_ (Караваева О. В.)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2023г.

«**Разработка программного модуля для распознавания государственного номера автомобиля»**

Пояснительная записка курсового проекта по дисциплине

«Курс знаний бакалавра»

ТПЖА 090301.387 ПЗ

Разработал студент группы ИВТ-31 \_\_\_\_\_\_\_\_\_\_\_\_/Жеребцов К.А./

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_/Караваева О. В./

Консультант \_\_\_\_\_\_\_\_\_\_/Долженкова М.Л./

Работа защищена с оценкой «\_\_\_\_\_\_\_\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(оценка) (дата)*

Члены комиссии \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

(подпись)

Киров 2023

**Реферат**

Жеребцов К. А. Разработка программного модуля «Разработка программного модуля для распознавания государственного номера автомобиля». ТПЖА. 090301.387 ПЗ: Курс. проект / ВятГУ, каф. ЭВМ; рук. Караваева О. В. - Киров, 2023. Графическая часть 3 л. – ф. А2, ПЗ 62 с, 22 рис., 4 табл., 7 источников, 5 приложений.

Разработка программного модуля «Разработка программного модуля для распознавания государственного номера автомобиля».

Объект исследования и разработки – программный модуль для распознавания государственного номера автомобиля

Цель курсового проекта – разработка рабочего прототипа программного модуля.

Результат работы – готовый прототип программного модуля.

# Содержание

[Содержание 3](#_Toc158659340)

[Введение 5](#_Toc158659341)

[1 Анализ предметной области 6](#_Toc158659342)

[1.1 Обоснование актуальности темы 6](#_Toc158659343)

[1.2 Обзор аналогов 6](#_Toc158659344)

[1.3 Основные компоненты системы распознавания номерных знаков 7](#_Toc158659345)

[1.4 Применение распознавания номерных знаков 8](#_Toc158659346)

[1.5 Технические проблемы и вызовы 9](#_Toc158659347)

[1.5 Обзор систем распознавания номерных знаков 10](#_Toc158659348)

[1.5.1 «Автомаршал» 10](#_Toc158659349)

[1.5.2 "Spetslab-Traffic" 13](#_Toc158659350)

[1.5.3 "SecurOS Auto" 15](#_Toc158659351)

[1.6 Выводы по разделу 21](#_Toc158659352)

[2. Постановка задачи 21](#_Toc158659353)

[2.1 Цели и задачи проекта 21](#_Toc158659354)

[2.2 Требования к программному модулю 22](#_Toc158659355)

[2.3 Выводы по разделу 22](#_Toc158659356)

[3 Выбор метода распознавания 22](#_Toc158659357)

[3.1 Методы распознавания номерных знаков 22](#_Toc158659358)

[3.2 Метод Виолы-Джонса 23](#_Toc158659359)

[3.3 Машины опорных векторов 24](#_Toc158659360)

[3.3 Использование искусственных нейронных сетей 26](#_Toc158659361)

[3.4 Обоснование выбора метода 30](#_Toc158659362)

[3.5 Выводы по разделу 32](#_Toc158659363)

[4 Описание выбранных моделей 33](#_Toc158659364)

[4.1 Обнаружение положения номера 33](#_Toc158659365)

[4.2 Определение номера автомобиля 35](#_Toc158659366)

[4.2 Определение номера автомобиля 37](#_Toc158659367)

[5 Программная реализация 38](#_Toc158659368)

[5.1 Программные средства 38](#_Toc158659369)

[5.2 Разработка 39](#_Toc158659370)

[5.3 Результат работы программы 43](#_Toc158659371)

[5.4 Выводы по разделу 45](#_Toc158659372)

[6 Оценка качества 46](#_Toc158659373)

[6.1 Тестирование 46](#_Toc158659374)

[Заключение 48](#_Toc158659375)

[Список литературы 50](#_Toc158659376)

[Приложение А 51](#_Toc158659377)

[Приложение Б 52](#_Toc158659378)

[Приложение В 54](#_Toc158659379)

[Приложение Г 55](#_Toc158659380)

[Приложение Д 56](#_Toc158659381)

# Введение

Распознавание номерных знаков - это важная область компьютерного зрения, направленная на автоматизацию процесса идентификации автомобилей по их номерным знакам. С развитием технологий компьютерного зрения и машинного обучения системы распознавания стали более точными и эффективными. Они находят применение в различных сферах, включая безопасность дорожного движения, управление транспортным потоком, контроль доступа и автоматизацию парковочных систем. Автоматическое распознавание номерных знаков помогает улучшить безопасность на дорогах, облегчить процессы контроля и управления, а также повысить эффективность работы в различных сферах, связанных с автомобильным транспортом.

В данном курсовом проекте мы рассмотрим основные аспекты создания и функционал программного модуля для распознавания номерных знаков. Кроме того, мы проанализируем технические аспекты его разработки.

# 1 Анализ предметной области

В данном разделе рассматриваются обоснование актуальности выбранной темы и аналоги, представленные существующим программным обеспечением.

## 1.1 Обоснование актуальности темы

Распознавание номерных знаков на парковке является крайне актуальной задачей в современном мире, где управление парковочными ресурсами становится все более важным в условиях растущего числа автомобилей и дефицита парковочных мест. Эффективное управление парковочными ресурсами не только обеспечивает удобство для водителей, но и позволяет оптимизировать использование доступного пространства, сокращая проблемы с парковкой и улучшая общую мобильность в городах.

Системы распознавания номерных знаков на парковке позволяют автоматизировать процесс контроля и управления парковочным пространством. Они могут использоваться для определения свободных парковочных мест, контроля за временем стоянки автомобилей, выявления нарушений правил парковки и обеспечения безопасности на парковочных территориях.

Таким образом, разработка и внедрение систем распознавания номерных знаков на парковке имеет большое практическое значение, помогая оптимизировать использование парковочных ресурсов, улучшать общественную инфраструктуру и обеспечивать удобство для пользователей.

## 1.2 Обзор аналогов

История развития технологий распознавания номерных знаков на парковке отражает постоянное стремление к улучшению эффективности и точности систем управления парковочными ресурсами. Начиная с простых систем ручного контроля и ввода данных, мы перешли к использованию современных методов компьютерного зрения и машинного обучения.

Первые системы контроля парковки включали в себя ручной ввод данных о номерных знаках автомобилей для контроля времени стоянки и учета нарушений. Однако этот метод был трудоемким и неэффективным.

В последующие десятилетия произошел значительный прогресс в области компьютерного зрения и обработки изображений, что привело к разработке автоматизированных систем распознавания номерных знаков. Алгоритмы компьютерного зрения стали способными распознавать номерные знаки на изображениях с высокой точностью, а внедрение методов машинного обучения, включая нейронные сети, позволило повысить уровень автоматизации и точности распознавания.

Современные системы распознавания номерных знаков на парковке основаны на использовании компьютерного зрения, алгоритмов обработки изображений и методов машинного обучения. Они способны автоматически обнаруживать номерные знаки на видеопотоке с камер парковки, извлекать информацию о номере автомобиля и его времени прибытия, а также предоставлять данные об использовании парковочных мест для управления парковочной инфраструктурой и предоставления услуг пользователям.

## 1.3 Основные компоненты системы распознавания номерных знаков

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.

## 1.4 Применение распознавания номерных знаков

Системы распознавания номерных знаков на парковке имеют широкий спектр применений, направленных на оптимизацию управления парковочными ресурсами и повышение удобства для пользователей. Некоторые из основных применений включают:

Определение свободных парковочных мест: Системы распознавания номерных знаков позволяют автоматически определять занятость и свободные парковочные места на парковке. Это позволяет водителям экономить время на поиске свободного места и повышает эффективность использования парковочных ресурсов.

Контроль времени стоянки: Путем анализа времени прибытия и ухода автомобилей с парковки системы распознавания номерных знаков могут обеспечивать контроль за временем стоянки. Это полезно для управления парковочными лимитами и тарифами, а также для предотвращения нарушений правил парковки.

Предоставление отчетов и аналитики: Данные, собранные системами распознавания номерных знаков, могут использоваться для создания отчетов и аналитики об использовании парковки. Это может включать в себя информацию о загрузке парковки, популярности различных времен и дней недели, а также статистику об использовании определенных зон парковки.

Обеспечение безопасности на парковке: Распознавание номерных знаков также может использоваться для обеспечения безопасности на парковке путем обнаружения и регистрации автомобилей, въезжающих и выезжающих с территории парковки. Это помогает предотвратить кражи и другие неправомерные действия.

Интеграция с другими системами: Системы распознавания номерных знаков могут интегрироваться с другими системами управления парковкой, такими как системы оплаты парковки и мобильные приложения для бронирования мест. Это создает единое централизованное управление парковочной инфраструктурой и повышает удобство для пользователей.

## 1.5 Технические проблемы и вызовы

Несмотря на значительные преимущества, системы распознавания номерных знаков на парковке сталкиваются с рядом технических проблем и вызовов, которые могут затруднить их эффективное функционирование.

Сложные условия освещения и погоды: Изменчивость условий освещения и внешние атмосферные условия, такие как дождь, снег или туман, могут значительно влиять на качество изображений, получаемых с камер. Это может затруднить процесс распознавания номерных знаков и повлиять на общую эффективность системы.

Разнообразие форм и размеров номерных знаков: Номерные знаки могут различаться по форме, размеру, цвету и стилю шрифта в зависимости от страны и региона. Это создает сложности для разработки универсальных алгоритмов распознавания, способных работать с разнообразными типами номерных знаков.

Сложности в разрешении изображений: Низкое разрешение изображений или дальнее расположение автомобилей от камеры также могут усложнить процесс распознавания номерных знаков. Это требует использования более продвинутых методов обработки изображений и улучшенных алгоритмов распознавания.

Обработка больших объемов данных: Системы распознавания номерных знаков на парковке должны быть способны обрабатывать большие объемы видеоданных в реальном времени. Это требует высокой вычислительной мощности и оптимизированных алгоритмов для обеспечения быстрой и точной обработки данных.

Приватность и защита данных: Сбор, хранение и обработка персональных данных о номерах автомобилей вызывает вопросы приватности и защиты данных. Необходимо обеспечить соответствие законодательству о защите персональных данных и предпринять меры по обеспечению безопасности и конфиденциальности данных пользователей парковки.

## 1.5 Обзор систем распознавания номерных знаков

## 1.5.1 «Автомаршал»

Автомаршал — это система, предназначенная для распознавания номерных знаков. Он используется для контроля проезда автомобилей, проверки их по разным базам данных и ограничения доступа автомобилей на территорию [5] (рис. 1.3).



Рисунок 1 – Автомаршал

Область применения:

* пункты пропуска предприятий;
* платные стоянки;
* парковка возле торговых центров;
* закрытые зоны;
* жилые комплексы;
* коттеджные поселки;
* автомойки;
* автоматический поиск автомобиля.

Возможности:

* выявление транспортных средств в зоне контроля и определение направлений их движения;
* признание государственных регистрационных номеров автомобилей;
* ведение базы проезжающих транспортных средств;
* контроль шлагбаумов и светофоров в автоматическом и автоматизированном режимах при организации проезда транспортных средств на закрытые территории (предприятия, автостоянки, ТСЖ, коттеджные поселки и т.п.);
* поддержка баз данных пользователей на транспортных средствах: тип «Клиенты», «VIP», «Разыскивается», «Черный список» и т.д.
* поиск распознанных номеров автомобилей в базах пользовательских данных и, если они совпадают, автоматический ответ по сценарию, определенному пользователем;
* передача данных распознавания в посторонние учетные системы;
* подготовка, печать, рассылка по SMS и электронной почте различных отчетов, формируемых как автоматически, так и по желанию пользователя;
* осуществление контроля за перемещением транспортных средств на территории предприятия и контроль за соблюдением установленных регламентов такого перемещения;
* дифференциация доступа различных категорий пользователей к сгенерированным данным по паролям или картам;

Методы и технологии распознавания

Распознавание номерных знаков осуществляется с помощью ядра распознавания RecarKernel (разработка научно-производственной компании Mullen). В системе распознавания номерных знаков

«Автомаршал» для максимально надежной работы в различных условиях (день, ночь, осадки, яркое солнце, фары) использует два параллельно работающих алгоритма. Первый алгоритм базируется на нейронных сетях, второй алгоритм классификатор на основе многомерной интерполяции и аппроксимации [5].

## 1.5.2 "Spetslab-Traffic"

"SL-Traffic" — программный модуль для считывания и распознавания номерных знаков в режиме реального времени. Модуль работает как видеофильтр в системе или может быть интегрирован в любую систему видеозаписи, в том числе посторонних.

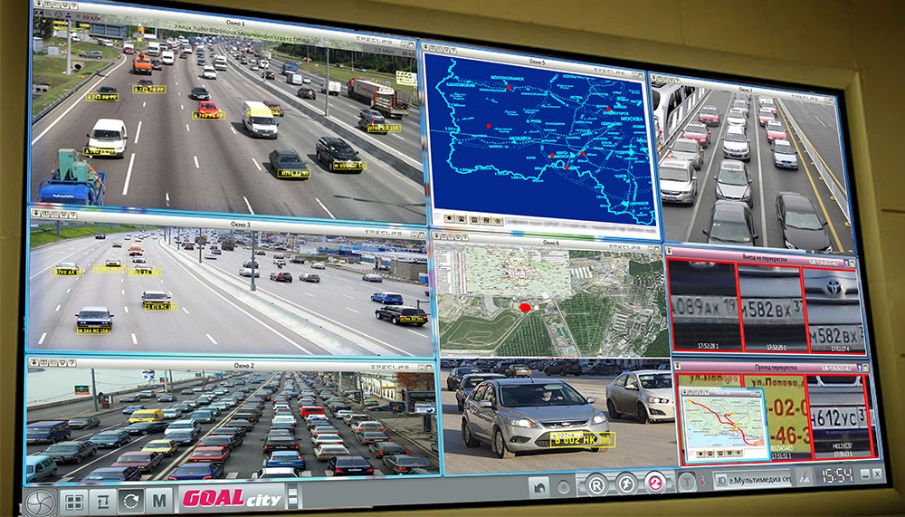


Рисунок 2 – Spetslab-Traffic

Область применения:

* в сочетании с программой SpetLab – Registration Check, разработанной специально для обеспечения эффективной работы сотрудников ГАИ, SL-Traffic позволяет автоматически регистрировать транспортные средства, проходящие через пункт пропуска ГАИ без участия человека.
* в сочетании с программой «СпецЛаб – Досье» SL-Traffic служит для обеспечения безопасности, контроля въезда/выезда транспортных средств со стоянок, коллективных гаражей и запрещенных зон;
* контроль движения автомобилей, как в черте города, так и на трассах.

Возможности:

* распознавание всех однострочных номеров, выпущенных в России и ряде других стран ближнего и дальнего зарубежья;
* одновременное распознавание до 20 номерных знаков на одном кадре (большее количество номеров просто не поместится в кадре с необходимым качеством изображения);
* оптическое определение скорости транспортных средств по скорости смены рамки номерного знака;
* нахождение наилучшего числа из последовательности определенных чисел для одного автомобиля (в случае, если номер автомобиля определялся несколько раз);
* ускоренная работа на поисковых зонах;
* использование четырех алгоритмов распознавания чисел для повышения эффективности;
* регулирование скорости распознавания;
* работать как с собственным архивом, так и с другими базами данных.

Методы и технологии распознавания

В пункте пропуска устанавливается видеокамера в соответствии с вышеперечисленными требованиями. Видеосигнал с этой камеры поступает в компьютер с помощью, установленной в нем платы видеозахвата и в режиме реального времени обрабатывается программным модулем для считывания и распознавания номерных знаков SL-Traffic в следующей последовательности:

* в зоне охвата камерой обнаружено движущееся транспортное средство;
* производится поиск номерного знака (или нескольких номеров);
* выявленный номер распознается;
* делаются снимки проезжающего автомобиля (по желанию пользователя);
* полученная информация передается для регистрации и архивирования в выбранную по усмотрению пользователя базу данных;
* архив хранится.

Пользователю предлагается 4 типа алгоритмов распознавания чисел [7]. Выбор лучшего из них пользователь делает самостоятельно для достижения наилучшего результата в конкретных условиях применения.

Работа программного модуля SL-Traffic в сочетании с программным обеспечением «SpetLab – Проверка регистрации» также производится в режиме реального времени и практически мгновенно (в течение одной секунды) выполняет следующие операции:

* осуществляет распознавание чисел;
* осуществляет поиск во всех базах данных, существующих в ГАИ;
* при обнаружении преступной информации выдает звуковое оповещение и тревожное сообщение на экран монитора;
* вся полученная информация записывается в журнал.

## 1.5.3 "SecurOS Auto"

Система интеллектуального видеоанализа SecurOS Auto обеспечивает распознавание номерных знаков транспортных средств. SecurOS Auto работает на базе интеграционной платформы управления видео SecurOS, позволяющей создать комплекс безопасности с необходимым заказчиком функционалом (рис.1.5).



Рисунок 3 – SecurOS Auto

Область применения:

Сфера деятельности SecurOS Auto охватывает значительный спектр задач от обеспечения безопасности на стоянках до контроля потоков автомобилей по городу.

Возможности

* распознавать номерные знаки более 50 стран мира;
* распознавать номерные знаки по отдельным кадрам без использования видео;
* быстро настроить обновления для распознавания ряда новых стандартов;
* сформировать базу данных распознанных номерных знаков с сохранением соответствующей информации о дате, времени и месте обнаружения
* транспортное средство, скорость и направление его движения, а также ссылка на видеоролик;
* организовать поиск ранее распознанных номерных знаков по заданным параметрам;
* поддерживать одновременно внешние и внутренние списки номеров (информационный, белый, черный);
* автоматически сверять признанный знак государственной регистрации (ОРЗ) по внутренним и внешним спискам (базам данных);
* настроить необходимые реакции системы на распознавание номерных знаков, на результаты поиска;
* оперативно информировать оператора (поддерживается также голосовое оповещение) и/или отправлять уведомления (e-mail, SMS и т.д.) внешним службам о результатах распознавания номерных знаков и/или результатах сравнения данных транспортного средства со списками номеров;
* создать необходимое количество рабочих мест оператора;
* автоматически формировать отчеты разного типа по результатам распознавания, поиска в базе данных.

Методы и технологии распознавания

Модуль локализации генерирует гипотезы о вероятном присутствии номерного знака на изображении (в кадре) и правильно обрабатывает ситуации, когда в кадре имеется несколько номерных знаков. Анализ базируется на поиске участков кадра, имеющих текстовую структуру разностей яркости. В настройках режима распознавания можно указать, производить ли поиск номерных знаков в фиксированных участках рамки.

Положение номерного знака в кадре прогнозируется не только на основе анализа текущего изображения, но и путем экстраполяции результатов распознавания предыдущих кадров.

Масштабирование и бинаризация изображения. Распознавание GRZ осуществляется в несколько этапов анализа двухградационных (бинаризированных) и серых изображений. Область кадра, заданная локализатором, масштабируется, а затем бинаризуется с помощью собственного алгоритма. В результате получается изображение, удобное для быстрого анализа и состоящее только из белых и черных пикселей.

Распознавание чисел. На этапе анализа двоичного изображения происходит обнаружение и распознавание символов номерных знаков, их выравнивание в последовательность символов, а также поиск отсутствующих символов на основе использованных номерных шаблонов определенной страны.

Сравнение изображения с набором шаблонов позволяет избежать ошибочного распознавания «артефактов», определения неправильного количества символов в номерном знаке, неправильного распознавания знаков, близких друг другу и т.п.

SecurOS Auto использует шаблоны, учитывающие геометрию местоположения символов на номерных знаках. Как правило, для правильного наложения шаблона на изображение достаточно распознавания трех-четырех символов в числе. Это позволяет определить расположение остальных символов и распознать их с высокой степенью вероятности. Шаблоны содержат информацию о разрешенных последовательностях букв и цифр, их количестве, физических размерах цифр – соотношении длины и ширины кадров, наличии изображения государственного флага или других графических элементов. SecurOS Auto позволяет одновременно работать с практически неограниченным количеством наборов шаблонов GRZ из разных стран, однако для надежного распознавания по скорости поступления видео рекомендуется использовать шаблоны не более пяти-шести стран.

Реализованный анализ полутонового изображения позволяет значительно повысить точность распознавания. Это особенно важно, если необходимо выбрать между несколькими «конкурирующими» вариантами, которые имеют схожие варианты написания, степень отличия которых может стать еще меньше из-за малого размера символов, низкой контрастности, шума и т.д.

Уточнение результата распознавания, то есть повторное распознавание, выполняется в зависимости от результатов первого прохода. Первоначальное локализованное изображение повторно обрабатывается с различными настройками контрастности, масштабированием и другими типами нормализации в зависимости от обнаруженной проблемы.

Межкадровое слияние результатов распознавания и выдача окончательного результата. Распознавание номерного знака производится на всех кадрах, где его размер и контрастность находятся в заданных пределах, а результаты передаются в модуль «Отслеживание» для межкадрового сравнения, прогнозирование траектории движения номерного знака на последующих кадрах видеоряда, а также выдача конечного результата (фиксация числа) Модуль "Отслеживание" может одновременно "отслеживать" все числа, найденные в кадре.

Для стабильно хороших результатов достаточно, чтобы номерной знак был виден не менее трех-пяти кадров. Например, одна камера 25-30 кадров в секунду, контролирующая участок проезжей части длиной 8 метров, позволяет получить четыре и более кадров движущегося автомобиля со скоростью 120-180 км/ч. Такого количества кадров, как правило, достаточно для уверенного распознавания всех символов номерного знака.

Качество и скорость распознавания зависят от таких показателей как качество оптики и экспозиции камеры, достаточность освещенности зоны наблюдения, мощность обработки сервера видеоаналитики и т.д. Рецензирование дизайна и лабораторное тестирование перед созданием спецификации проекта сохраняет уровень распознавания на уровне 96-99,9%.

Окончательный результат распознавания выдается в виде «события», доступного всем модулям в «SecurOS». В зависимости от пользовательских настроек, SecurOS сохраняет весь фрагмент проезда транспортного средства или его «лучшую раму», определенную на основе промежуточных результатов распознавания.

SecurOS Auto имеет несколько режимов выдачи результатов распознавания, зависящих от настроек пользователя. Двое из них – «дорогая» и «парковка» предполагают, что камера установлена стационарно, а третья, «мобильная», что камера устанавливается в движущийся, например, патрульный автомобиль.

В режиме "дорога" распознаются только движущиеся номера, а результат выдается, когда номерной знак выходит из рамки или если он "утерян" в кадре (обычно в результате блокировки).

В режиме «парковки» также распознаются фиксированные номера, а результат также выдается в случае остановки транспортного средства, ранее находившегося в движении. В последнем случае "SecurOS Auto" определяет, в какой момент автомобиль с распознанным номерным знаком "вышел из рамки"; это событие можно перенести в систему управления шлагбаумом.

Сохранение результатов распознавания в базе данных и совпадение со списками номерных знаков, и проверка на превышение скорости

Событие распознавания GRZ обрабатывается, среди прочего, в модуле «Контроль» номерной знак принадлежит той или иной полосе, скорость проезжающего транспортного средства определяется по данным радара, результат распознавания сохраняется в базе данных SecurOS Auto и проверяется по внутреннему и внешнему спискам. Результат проверки, например, наличие этого номера в базе краж или превышение разрешенной скорости на этой полосе, также сохраняется в собственной базе данных, и, кроме того, отправляется как событие в SecurOS. В результате в пользовательском интерфейсе «SecurOS Auto» появляется строка «тревожная».

В таблице 1 представлен сравнительный анализ рассмотренных систем.

Таблица 1 – Обобщенная таблица сравнения

|  |  |  |
| --- | --- | --- |
| Название системы | Количество правильно распознанных номеров (T ), % | «Мусор» (F), % |
| Spetslab-Traffic | 87,5 | 12,5 |
| Автомаршал | 84,8 | 15,2 |
| SecurOS Auto | 91,2 | 8,8 |

1.6 Выводы по разделу

В данном разделе была проанализирована предметная область, произведено обоснование актуальности темы, осуществлен обзор аналогов.

# 2. Постановка задачи

В данном разделе будет представлена постановка задачи, необходимая для успешного выполнения проекта по разработке программного модуля распознавания номерных знаков на парковке. В связи с растущими проблемами, связанными с управлением парковочными ресурсами, разработка такого модуля имеет высокую актуальность и важность для обеспечения эффективного использования парковочных зон и повышения удобства для пользователей.

2.1 Цели и задачи проекта

Цель проекта: разработать систему распознавания номерных знаков на парковке, которая будет обеспечивать эффективное управление парковочными ресурсами и повышать удобство для пользователей.

Задачи проекта:

* Исследовать существующие методы и технологии распознавания номерных знаков
* Разработать алгоритмы обработки изображений и распознавания номерных знаков
* Создать программный модуль распознавания номерных знаков и провести тестирование
* Оптимизировать производительность и точность системы с помощью

2.2 Требования к программному модулю

По результатам испытаний на рынке систем распознавания номерных знаков можно сформулировать основные требования к алгоритму распознавания. Основным показателем качества алгоритма является вероятность распознавания.

Средняя вероятность распознавания составляет более 80%. Таким образом, разработанный алгоритм распознавания должен давать результаты с вероятностью распознавания 80% и выше.

2.3 Выводы по разделу

В данном разделе были установлены цель и задачи, а также основные требования к разрабатываемому программному модулю.

3 Выбор метода распознавания

В данном разделе будут рассмотрены основные методы, которые используются для распознавания номерных знаков автомобилей.

3.1 Методы распознавания номерных знаков

Для выявления и идентификации автомобильных номерных знаков применяются различные техники. Разберем несколько из них: метод Виолы-Джонса и машины опорных векторов.

Метод Виолы-Джонса, однако, страдает от того, что при углах наклона, превышающих 30 градусов, его точность и эффективность снижаются [3].

Опорные векторные машины, в свою очередь, ограничены в том, что они применимы только к задачам с двумя классами.

Тем временем, сверточные нейронные сети демонстрируют лучшие результаты при распознавании объектов на изображениях. В связи с этим в данном исследовании мы решили прибегнуть к использованию сверточных нейронных сетей для обнаружения и идентификации номерных знаков автомобилей при различных углах наклона и в условиях шума.

Рассмотрим вышеуказанные методы более подробно.

3.2 Метод Виолы-Джонса

Метод, придуманный П. Виолой и М. Джонсом в 2001 году и известный как метод Виолы-Джонса, представляет собой инновационный алгоритм для обнаружения объектов в реальном времени. Главной его задачей является выявление лиц, однако он также успешно применяется для определения различных классов изображений [3].

Этот метод можно разбить на две ключевые части: алгоритм обучения и алгоритм распознавания. Основные принципы, лежащие в его основе, включают:

* Использование целостного представления изображений, что значительно ускоряет вычисления, необходимые для поиска объектов.
* Применение знаков Хаара для точного обнаружения интересующих объектов, в данном случае - угловых точек.
* Строительство классификатора, использующего алгоритм boosting (что переводится с английского как "усиление"), для выбора наиболее подходящих признаков в заданной части изображения.
* Все извлеченные признаки подвергаются анализу классификатором, который выдает результат "истина" или "ложь".
* Использование каскадов функций для эффективного отсеивания ненужных окон без объекта, что ускоряет процесс обнаружения.

Перечисленные принципы способствуют поиску объектов в реальном времени, однако следует учитывать, что основным недостатком данного подхода является длительное время обучения и необходимость большого объема обучающих данных для классификатора. Если угол наклона объекта на изображении превышает 30 градусов, процент успешного обнаружения снижается. Кроме того, при изменениях в освещении или сильном затемнении объекта, алгоритм может не давать нужных результатов.

Однако у метода есть и свои преимущества:

* возможность обнаружения более чем одного объекта;
* применение простых классификаторов;
* высокая скорость поиска;
* возможность использования в видеопотоке.

В таблице 2 представлены результаты тестирования данного метода. Точность определяется процентом правильно распознанных символов по отношению к общему количеству символов. [4].

Таблица 2 – Расчет точности

|  |  |
| --- | --- |
| Группа | Точность сегментации, % |
| Нормальный | 90 |
| Под углом | 80 |
| Искаженный | 73 |

3.3 Машины опорных векторов

Метод опорных векторов (SVM) представляет собой управляемый алгоритм обучения, связанный с линейными классификаторами, применяемый для задач регрессии и классификации. Тихоновская регуляризация может быть рассмотрена как частный случай SVM [5]. Этот метод классификации с максимальным разрывом постоянно уменьшает эмпирическую ошибку классификации и увеличивает разрыв.

Вспомогательная векторная машина представляет интересную идею: преобразование исходных векторов в пространство большей размерности, где ищется разделительная гиперплоскость. Полученная гиперплоскость имеет две параллельные гиперплоскости по бокам и максимальный зазор в этом пространстве. Разделяющая плоскость является плоскостью с максимальным расстоянием до этих параллельных гиперплоскостей. Средняя погрешность результатов метода зависит от разницы или расстояния между этими параллельными гиперплоскостями; с увеличением расстояния погрешность уменьшается [6].

Классификация данных - это распространенная задача в мире алгоритмов машинного обучения. В этой задаче каждый элемент данных представлен вектором чисел в многомерном пространстве. Каждый элемент данных принадлежит одному из двух классов, и наша цель - найти гиперплоскость, которая разделяет эти классы максимально четко. Это достигается путем поиска гиперплоскости такой, что расстояние от нее до ближайшего элемента данных наибольшее. Эта оптимальная разделяющая гиперплоскость, а также соответствующий ей линейный классификатор, играют ключевую роль в успешной классификации данных.

В таблице 3 представлены результаты тестирования данного метода. Точность определяется процентом правильно распознанных символов по отношению к общему количеству символов

Таблица 3 – Расчет точности

|  |  |
| --- | --- |
| Группа | Точность сегментации, % |
| Нормальный | 95 |
| Под углом | 94 |
| Искаженный | 92 |

3.3 Использование искусственных нейронных сетей

Человеческий мозг проявляет выдающиеся способности в решении задачи распознавания образов, превосходя в этом аспекте даже самые мощные цифровые компьютеры. Это обусловлено особенностями информационной обработки мозга, которые существенно отличаются от применяемых в обычных цифровых компьютерах методов. Мозг человека представляет собой сложную, нелинейную, параллельную систему обработки информации. Он способен так организовать свои структурные компоненты, чтобы успешно выполнять конкретные задачи, такие как распознавание образов, с многократным преимуществом в скорости по сравнению с самыми быстрыми компьютерами [2].

В данной работе предлагается использовать искусственные нейронные сети (ИНС). Согласно [1], нейронную сеть можно определить следующим образом:

"Искусственная нейронная сеть представляет собой масштабный распределенный параллельный процессор, состоящий из элементарных блоков для обработки информации, называемых нейронами. Эти нейроны накапливают экспериментальные знания и предоставляют их для последующей обработки".

Преимущества нейронных сетей были описаны в [4] и [3]. Давайте перечислим некоторые из них:

* Нелинейность, что позволяет моделировать сложные отношения в данных.
* Преобразование входящей информации в выходные результаты.
* Адаптивность, позволяющая сети настраиваться под различные задачи.
* Понятность ответов, делая выводы из сети более интерпретируемыми.
* Учёт контекстной информации для более точных прогнозов.
* Способность к отказоустойчивости, что позволяет сохранять работоспособность даже при некоторых повреждениях.
* Единообразный подход к анализу и проектированию.

Процесс обучения сети, или настройки, представляет собой процесс модификации свободных параметров нейронной сети, осуществляемый путем моделирования её взаимодействия с окружающей средой. Тип обучения определяется методом выбора характеристик [2].

1. Этот процесс настройки можно описать следующей последовательностью действий:
2. Сеть получает стимулы извне.
3. Свободные параметры сети подвергаются изменениям.

После изменений во внутренней структуре нейронная сеть реагирует на входные стимулы по-новому.

Алгоритм обратного распространения, известный как один из основных методов обучения, находит свое место среди разнообразных алгоритмов обучения. Наиболее известным и широко применяемым из них является алгоритм обратного распространения. Этот метод нацелен на минимизацию разницы между реальными выходными значениями нейронной сети (ANN) и желаемыми выходами. Его детальное описание можно найти в источниках [1].

Суть обучения через алгоритм обратного распространения заключается в переводе задачи отображения от входных данных к выходным значениям (путем использования набора примеров) в установку оптимальных синаптических весов и порогов для многослойного персептрона [1]. Процесс настройки сети можно рассматривать как выбор наилучшей модели из набора "кандидатов" структур, основанный на определенных критериях.

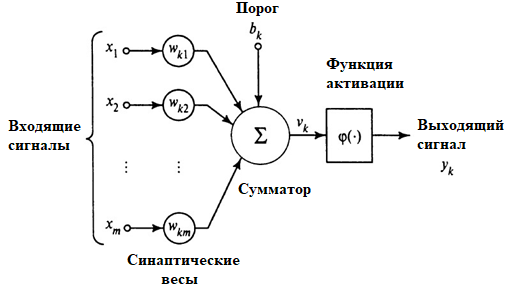


Рисунок 4 – Нелинейная модель нейрона

Нейрон, в контексте нейронных сетей, представляет собой ключевую единицу обработки информации. На иллюстрации 4 мы можем увидеть модель нейрона, которая лежит в основе искусственных нейронных сетей. Эта модель включает в себя три основных компонента:

Ансамбль синапсов (или связей), каждый из которых имеет свой собственный вес или силу.

Сумматор, который складывает взвешенные входные сигналы, учитывая их веса, связанные с соответствующими синапсами нейрона.

Функция активации, которая ограничивает амплитуду выходного сигнала нейрона.

Модель нейрона также может включать пороговый элемент, обозначаемый символом , который влияет на функцию активации, регулируя входной сигнал.

Математически, работу нейрона можно описать следующими уравнениями [1]:





где , , … ,  – входные сигналы; , , … ,  — синаптические весы нейрона k;  - линейная комбинация входных действий (выход линейного комбинатора);  - порог;  – функция активации;  является выходным сигналом нейрона.

Постсинаптический потенциал рассчитывается следующим образом:



Функции активации – это специальные инструменты, которые формируют выход сигнала нейрона, в зависимости от воздействия на него локального поля. Давайте рассмотрим несколько разнообразных функций активации.

Одной из таких функций является сигмовидная функция. Она выделяется своей способностью стремительно переходить от линейного к нелинейному поведению. Логистическая функция является примером сигмовидной функции и может быть представлена следующим образом:



Другой функцией активации является линейный выпрямитель (ReLU), заданный следующим выражением:



Сверточные нейронные сети (CNN), разработанные Яном Лекуном в 1988 году, представляют собой инновационную архитектуру в мире искусственного интеллекта. Их первоначальная цель заключалась в улучшении процесса распознавания объектов на изображениях, и сегодня они остаются неотъемлемой частью глубокого обучения. CNN считаются одной из самых важных технологических новшеств в области компьютерного зрения.

Структура CNN включает в себя разнообразные слои, такие как сверточные, нелинейные, объединяющие и полностью связанные. Процесс обработки изображения через эти слои завершается формированием ответа, который может представлять собой класс объекта или вероятность принадлежности к определенным классам, наиболее точно описывающим изображение. CNN являются однонаправленными, то есть они не имеют обратной связи и представляют собой многослойную нейронную сеть.

В процессе обучения CNN используются стандартные методы, часто опираясь на алгоритм обратного распространения ошибки. Отмечается, что функция активации в таких сетях может быть выбрана исследователем в зависимости от конкретных задач и требований.

3.4 Обоснование выбора метода

Для реализации модуля будут использоваться сверточная нейронная сеть, так как она имеет ряд преимуществ:

1. **Автоматическое извлечение признаков**: Нейронные сети могут автоматически извлекать признаки из изображений на разных уровнях абстракции, начиная с низкоуровневых признаков, таких как грани и текстуры, и заканчивая более высокоуровневыми концепциями, такими как формы и объекты. Это позволяет модели самостоятельно находить характеристики объектов на изображении без необходимости вручную определять признаки.
2. **Гибкость**: Нейронные сети предоставляют гибкую платформу для обучения на разнообразных данных и задачах. Их архитектура может быть адаптирована для различных видов объектов и условий съемки.
3. **Обучение на больших наборах данных**: Современные нейронные сети обычно требуют больших объемов данных для обучения. Однако, когда доступны достаточно большие и разнообразные наборы данных, нейронные сети способны достичь высокой точности и обобщающей способности в распознавании объектов.
4. **Прогрессивные результаты**: С появлением новых архитектур и методов обучения, нейронные сети продолжают достигать новых рекордов в точности распознавания объектов на изображениях. Такие архитектуры, как сверточные нейронные сети и их вариации, показывают впечатляющие результаты в области компьютерного зрения.
5. **Автоматизация**: После обучения нейронной сети ее можно использовать для автоматического распознавания объектов на новых изображениях без необходимости ручной обработки. Это обеспечивает высокую эффективность и ускоряет процесс обработки изображений в различных приложениях.

Методы, такие как SVM и метод Виолы-Джонса, также имеют свои преимущества и недостатки. Но существует несколько причин, по которым они могут быть менее предпочтительными по сравнению с нейронными сетями для распознавания объектов на изображении:

1. **Ограниченные возможности в извлечении признаков**: SVM и метод Виолы-Джонса требуют ручной инженерии признаков, что может быть сложным и трудоемким процессом. В случае большого разнообразия объектов и условий съемки создание подходящих признаков может быть непростой задачей. В отличие от этого, нейронные сети способны автоматически извлекать признаки из данных, что делает их более гибкими и способными к обучению на разнообразных наборах данных.
2. **Требования к размеру обучающего набора данных**: SVM и метод Виолы-Джонса могут быть более чувствительны к размеру и разнообразию обучающего набора данных. Они могут показывать хорошие результаты только при наличии достаточного количества представительных примеров изображений объектов, что может быть проблематично для некоторых приложений.
3. **Производительность на сложных задачах**: В случае больших и сложных наборов данных или задач компьютерного зрения, например, сегментации объектов с высоким уровнем детализации или с распознаванием объектов в условиях переменного освещения, нейронные сети обычно показывают более высокую производительность по сравнению с SVM и методом Виолы-Джонса.
4. **Необходимость в переобучении**: SVM и метод Виолы-Джонса могут иметь склонность к переобучению на некоторых типах данных, особенно если обучающий набор данных недостаточно разнообразен или если не проводится должная настройка параметров алгоритмов.

Это не означает, что SVM и метод Виолы-Джонса не имеют применения в распознавании объектов на изображениях. Они могут быть эффективны в некоторых специфических сценариях и при условии, что их преимущества и недостатки учитываются при выборе подходящего метода для конкретной задачи. Однако в целом, современные нейронные сети, особенно сверточные, обычно являются более предпочтительным выбором для задач распознавания объектов на изображениях благодаря своей гибкости.

Таким образом, использование нейронных сетей для распознавания объектов на изображении обеспечивает эффективный и точный метод автоматической обработки изображений в различных областях, таких как медицина, автомобильная промышленность, безопасность и многое другое.

3.5 Выводы по разделу

В данном разделе были рассмотрены основные методы, использующиеся для распознавания объектов на изображении, и был выбран метод, который будет использоваться при разработке.

4 Описание выбранных моделей

В данном разделе будут рассмотрены модели нейронных сетей, использующиеся при разработке.

4.1 Обнаружение положения 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.

Архитектура ResNet50 разделена на четыре основные части: сверточные слои, блок идентификации, сверточный блок и полностью связанные слои. Сверточные слои отвечают за извлечение объектов из входного изображения, в то время как блок идентификации и сверточный блок отвечают за обработку и преобразование этих объектов. Наконец, для окончательной классификации используются полностью связанные уровни.

Сверточные слои в ResNet50 состоят из нескольких сверточных слоев, за которыми следует пакетная нормализация и повторная активация. Эти слои отвечают за извлечение объектов из входного изображения, таких как края, текстуры и формы. За сверточными слоями следуют слои с максимальным объединением, которые уменьшают пространственные размеры карт объектов при сохранении наиболее важных объектов.

Блок идентификации и сверточный блок являются ключевыми строительными блоками ResNet50. Блок идентификации – это простой блок, который передает входные данные через серию сверточных слоев и добавляет входные данные обратно к выходным. Это позволяет сети изучать остаточные функции, которые сопоставляют входные данные с желаемыми выходными данными. Сверточный блок аналогичен блоку идентификации, но с добавлением сверточного слоя 1x1, который используется для уменьшения количества фильтров перед сверточным слоем 3x3.

Заключительная часть ResNet50 – это полностью подключенные слои. Эти слои отвечают за окончательную классификацию. Выходные данные конечного полностью подключенного уровня передаются в функцию активации softmax для получения конечных вероятностей классов.

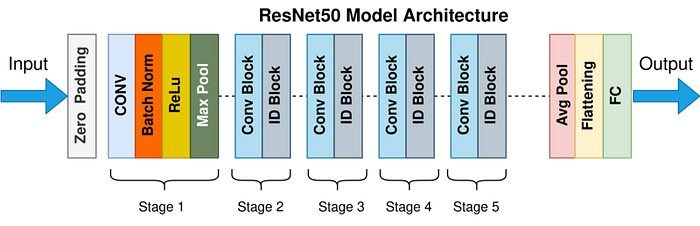


Рисунок 5 – Архитектура ResNet-50

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.

Тестирование модели показало достаточно хороший уровень точности. Результаты тестирования представлены на рисунке 6.



Рисунок 6 – Тестирование модели

4.2 Определение номера автомобиля

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 модели, описанной в п. 4.1. 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.

Структура модели представлена на рисунке 7.

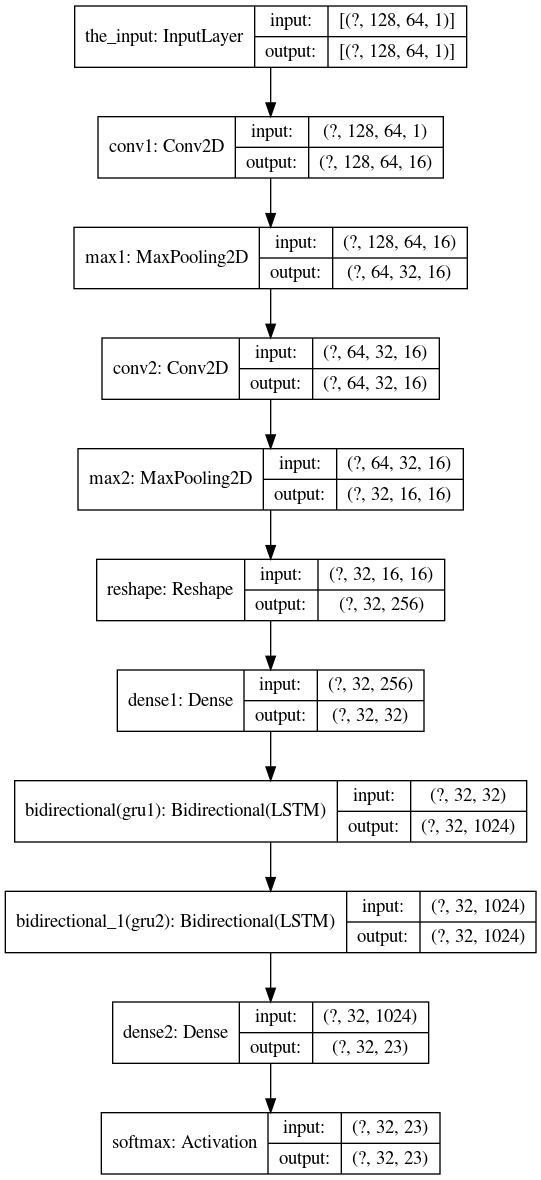
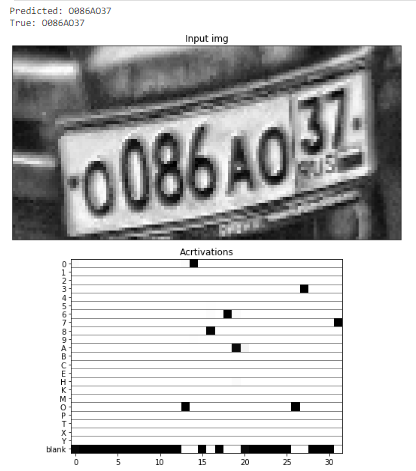


Рисунок 7 – Структура модели

Результаты тестирования представлены на рисунке 8.



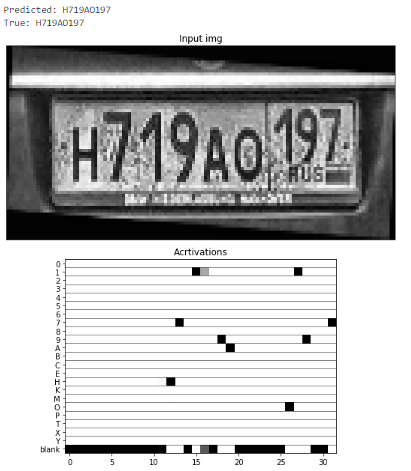


Рисунок 8 – Результаты тестирования

4.2 Определение номера автомобиля

В данном разделе были описаны модели, которые используются при разработке программного модуля.

5 Программная реализация

В данном разделе будут рассмотрены средства, методы и библиотеки, которые использовались при программной реализации.

5.1 Программные средства

При разработке программного модуля использовались следующие основные инструменты:

1. **OpenCV**: OpenCV (Open Source Computer Vision Library) является библиотекой компьютерного зрения с открытым исходным кодом. В коде OpenCV используется для обработки изображений, таких как загрузка видео, работа с кадрами из видео, обработка изображений (изменение размеров, изменение цветовых пространств, наложение фильтров, выделение контуров и т. д.).
2. **PyQt6**: PyQt6 - это набор Python-биндингов для библиотеки Qt, который позволяет создавать графические пользовательские интерфейсы (GUI). В коде PyQt6 используется для создания графического интерфейса пользователя (GUI) для вашего приложения.
3. **sqlite3**: sqlite3 - это встроенная база данных SQLite, которая является легкой и удобной в использовании. Она применяется для хранения и управления данными о номерных знаках и количестве посещений.
4. **tensorflow**: TensorFlow - это открытая библиотека глубокого обучения, разработанная компанией Google. TensorFlow используется для выполнения инференса модели для обнаружения номерных знаков и распознавания текста на изображениях.
5. **scikit-image**: scikit-image - это библиотека для обработки изображений, которая предоставляет множество функций для работы с изображениями. Она применяется в коде для выполнения различных операций обработки изображений, таких как выделение границ и линий на изображениях.
6. **matplotlib**: Matplotlib - это библиотека для создания различных видов графиков и визуализации данных в Python. Matplotlib используется для визуализации изображений и результатов обработки.
7. **numpy**: NumPy - это библиотека Python для выполнения вычислений с многомерными массивами и матрицами. Она применяется для работы с массивами изображений и выполнения различных математических операций.

5.2 Разработка

Разработанный программный модель можно разбить на несколько частей:

1. **Пользовательский интерфейс**:
   * Скрипт загружает макет пользовательского интерфейса из файла **.ui** с помощью PyQt6.
   * Он отображает кнопки для выполнения действий, таких как загрузка видеофайла, разделение видео на кадры, выбор изображения и распознавание номерных знаков автомобилей.
   * В интерфейсе также есть метки для отображения изображений и распознанных номеров.

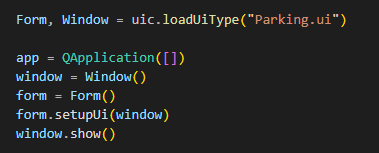


Рисунок 9 – Фрагмент кода

1. **Операции с базой данных**:
   * Скрипт взаимодействует с базой данных SQLite с названием **server.db**.
   * Создается таблица с названием **parking** для хранения номеров автомобильных номеров и количества посещений.
   * Функции **load\_file()** и **save\_file()** обрабатывают загрузку данных из базы данных и сохранение данных в нее соответственно.

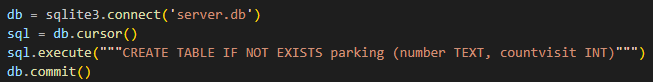


Рисунок 10 – Взаимодействие с БД

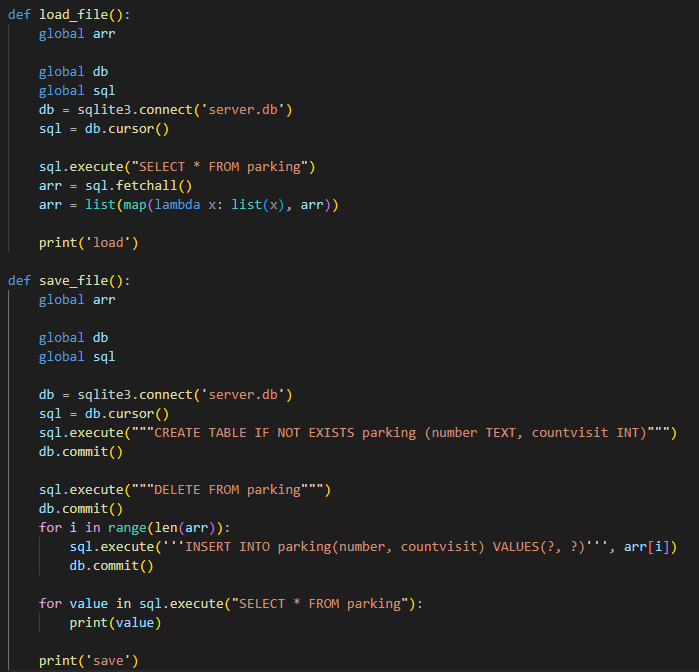


Рисунок 11 – Функции load\_file() и save\_file()

1. **Обработка видео**:
   * Скрипт позволяет загружать видеофайл и разделять его на кадры.
   * Функция **split()** использует MoviePy для разделения видео на кадры и сохранения их как изображения.

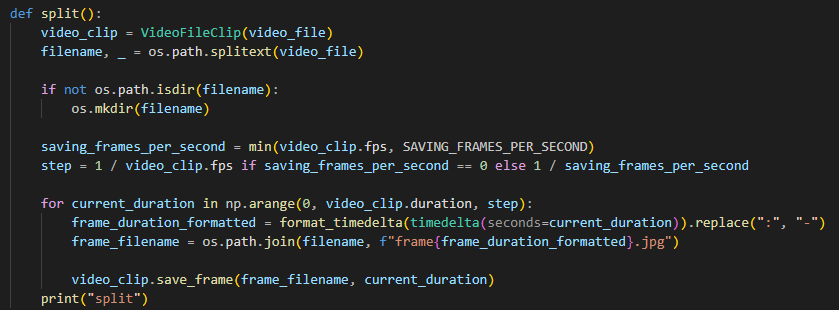


Рисунок 12 – Функция split()

1. **Обработка изображений и распознавание номерных знаков**:
   * Пользователи могут выбрать файл изображения с автомобильным номером.
   * Функция **carplate\_text()** выполняет распознавание номерного знака с использованием модели TensorFlow Lite.
   * Производится предварительная обработка изображения, обнаружение номерного знака, применение преобразований и распознавание текста номерного знака.
   * Распознанный текст номерного знака возвращается для отображения.
2. **Распознавание и обновление UI**:
   * Функция **recognize()** обновляет пользовательский интерфейс с распознанным номером автомобильного номера.
   * Также обновляется количество посещений в базе данных и рассчитывается процент скидки на основе количества посещений.



Рисунок 13 – Функция recognize()

5.3 Результат работы программы

На рисунке 14 представлено окно программы.

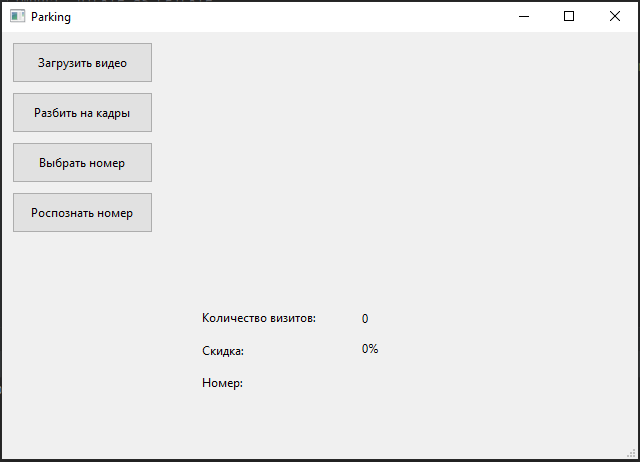


Рисунок 14 – Окно программы

После загрузки видеофайла и разбиения его на кадры можно выбрать один из полученных кадров.

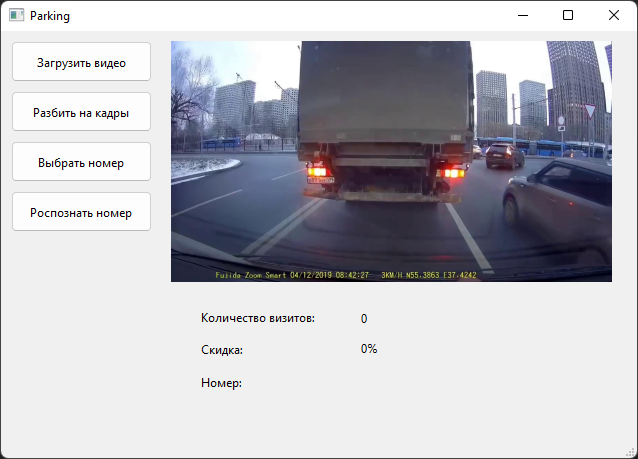


Рисунок 15 – Кадр выбран

Дальше можно распознать номер. Результат работы показан на рисунках 16-17.



Рисунок 16 – Определение номера

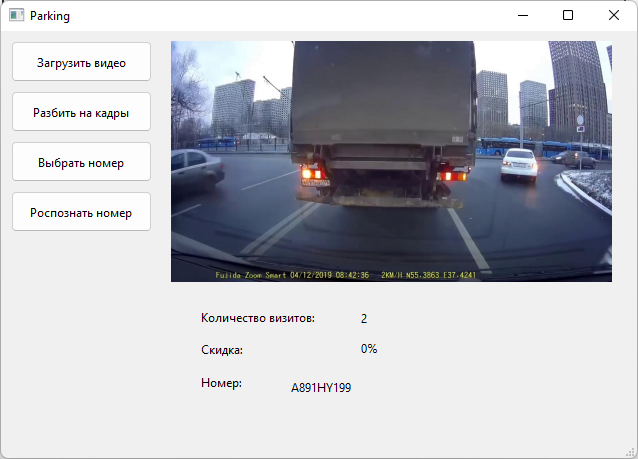


Рисунок 17 – Окно программы после определения

5.4 Выводы по разделу

В данном разделе были описаны основные разработанные функции.

6 Оценка качества

В данном разделе будет проведено тестирование разработанного программного модуля, а также проведена оценка качества его работы.

6.1 Тестирование

Для тестирования было подготовлено 100 изображений автомобилей в разных условиях. Примеры приведены на рисунке 18.





Рисунок 18 – Примеры тестовых изображений

Фрагмент итогов тестирования представлены в таблице 4.

Таблица 4 - Тестирование

|  |  |  |  |
| --- | --- | --- | --- |
| Изображение | Предсказанный номер | Эталонный номер | Распознано верно? |
| 1 | А123ВС78 | А123ВС78 | да |
| 2 | К456МО34 | К456МО34 | да |
| 3 | У789ХУ56 | У789ХУ56 | да |
| 4 | М012ОР78 | М012ОР78 | да |
| 5 | Т345НС72 | Т345НС12 | нет |
| 6 | А890СТ56 | А890СТ56 | да |
| 7 | М123КУ78 | М123КУ78 | да |
| 8 | О456СХ34 | О456СХ34 | да |
| 9 | А012МО78 | А012МО78 | да |
| 10 | У678ХС56 | У678ХС56 | да |

Диаграмма тестирования представлена на рисунке 19.



Рисунок 16 - Диаграмма

Заключение

В ходе выполнения данной пояснительной записки был разработан и реализован программный проект, представляющий собой систему распознавания автомобильных номеров на изображениях. Программа успешно использует различные технологии, такие как TensorFlow Lite, OpenCV, SQLite, и другие, для решения задачи распознавания номеров.

Проект предоставляет пользователю интерфейс для загрузки видео и изображений, разделения видео на кадры, выбора конкретного кадра для распознавания, а также автоматического распознавания номеров на изображениях.

Основные функции проекта включают в себя обработку изображений с использованием алгоритмов компьютерного зрения, таких как алгоритмы Canny и Hough, а также использование нейронных сетей для распознавания символов на номерных знаках.

Благодаря разработанной программе пользователь получает возможность эффективно обрабатывать видео и изображения с автомобильными номерами, что может быть полезно для различных целей, включая контроль парковок, автоматизацию учета автомобилей и другие.

Для оценки качества работы программы была проведена серия тестов на сгенерированной тестовой выборке, результаты которых показали, что точность распознавания составляет более 80%, что свидетельствует о высокой эффективности и точности разработанной системы.

Таким образом, разработанный программный проект представляет собой важный шаг в области распознавания автомобильных номеров, что может быть полезно для широкого спектра приложений, связанных с автомобильной безопасностью, учетом транспортных средств и другими задачами.

Список литературы

1. Habr Mask R-CNN: архитектура современной нейронной сети для сегментации объектов в изображениях [Электронный ресурс]: <https://habr.com/post/421299>
2. Habr Алгоритм распознавания номера на изображении с низкой вероятностью ошибки второго рода [Электронный ресурс]: <https://habr.com/ru/articles/501436/>
3. GitHub tflite\_avto\_num\_recognation [Электронный ресурс]: <https://github.com/sovse/tflite_avto_num_recognation/tree/main>
4. Habr Что такое свертка нейронная сеть [Электронный ресурс]: https://habr.com/post/309508
5. Автомаршал [Электронный ресурс]: уч. веб-сайт. 2014. – Режим доступа: http://avtomarshal
6. SL Трафик. Безопасный город [Электронный ресурс]: оф. веб-сайт – Режим доступа: <http://goal.ru/>
7. Evaluating image segmentation models [Электронный ресурс] / Blog Jeremy Jordan – Режим доступа: <https://www.jeremyjordan.me/evaluating-image-segmentation-models>

# Приложение А

(обязательное)

Схема алгоритмов

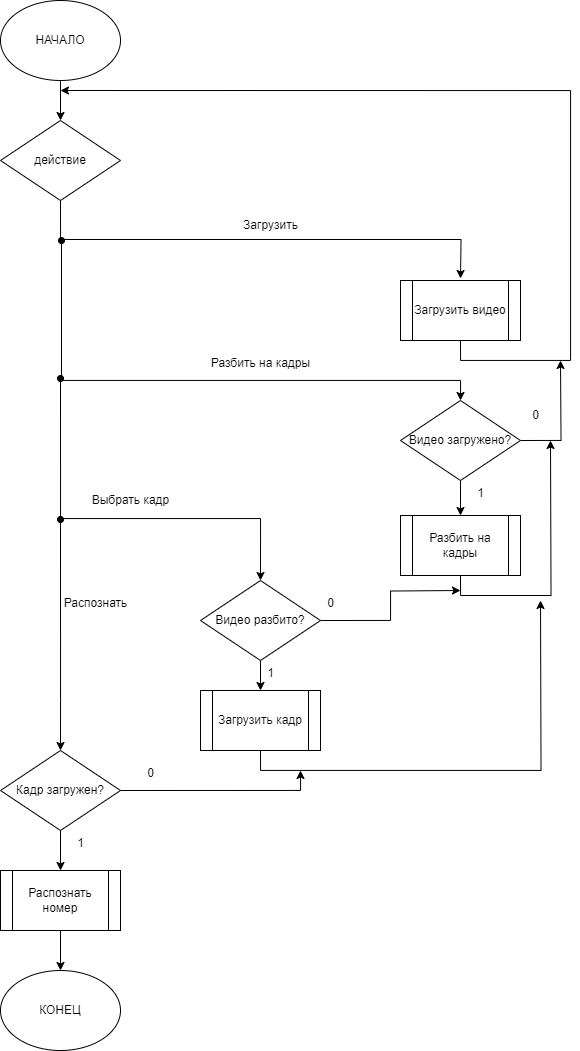


Рисунок 17 – Схема алгоритмов

Приложение Б

(Обязательное)

Экранные формы

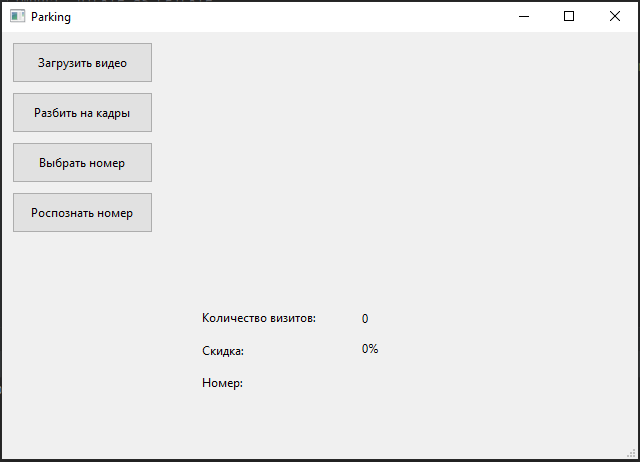


Рисунок 18 – Главное окно

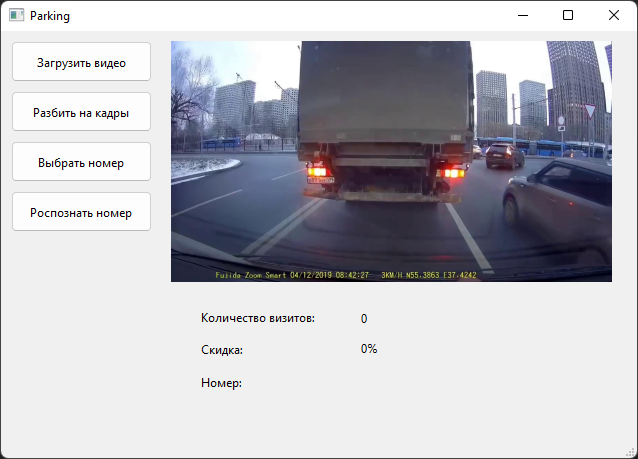


Рисунок 19 – Выбор кадра



Рисунок 20 – Определение номера

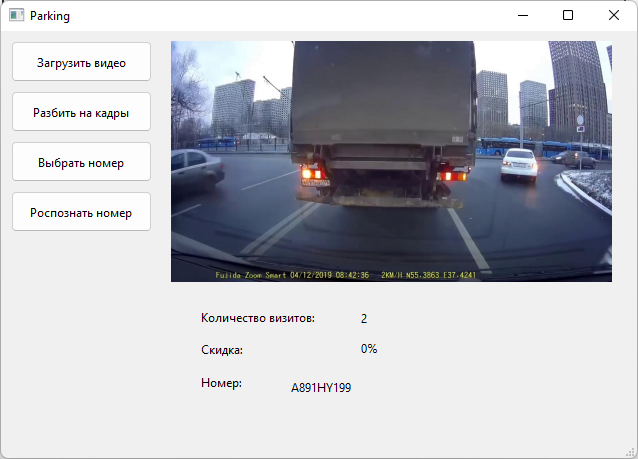


Рисунок 21 – Окно программы после определения

Приложение В

(Обязательное)

Структура нейронной сети

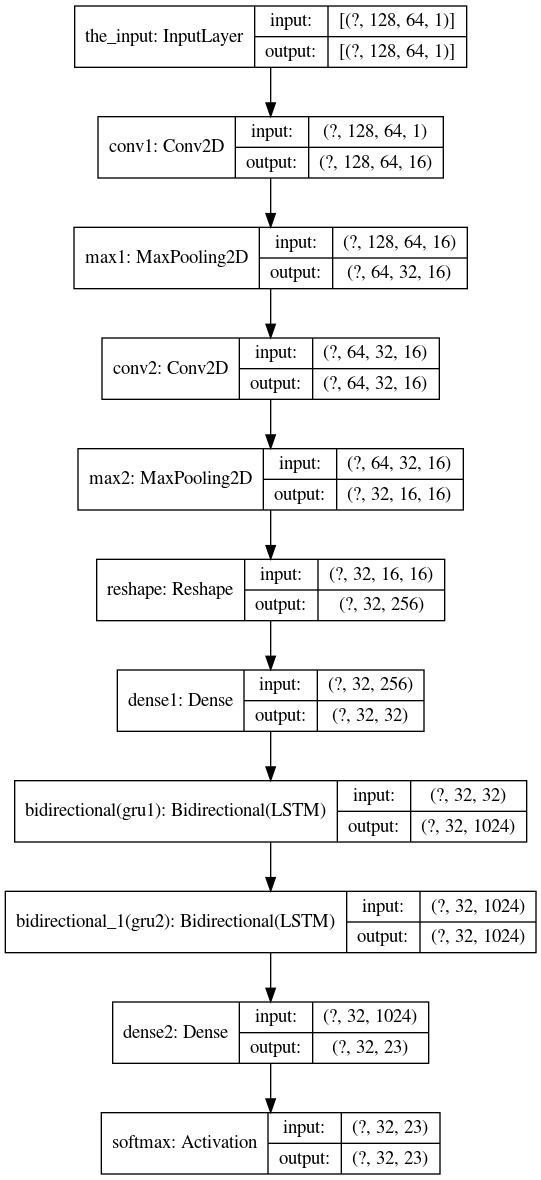


Рисунок 21 – Структура нейронной сети

Приложение Г

(Обязательное)

Оценка качества



Рисунок 22 - Диаграмма

Приложение Д

(Обязательное)

Программный код

import os

from datetime import timedelta

from pathlib import Path

import matplotlib.pyplot as plt

import cv2

#import \_pickle as cPickle

import numpy as np

import re

import imutils

import pytesseract

import pytesseract as tess

tess.pytesseract.tesseract\_cmd = r'Tesseract-OCR\tesseract.exe'

from PyQt6 import uic

from PyQt6.QtGui import QPixmap

from PyQt6.QtWidgets import QApplication, QFileDialog, QMessageBox

from imutils import contours

from moviepy.editor import VideoFileClip

import json

import sqlite3

import itertools

import tensorflow as tf

from skimage.feature import canny

from skimage.transform import hough\_line, hough\_line\_peaks

from skimage.transform import rotate

from skimage.color import rgb2gray

import matplotlib.gridspec as gridspec

SAVING\_FRAMES\_PER\_SECOND = 1

Form, Window = uic.loadUiType("Parking.ui")

app = QApplication([])

window = Window()

form = Form()

form.setupUi(window)

window.show()

video\_file = ''

image\_file = ''

result = ''

arr=[]

db = sqlite3.connect('server.db')

sql = db.cursor()

db = sqlite3.connect('server.db')

sql = db.cursor()

sql.execute("""CREATE TABLE IF NOT EXISTS parking (number TEXT, countvisit INT)""")

db.commit()

def format\_timedelta(td):

    result = str(td)

    try:

        result, ms = result.split(".")

    except ValueError:

        return result + ".00".replace(":", "-")

    ms = round(int(ms) / 10000)

    return f"{result}.{ms:02}".replace(":", "-")

def load\_file():

    global arr

    global db

    global sql

    db = sqlite3.connect('server.db')

    sql = db.cursor()

    sql.execute("SELECT \* FROM parking")

    arr = sql.fetchall()

    arr = list(map(lambda x: list(x), arr))

    print('load')

def save\_file():

    global arr

    global db

    global sql

    db = sqlite3.connect('server.db')

    sql = db.cursor()

    sql.execute("""CREATE TABLE IF NOT EXISTS parking (number TEXT, countvisit INT)""")

    db.commit()

    sql.execute("""DELETE FROM parking""")

    db.commit()

    for i in range(len(arr)):

        sql.execute('''INSERT INTO parking(number, countvisit) VALUES(?, ?)''', arr[i])

        db.commit()

    for value in sql.execute("SELECT \* FROM parking"):

        print(value)

    print('save')

def load():

    global video\_file

    video\_file= QFileDialog.getOpenFileName()

    path=Path(video\_file[0])

    video\_file=path.name

    print("load")

def split():

    video\_clip = VideoFileClip(video\_file)

    filename, \_ = os.path.splitext(video\_file)

    if not os.path.isdir(filename):

        os.mkdir(filename)

    saving\_frames\_per\_second = min(video\_clip.fps, SAVING\_FRAMES\_PER\_SECOND)

    step = 1 / video\_clip.fps if saving\_frames\_per\_second == 0 else 1 / saving\_frames\_per\_second

    for current\_duration in np.arange(0, video\_clip.duration, step):

        frame\_duration\_formatted = format\_timedelta(timedelta(seconds=current\_duration)).replace(":", "-")

        frame\_filename = os.path.join(filename, f"frame{frame\_duration\_formatted}.jpg")

        video\_clip.save\_frame(frame\_filename, current\_duration)

    print("split")

def check\_format(variable):

    pattern = r'^[A-Z]\d{3}[A-Z]{2}\d{3}$'

    pattern2 = r'^[A-Z]\d{3}[A-Z]{2}\d{2}$'

    if re.match(pattern, variable) or re.match(pattern2, variable):

        return True

    else:

        return False

def choose():

    global image\_file

    image\_file = QFileDialog.getOpenFileName()

    image\_file = image\_file[0]

    form.label\_6.setPixmap(QPixmap(image\_file))

    form.label\_6.setScaledContents(True)

    print("choose")

def carplate\_text():

    global image\_file

    image0 = cv2.imread(image\_file)

    image\_height, image\_width, \_ = image0.shape

    image = cv2.resize(image0, (1024, 1024))

    image = image.astype(np.float32)

    paths = './model\_resnet.tflite'

    interpreter = tf.lite.Interpreter(model\_path=paths)

    interpreter.allocate\_tensors()

    input\_details = interpreter.get\_input\_details()

    output\_details = interpreter.get\_output\_details()

    X\_data1 = np.float32(image.reshape(1, 1024, 1024, 3))

    interpreter.set\_tensor(input\_details[0]['index'], X\_data1)

    interpreter.invoke()

    detection = interpreter.get\_tensor(output\_details[0]['index'])

    net\_out\_value2 = interpreter.get\_tensor(output\_details[1]['index'])

    net\_out\_value3 = interpreter.get\_tensor(output\_details[2]['index'])

    net\_out\_value4 = interpreter.get\_tensor(output\_details[3]['index'])

    img = image0

    razmer = img.shape

    img2 = cv2.cvtColor(img, cv2.COLOR\_BGR2RGB)

    img3 = img[:, :, :]

    number = 0

    while number < len(detection[0][number]) and detection[0, number, 0] > 0.9:

        number = number + 1

    box\_x = int(detection[0, number, 0] \* image\_height)

    box\_y = int(detection[0, number, 1] \* image\_width)

    box\_width = int(detection[0, number, 2] \* image\_height)

    box\_height = int(detection[0, number, 3] \* image\_width)

    cv2.rectangle(img2, (box\_y, box\_x), (box\_height, box\_width), (230, 230, 21), thickness=5)

    plt.imshow(img2)

    plt.xticks([]), plt.yticks([])  # Hides the graph ticks and x / y axis

    plt.show()

    net\_out\_value3

    image = image0[box\_x:box\_width, box\_y:box\_height, :]

    img2 = cv2.cvtColor(image, cv2.COLOR\_BGR2RGB)

    grayscale = rgb2gray(image)

    edges = canny(grayscale, sigma=3.0)

    out, angles, distances = hough\_line(edges)

    h, theta, d = out, angles, distances

    angle\_step = 0.5 \* np.diff(theta).mean()

    d\_step = 0.5 \* np.diff(d).mean()

    bounds = [np.rad2deg(theta[0] - angle\_step),

              np.rad2deg(theta[-1] + angle\_step),

              d[-1] + d\_step, d[0] - d\_step]

    \_, angles\_peaks, \_ = hough\_line\_peaks(out, angles, distances, num\_peaks=20)

    angle = np.mean(np.rad2deg(angles\_peaks))

    angle

    if 0 <= angle <= 90:

        rot\_angle = angle - 90

    elif -45 <= angle < 0:

        rot\_angle = angle - 90

    elif -90 <= angle < -45:

        rot\_angle = 90 + angle

    if abs(rot\_angle) > 20:

        rot\_angle = 0

    rotated = rotate(image, rot\_angle, resize=True) \* 255

    rotated = rotated.astype(np.uint8)

    rotated1 = rotated[:, :, :]

    if rotated.shape[1] / rotated.shape[0] < 2:

        minus = np.abs(int(np.sin(np.radians(rot\_angle)) \* rotated.shape[0]))

        rotated1 = rotated[minus:-minus, :, :]

        print(minus)

    lab = cv2.cvtColor(rotated1, cv2.COLOR\_BGR2LAB)

    l, a, b = cv2.split(lab)

    clahe = cv2.createCLAHE(clipLimit=3.0, tileGridSize=(8, 8))

    cl = clahe.apply(l)

    limg = cv2.merge((cl, a, b))

    final = cv2.cvtColor(limg, cv2.COLOR\_LAB2BGR)

    letters = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'E', 'H', 'K', 'M', 'O', 'P', 'T', 'X',

               'Y']

    def decode\_batch(out):

        ret = []

        for j in range(out.shape[0]):

            out\_best = list(np.argmax(out[j, 2:], 1))

            out\_best = [k for k, g in itertools.groupby(out\_best)]

            outstr = ''

            for c in out\_best:

                if c < len(letters):

                    outstr += letters[c]

            ret.append(outstr)

        return ret

    paths = './model1\_nomer.tflite'

    interpreter = tf.lite.Interpreter(model\_path=paths)

    interpreter.allocate\_tensors()

    input\_details = interpreter.get\_input\_details()

    output\_details = interpreter.get\_output\_details()

    img = final

    img = cv2.cvtColor(img, cv2.COLOR\_BGR2GRAY)

    img = cv2.resize(img, (128, 64))

    img = img.astype(np.float32)

    img /= 255

    img1 = img.T

    img1.shape

    X\_data1 = np.float32(img1.reshape(1, 128, 64, 1))

    input\_index = (interpreter.get\_input\_details()[0]['index'])

    interpreter.set\_tensor(input\_details[0]['index'], X\_data1)

    interpreter.invoke()

    net\_out\_value = interpreter.get\_tensor(output\_details[0]['index'])

    pred\_texts = decode\_batch(net\_out\_value)

    pred\_texts

    fig = plt.figure(figsize=(10, 10))

    outer = gridspec.GridSpec(2, 1, wspace=10, hspace=0.1)

    ax1 = plt.Subplot(fig, outer[0])

    fig.add\_subplot(ax1)

    ax2 = plt.Subplot(fig, outer[1])

    fig.add\_subplot(ax2)

    return pred\_texts[0]

def recognize():

    load\_file()

    global result

    result=carplate\_text().upper()

    if result=='':

        QMessageBox.information(None, "Ошибка распознания", "НЕ РАСПОЗНАНО")

    elif check\_format(result)!=True:

        QMessageBox.information(None, "Ошибка формата", "Результат:"+result)

    else:

        form.label\_7.setText(result)

        count = 0;

        for i in range(len(arr)):

            if arr[i][0] == result:

                count += 1

        if count == 0:

            arr.append([result, 1])

        else:

            for i in range(len(arr)):

                if arr[i][0] == result:

                    arr[i][1] += 1

        for i in range(len(arr)):

            if arr[i][0] == result:

                form.label\_2.setText(str(arr[i][1]))

                if (arr[i][1]<5):

                    form.label\_5.setText('0%')

                elif (arr[i][1]>=5 and arr[i][1]<10):

                    form.label\_5.setText('5%')

                elif (arr[i][1]>=10 and arr[i][1]<20):

                    form.label\_5.setText('10%')

                else:

                    form.label\_5.setText('15%')

    save\_file()

    print("recognize")

form.pushButton.clicked.connect(load);

form.pushButton\_2.clicked.connect(split);

form.pushButton\_3.clicked.connect(choose);

form.pushButton\_4.clicked.connect(recognize);

app.exec()