МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

02.03.03 Математическое обеспечение и администрирование информационных систем
(код и наименование направления подготовки, специальности)

Мобильные и сетевые технологии
(направленность (профиль)/специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему Разработка ПО для классификации состояния фрагмента изображения

Обучающийся О.С. Елец _____
(Инициалы Фамилия) (личная подпись)

Руководитель С.В. Митин
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант А.В. Москалюк
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2023

Аннотация

Название бакалаврской работы – «Сегментация изображения и последующая классификация состояния сегментов».

Объект исследования – применение решений на базе «Сегментации и классификации состояния сегментов». Поиск подходящих алгоритмов распознавания объекта и его состояния на изображении и разработка ПО, отвечающего за это.

Предмет исследования – способы выделения информации в видеопотоке с применением технологии ИИ.

Цель исследования – на базе выбранной методики выполнить разработку ПО для прототипа системы диагностики состояния блока фары автомобиля.

Для достижения цели решаются следующие задачи:

- проанализировать существующие способы алгоритмовраспознавания объектов;

- описать математический аппарат выбранного алгоритма;

- разработать и протестировать систему, на основе построенного математического аппарата и выявленных требований.

Первая глава содержит информацию о возможных способах определения объектов на изображениях с применением искусственного интеллекта.

Во второй главе рассматривается математическая модель выбранного метода и приводятся аргументы, почему именно она была выбрана. В данной главе также описывается математический аппарат алгоритма, его реализация и способы решения поставленной задачи.

В третьей главе предоставляются результаты тестирования разработанного алгоритма на основе построенного математического аппарата и выявленных требований. Производится корректировка системы и описываются возможности по её улучшению.

Abstract

The title of the graduation work is "Image Segmentation and Subsequent Segment State Classification".

The object of the research is the application of solutions based on segmentation and classification of the state of segments. The search for suitable algorithms to recognize the object and its state on the image and the development of the software is responsible for this.

The subject of the research is ways of selecting information in the video stream using AI technology.

The aim of the study is to develop software for a prototype system for diagnosing the state of the headlight unit of the car based on the selected methodology.

To achieve the goal, the following tasks are solved:

- To analyze the existing methods of object recognition algorithms;

- To describe the mathematical apparatus of the selected algorithm;

- To develop and test the system based on the constructed mathematical apparatus and the identified requirements.

The first chapter contains information about possible ways to identify objects in images using artificial intelligence.

The second chapter discusses the mathematical model of the chosen method and provides arguments for this choice. This chapter also describes the mathematical apparatus of the algorithm, its implementation, and how to solve the problem.

The third chapter provides the results of testing the developed algorithm based on the constructed mathematical apparatus and identified requirements. The correction of the system is made and the possibilities of its improvement are described.

Оглавление

Введение

Сегментация изображения — это процесс разделения изображения на несколько сегментов или областей на основе определенных характеристик, таких как цвет, текстура или форма. Основная цель сегментации изображения — упростить изображение или сделать его более осмысленным и легким для анализа.

После завершения сегментации изображения следующим шагом является классификация состояния каждого сегмента на основе определенных критериев. Это может включать в себя определение типа объекта или материала, присутствующего в сегменте, идентификацию любых дефектов или аномалий, или даже обнаружение специфических особенностей, таких как края или углы.

Одно из основных применений сегментации изображений и последующей классификации находится в области компьютерного зрения, где она используется в различных задачах, таких как обнаружение, распознавание и отслеживание объектов. Например, при автономном вождении сегментация изображения может использоваться для идентификации различных объектов, таких как пешеходы, транспортные средства и дорожные знаки, а последующая классификация может помочь определить состояние каждого объекта (например, переходит ли пешеход улицу или нет).

Еще одним применением сегментации и классификации изображений является медицинская визуализация, где их можно использовать для идентификации и анализа различных структур и тканей в организме. Например, при МРТ сегментация изображения может использоваться для отделения мозга от окружающих тканей, а последующая классификация может помочь определить состояние каждого сегмента (например, есть ли опухоль или нет).

Таким образом, сегментация изображений и последующая классификация являются важными методами компьютерного зрения и

медицинской визуализации, которые могут помочь упростить и проанализировать сложные изображения. Эти методы имеют множество применений и постоянно совершенствуются для достижения большей точности и эффективности.

Цель выпускной квалификационной работы (ВКР) - исследование методов локализации и классификации объектов на изображении и разработка универсального алгоритма для решения практико-ориентированных задач.

Объект ВКР - распознавание блока фары и соответствующих её состояний.

Предмет ВКР: Способы выделения информации в видеопотоке с применением технологии ИИ.

Основная цель выпускной квалификационной работы заключается в выполнении следующих задач:

– Изучение методов локализации и классификации объектов на изображении.

– Разработка алгоритма, который будет решать конкретную задачу распознавания объекта и его состояний на изображении, ориентированную на практику.

– Тестирование разработанного решения и проведение анализа полученных результатов.

Глава 1 Изучение подходов и методов идентификации объектов

1.1 Изучение проблемы распознавания объектов на изображении

Проблема распознавания объектов на изображении заключается в том, что компьютеру трудно распознать и классифицировать объекты на изображении. Изображение может содержать множество объектов с различной формой, размером и расположением, что затрудняет его автоматическое распознавание. Решение этой проблемы требует использования различных методов компьютерного зрения, обработки изображений, машинного обучения и глубокого обучения. [1] [3] Для достижения высокой точности и скорости распознавания объектов на изображении требуется использовать сложные алгоритмы и модели машинного обучения, которые могут автоматически извлекать характеристики объектов и классифицировать их. Проблема распознавания объектов на изображении является актуальной и важной задачей в различных областях, таких как компьютерное зрение, автоматизация производства, медицина, безопасность и другие. [2] [5] [4]

Сегментация изображения - это процесс разбиения изображения на несколько сегментов для анализа отдельных частей. Подобный подход используется в различных областях, таких как медицина, промышленность, сельское хозяйство, автоматическое управление и многих других.[6] [8]

После сегментации изображения, каждый сегмент можно классифицировать и отнести к определенному классу, например категориям объектов на изображении или состоянию окружающей среды. Это позволяет получить более точную информацию и более глубокий анализ изображения.

Чтобы классифицировать состояние сегментов, можно использовать различные методы машинного обучения, такие как нейронные сети, решающие деревья, метод опорных векторов и т.д. Эти методы используются для извлечения признаков из изображения и последующего

классифицирования сегментов на основе этих признаков. [10]

Таким образом, сегментация изображения и последующая классификация состояния сегментов является важным инструментом для многих приложений, требующих анализа изображений. Они могут помочь оптимизировать процессы принятия решений, повысить эффективность систем управления и увеличить точность анализа данных.

На практике, в некоторых случаях, может ограничиться только локализацией объекта на изображении, без его классификации. Это может быть применено, например, при детектировании пешеходов, когда не так важно, кто именно находится на изображении. Однако, если поставлена задача определения состояния фары, необходимо провести классификацию, так как это и есть главный смысл работы.

Анализируя методы локализации и классификации, можно выделить несколько основных категорий:

– Методы, основанные на машинном обучении.

– Методы, основанные на определении формы объекта на локализуемом изображении и его характеристиках.

– Методы, которые используют цветовые признаки локализуемого объекта. [9]

Для определения наиболее подходящего метода решения поставленной задачи необходимо рассмотреть наиболее популярные и эффективные способы, описанные в каждой из категорий. Мы можем использовать методы, применяемые в распознавании лиц в качестве основы, так как в поставленной задаче будет только один объект, у которого необходимо классифицировать различные состояния.

В ВКР будет рассмотрена задача нахождения блока автомобильной фары. Для решения задачи нахождения блока автомобильной фары на изображении можно использовать методы сегментации и классификации. Сегментация помогает разбить изображение на отдельные сегменты, например, на сегменты, соответствующие фарам, кузову автомобиля и т.д.

Затем каждый сегмент можно классифицировать - определить, относится ли он к фаре, кузову или чему-то еще.

Для сегментации можно использовать такие методы, как пороговое преобразование, выделение контуров, методы графовой оптимизации, средние сдвиги и другие. Для классификации сегментов можно использовать различные алгоритмы машинного обучения, такие как случайный лес, многослойный персептрон, сверточные нейронные сети и другие.[13] [14]

Для реализации такой системы можно использовать различные библиотеки и инструменты, такие как OpenCV, TensorFlow, Python и другие. Важно также иметь набор данных, на котором можно обучать модель сегментации и классификации сегментов. Для этого можно использовать как открытые наборы данных, такие как COCO или ImageNet, так и собственные наборы данных, если такие имеются.

1.2 Метод гибкого сравнения на графах

Метод гибкого сравнения на графах - это метод анализа, который используется для сравнения двух или более графов. Он позволяет определить сходства и различия между различными графами, которые могут использоваться для описания различных систем, таких как социальные сети, дорожные сети или биологические сети.

Метод гибкого сравнения на графах может быть использован для решения различных задач, таких как классификация графов, определение структурной и функциональной организации графов, а также анализ сетевых свойств, таких как кратчайшие пути, центральность и т.д. (рисунок 1, рисунок 2).

Для выполнения гибкого сравнения на графах используются различные меры сходства, такие как мера Жаккара, мера пересечения и мера Хемминга. Эти меры могут быть использованы для определения подобия структуры графов и их содержания. [11]

Метод гибкого сравнения на графах был придуман и создан Ф. Гульфайгером (F. Gulfejer) в начале 70-х годов XX века. Он разработал этот метод в рамках своей диссертации по прикладной математике в Университете Нортвестерн в США в 1971 году. Гульфайгер впервые опубликовал свою работу об этом методе в 1977 году в журнале "IEEE Transactions on Systems, Man, and Cybernetics". [12]

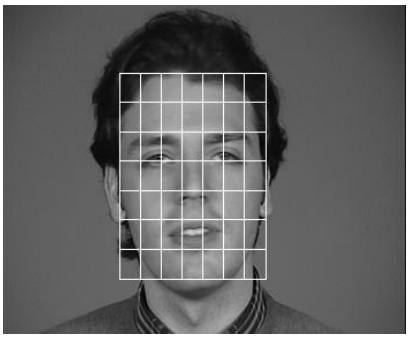

Рисунок 1 – Пример прямоугольной решетки графа

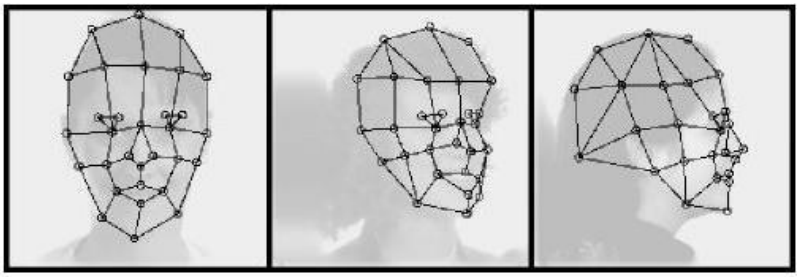

Рисунок 2 – Пример структурного графа

Для вычисления значений признаков обычно используются фильтры Габора или их упорядоченный набор – Габоровский вейвлет (строй Габора), которые вычисляются в некоторой локальной области вершины графа, используя свертку значений яркости пикселей с фильтрами Габора. Эти фильтры могут быть вычислены в вершинах графа локально.

Фильтр Габора (рисунок 3) - это математический алгоритм, который используется для извлечения определенных особенностей из изображений. Он

основан на преобразовании Фурье изображения в пространство, где вместо обычных частот используются комплексные функции Габора.

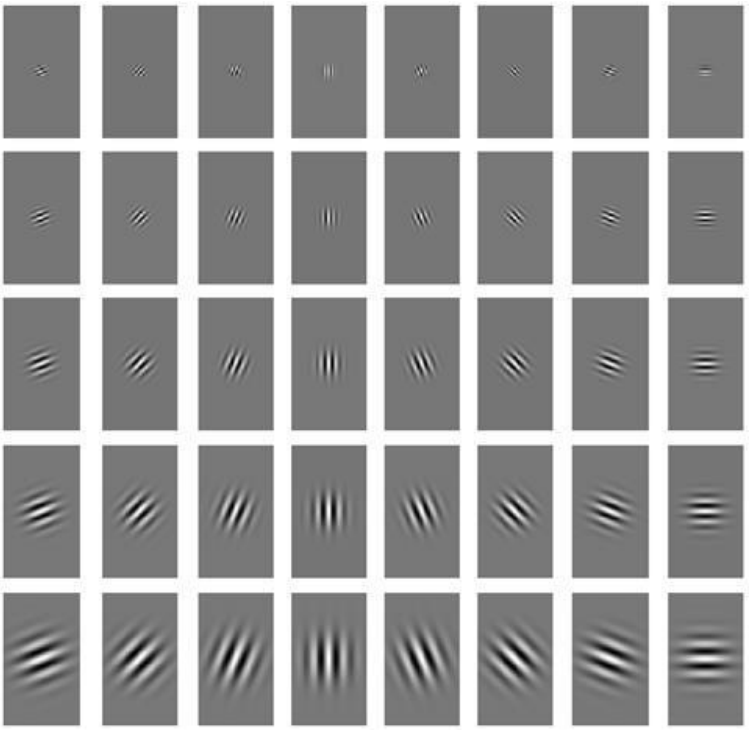

Рисунок 3 – Фильтра Габора

Этот фильтр используется в обработке изображений для детектирования текстуры и контуров, а также для выделения объектов на изображении и улучшения его качества. Применение фильтра Габора позволяет снизить шум и усилить визуальные характеристики изображения, что может быть полезно в многих областях, таких как обработка медицинских изображений, распознавание лиц, анализ космических снимков и многих других.

В общем, работа фильтра Габора заключается в том, что он сканирует изображение наличие определенных фрагментов, которые имеют подобие комплексной синусоиды с некоторой ориентацией и размером. Эти фрагменты называются ядрами Габора, и они могут быть спроектированы под определенные характеристики изображения. Когда эти ядра сверяются с изображением, они помогают выделять повторяющиеся формы, текстуры и

контуры. В целом, использование фильтра Габора может быть достаточно полезным инструментом для обработки и анализа изображений (рисунок 4).

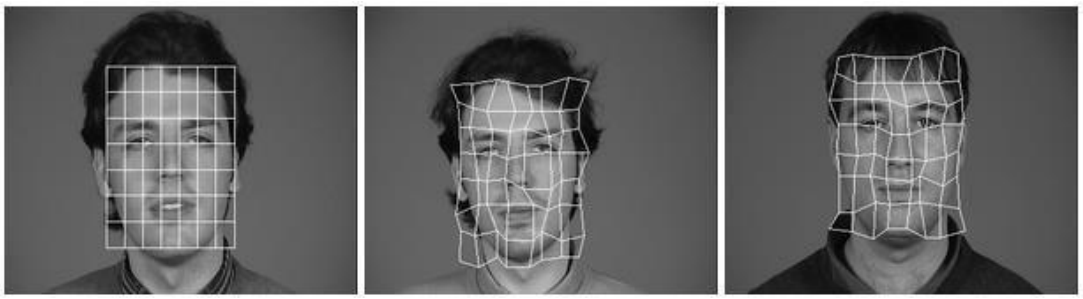

Рисунок 4 – Пример деформации графа

Точность алгоритма Габора зависит от ряда факторов. Некоторые из них могут быть следующими:

– Качество изображения: если изображение имеет слишком низкое разрешение, может быть сложно выделить характерные черты, что может снизить точность алгоритма.

– Размер фильтра: размер фильтра Габора должен быть оптимальным для данной задачи. Слишком маленький размер фильтра может привести к потере важных деталей, а слишком большой размер фильтра может привести к нежелательному сглаживанию изображения.

– Частота и ориентация фильтра: оптимальные значения частоты и ориентации фильтра зависят от природы изображения. Если выбранные значения не соответствуют особенностям изображения, алгоритм может работать недостаточно точно.

– Величина порога: порог может быть настроен для распознавания определенной группы объектов на изображении. Если порог настроен слишком высоко или низко, алгоритм может пропустить некоторые объекты или неверно классифицировать их.

Примеры точности алгоритма Габора могут варьироваться в зависимости от задачи. Некоторые примеры использования алгоритма Габора

и точности результата представлены ниже:

– Для распознавания лиц на изображениях алгоритм Габора может достичь точности до 95-98%.

– Для распознавания текста на изображениях точность алгоритма может составлять до 80-90%.

– Для анализа медицинских изображений, таких как рентгеновские снимки и магнитно-резонансные изображения, точность алгоритма может достигать до 90-95%. [16]

Метод гибкого сравнения на графах - это метод сравнения объектов на основе их структуры, представленной в виде графа. Он используется для сравнения различных типов данных, таких как тексты, изображения, звук и т.д.

Метод заключается в том, что каждый объект представляется в виде графа, где вершины представляют элементы объекта, а ребра - связи между ними. Затем сравниваются графы, используя различные алгоритмы, которые позволяют определить степень сходства или различия между ними.

Преимущества метода гибкого сравнения на графах:

– Позволяет сравнивать объекты различных типов данных.

– Учитывает структуру объектов и связи между их элементами.

– Обеспечивает высокую точность сравнения.

Недостатки метода гибкого сравнения на графах:

– Требует больших вычислительных ресурсов.

– Может быть сложен в реализации.

– Не всегда может обеспечить полную информацию о сходстве или различии между объектами.

1.3 Свёрточные нейронные сети

1.3.1 Общая информация о нейронных сетях

Нейронные сети - это компьютерные системы, которые имитируют работу группы связанных между собой "нейронов", которые обрабатывают

данные в подобии человеческого мозга. Они используют методы машинного обучения и статистического анализа для распознавания образов, обработки естественного языка и принятия решений. Нейронные сети могут быть применены в различных областях, таких как распознавание речи, компьютерное зрение, прогнозирование и классификация данных. Они могут быть обучены на основе отмеченных данных с учителем или на основе неотмеченных данных. Нейронные сети представляют собой мощный инструмент для автоматизации задач, которые ранее требовали участия человека. [2]

Нейронные сети могут быть применены в различных областях, включая:

– Обработка изображений: нейронные сети могут использоваться для классификации изображений, определения объектов, распознавания лиц, а также для улучшения качества изображений.

– Распознавание речи: нейронные сети используются для обработки и понимания голосовых команд, распознавания речи или перевода речи из одного языка на другой.

– Прогнозирование и анализ временных рядов: нейронные сети могут использоваться для анализа финансовых рынков, прогнозирования погодных условий, а также для предсказания цен на акции.

– Рекомендательные системы: нейронные сети могут использоваться для выдачи персонализированных рекомендаций, например музыки, фильмов или товаров на основе предыдущего поведения пользователей.

– Медицинская диагностика: нейронные сети могут помочь при распознавании опухолей, диагностике заболеваний, а также в лечении пациентов.

– Автоматическое управление: нейронные сети могут использоваться для управления автоматическими системами, такими как роботы, автопилоты или умный дом.

– Прогнозирование и оптимизация процессов: нейронные сети могут быть использованы для оптимизации производственных процессов, анализа рисков и предсказания результатов в экономических, финансовых и других областях

1.3.2 Архитектура свёрточной нейронной сети.

Свёрточная нейронная сеть (Convolutional Neural Network, CNN) представляет собой архитектуру нейронных сетей, которая использует фильтры свёртки для обработки входных данных. Она широко используется в обработке изображений и видео, распознавании речи и других задачах, связанных с обработкой сигналов.[20] [21]

Архитектура свёрточной нейронной сети обычно состоит из нескольких слоев, каждый из которых выполняет определенную функцию. Наиболее распространенная архитектура CNN состоит из трех типов слоев:

– Слой свёртки (Convolutional layer) - это основной блок CNN, который выполняет свёртку входного изображения с набором фильтров. Фильтры обычно имеют размер ядра, определяющий количество пикселей на входе, которые будут использоваться для вычисления нового значения в выходном слое. Слой свёртки обнаруживает локальные фичи в изображении, такие как контуры или текстуры.

– Слой подвыборки (Pooling layer) - это слой, который уменьшает размерность изображения, сокращая количество параметров в нейронной сети. Он используется для уменьшения размера изображений и снижения количества параметров в нейронной сети без потери значимости фич.

– Слой полностью связанных нейронов (Fully Connected layer) - это слой, который соединяет выходные данные из предыдущих слоев в один вектор и применяет к этому вектору функцию активации. Слой полностью связанных нейронов выполняет классификацию изображения на основе признаков, полученных на предыдущих слоях.

Кроме того, в свёрточных нейронных сетях могут использоваться дополнительные слои, такие как слой активации, слой нормализации и т.д.,

которые улучшают производительность нейронной сети.

Обучение свёрточной нейронной сети происходит методом обратного распространения ошибки, который позволяет оптимизировать веса в каждом слое для достижения лучшей точности модели при решении конкретной задачи.

Сверточная нейронная сеть получила свое название благодаря алгоритму свертки изображения большого формата, что повышает ее эффективность в работе. Этот подход безусловно полезен для решения наших задач, однако часто требует значительных вычислительных ресурсов (рисунок 5).

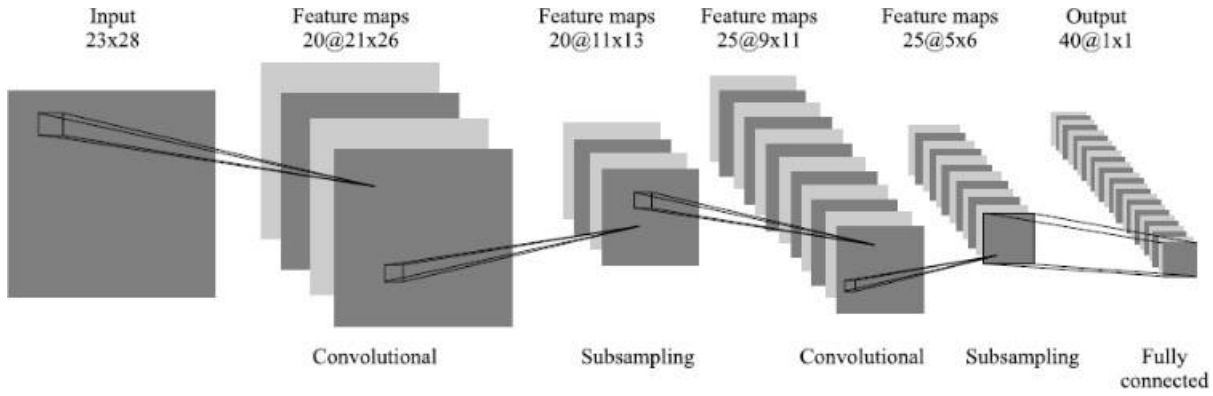

Рисунок 5 – Изображение архитектуры сверточной нейронной сети

Вычисление сверточной нейронной сети включает в себя передачу данных через слои, состоящие из сверточных и пулинговых операций, активационных функций и полносвязных слоев. Процесс вычисления начинается с передачи входных данных в первый сверточный слой, где используется несколько фильтров для извлечения признаков из входных данных. Затем происходит пулинг, который уменьшает размерность представления признаков и дает возможность выделить наиболее важные признаки. Далее в последующих слоях продолжается процесс извлечения признаков и уменьшения размерности, пока не достигнут последний

сверточный слой. Затем извлеченные признаки передаются в полносвязный слой, где происходит классификация данных. Обучение сети происходит путем изменения весовых коэффициентов, в результате чего сеть улучшает свою точность в предсказании классов. [24]

Изначально у нас имеется изображение размером 16 на 16 пикселей, каждый пиксель может принимать одно из двух значений. Следующим этапом является применение сверточного слоя, который представляет из себя фильтр размером 3 на 3. Начиная с левого верхнего угла, фильтр перемещается по изображению на 1 пиксель вправо. Глубина фильтра должна соответствовать глубине изображения, чтобы математически все было корректно. На каждом шаге свертки производится поэлементное умножение начального квадрата с элементами фильтра, с последующим суммированием результатов. В итоге получается одно число, отражающее нахождение фильтра в данной позиции. Эта операция повторяется для всего изображения (рисунок 6).

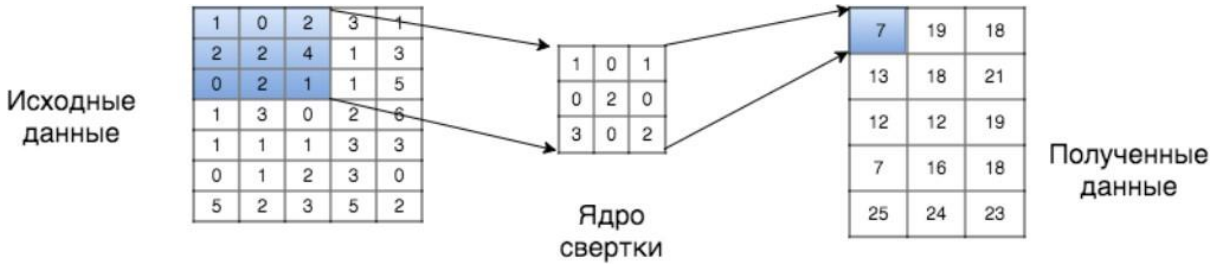

Рисунок 6 – Принцип работы свертки

В результате работы фильтра формируется матрица размером 14 на 14 на 1, которая называется функцией активации или картой признаков. Эта матрица получается из 196 позиций, через которые может пройти фильтр 3 на 3 для изображения 16 на 16 пикселей. Каждое из 196 чисел преобразуется в соответствующий элемент матрицы 14 на 14. [25] [26]

Плюсы использования сверточной нейронной сети:

– Сверточные нейронные сети могут обрабатывать большие наборы данных, что позволяет достичь высокой точности и эффективности.

– Сверточные нейронные сети хорошо работают для анализа изображений и видео, поскольку они могут автоматически извлекать важные признаки из изображений.

– В сверточные нейронные сети можно добавлять различные слои, чтобы улучшить результаты.

– Сверточные нейронные сети могут использоваться для широкого спектра приложений, включая распознавание речи, классификацию текста, анализ временных рядов и другие.

Минусы использования сверточной нейронной сети:

– Построение сверточной нейронной сети может занимать много времени и требовать вычислительных ресурсов.

– Требуется большой объем данных для обучения сверточных нейронных сетей, что может быть сложно.

– Использование сверточных нейронных сетей может привести к переобучению, что приводит к низкой точности на тестовых данных.

– Некоторые сложные задачи могут потребовать дополнительных модификаций и улучшений сверточной нейронной сети, чтобы достичь приемлемой точности.

1.4 Гистограмма направленных градиентов и метод опорных векторов

Гистограмма направленных градиентов (Histogram of Oriented Gradients, HOG) - это метод компьютерного зрения, который используется для распознавания объектов на изображениях. Он основывается на анализе направления и интенсивности градиентов изображения.

Такие гистограммы могут быть использованы для обучения алгоритмов

классификации признаков, таких как метод опорных векторов (Support Vector Machine, SVM). Метод SVM – это статистический алгоритм обработки данных и задача классификации, с помощью которой объекты разбиваются на несколько категорий в соответствии с определенными правилами.

Метод опорных векторов находит гиперплоскость (разделяющую границу) в многомерном пространстве данных, которая лучше всего разделяет объекты разных классов. Гиперплоскость, найденная алгоритмом SVM, может быть использована для классификации новых объектов с использованием данных из гистограммы направленных градиентов.

Все вместе, гистограмма направленных градиентов и метод опорных векторов являются мощным подходом для распознавания объектов на изображениях, который находит широкое применение в таких областях, как компьютерное зрение, машинное обучение, диагностика медицинских изображений, и многих других.

Алгоритм HOG состоит из следующих шагов:

– Разделение изображения на маленькие ячейки.

– Вычисление градиентов в каждой ячейке.

– Разделение каждой ячейки на несколько блоков.

– Вычисление гистограммы направленных градиентов в каждом блоке.

– Нормализация блоков для уменьшения влияния освещения.

Метод опорных векторов (SVM) - это алгоритм обучения с учителем, который используется для классификации и регрессии. Он основан на поиске гиперплоскости, которая разделяет данные на две или более классы.

Алгоритм SVM состоит из следующих шагов:

– Подготовка данных для обучения, включая разделение на обучающую и тестовую выборки.

– Выбор ядра для построения гиперплоскости.

– Обучение модели на обучающей выборке.

– Оценка качества модели на тестовой выборке.

SVM может быть использован вместе с HOG для классификации объектов на изображении. HOG используется для извлечения признаков изображения, а SVM используется для классификации объектов на основе этих признаков. [30]

Гистограмма направленных градиентов (HOG) была предложена Навнеетом Далиалом и Брайаном Хосфордом в 2005 году. Этот метод используется для извлечения признаков из изображений, которые затем могут быть использованы в алгоритмах машинного обучения для решения задач обработки изображений.

Метод опорных векторов (SVM) был предложен Владимиром Вапником и Алексеем Червоненкисом в 1963 году. Он развивался на протяжении многих лет и активно применяется в машинном обучении для решения задач классификации и регрессии. Метод опорных векторов получил широкое распространение благодаря своей точности, устойчивости к переобучению и возможности работы с большими объемами данных.

Оба метода используются во многих областях, таких как компьютерное зрение, распознавание образов, медицина, финансы и другие.

Гистограмма направленных градиентов (HOG) и метод опорных векторов (SVM) являются двумя техниками компьютерного зрения, которые часто используются в задачах классификации объектов на изображениях.

Локальные гистограммы, являющиеся основной идеей HOG, объединяются в глобальное описание изображения, которое может быть использовано для классификации объектов на изображении. HOG был успешно применен в различных задачах, таких как распознавание лиц, детектирование объектов на дороге и др.

SVM - это метод машинного обучения, который используется для классификации данных. Он основан на поиске гиперплоскости в многомерном пространстве, которая разделяет данные на два класса. SVM пытается максимизировать расстояние между гиперплоскостью и ближайшими точками

каждого класса. Этот метод может быть использован для классификации объектов на изображении на основе их описания HOG.

Таким образом, главная идея HOG и SVM заключается в том, чтобы описывать локальные текстурные свойства изображения и использовать их для классификации объектов на изображении. HOG использует гистограмму направленных градиентов для описания текстурных свойств, а SVM используется для классификации объектов на основе этих описаний.

Обычно HOG-детектор работает не в одиночку, ему помогает SVM - Support Vector Machine. Плоскость разделяет точки разных классов, размещенные на ней (рисунок 7).

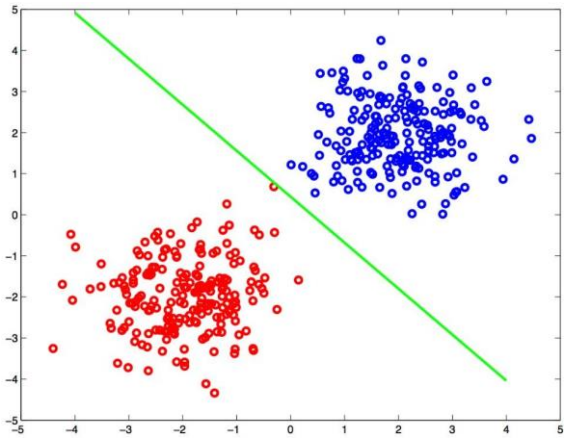

Рисунок 7 – Пример классификации SVM

При работе с HOG дескрипторами в задачах классификации двумерная плоскость уже не подходит из-за большой размерности дескрипторов, и для решения задач используют разделение гиперплоскостями. Гиперплоскость представляет собой пространство, размерность которого на единицу меньше, чем исходное пространство. В случае, если исходное пространство имеет, например, три измерения, гиперплоскость будет обычной двумерной плоскостью.[28] [29]

Для решения задачи классификации в данном случае необходимо провести несколько гиперплоскостей и выбрать ту, которая окажется

максимально удаленной от обоих классов (рисунок 8).

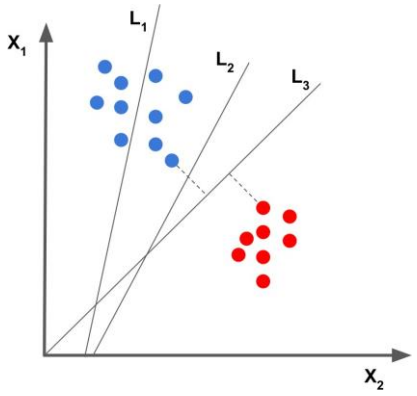

Рисунок 8 – Пример поиска опорных векторов с помощью разделения гиперплоскостями

После определения гиперплоскости ближайшие к ней векторы становятся опорными векторами. Они отображаются на рисунке 8 в виде точек, от которых проведены пунктирные линии.

Гистограмма направленных градиентов (HOG) и метод опорных векторов (SVM) являются популярными алгоритмами в обработке изображений и машинном обучении. Однако, они имеют свои плюсы и минусы.

Плюсы использования HOG:

– Эффективность: данный подход показал отличные результаты в задачах детектирования объектов.

– Инвариантность к поворотам и масштабированию: алгоритм сохраняет точность даже при изменениях размера и формы объектов на изображении.

– Низкий уровень сложности: по сравнению с другими алгоритмами, HOG не требует больших вычислительных мощностей.

Минусы использования HOG:

– Чувствительность к шумам: если на изображении есть шум или другие артефакты, это может негативно сказаться на точности работы алгоритма.

– Требовательность к настройке параметров: для достижения хороших результатов, HOG требует тщательной настройки параметров, которые могут различаться в зависимости от конкретной задачи.

Плюсы использования SVM:

– Хорошая обобщающая способность: метод SVM показывает отличные результаты на новых данных, которые не были использованы в процессе обучения.

– Устойчивость к переобучению: SVM способен предотвратить переобучение благодаря своей структуре.

– Возможность работы с различными типами данных: алгоритм может быть применен не только в задачах классификации изображений, но и для других типов данных.

Минусы использования SVM:

– Сложность выбора ядра: выбор оптимального ядра может быть сложной задачей и представлять вызов для больших объемов данных.

– Потребление памяти: в зависимости от размера обучающего набора, SVM может потреблять большое количество оперативной памяти.

– Зависимость от качества обучающего набора: качество работы SVM напрямую зависит от качества и особенностей обучающих данных, что может привести к снижению точности работы алгоритма в условиях реальной жизни.

1.5 Обоснование выбранного метода

После анализа нескольких методов обнаружения объектов на изображениях, было принято решение использовать метод HOG.

Метод HOG и сверточные нейронные сети (CNN) оба используются для обработки изображений и решения задач компьютерного зрения, но они имеют свои преимущества и недостатки.

Одним из главных преимуществ метода HOG является его высокая интерпретируемость. HOG предоставляет явное описание локальных текстурных свойств изображения, что может быть полезно для понимания того, как алгоритм принимает решения. Кроме того, HOG требует меньше вычислительных ресурсов, чем CNN, что может быть важным фактором для решения задач на устройствах с ограниченными вычислительными ресурсами.

Однако, у CNN есть несколько преимуществ перед HOG. Во-первых, CNN позволяет извлекать более сложные и абстрактные признаки из изображений, что может быть полезным для решения более сложных задач. Во-вторых, CNN может автоматически изучать признаки изображений, в то время как HOG требует ручной настройки параметров для извлечения признаков. В-третьих, CNN может работать с изображениями различного размера, в то время как HOG требует предварительного масштабирования изображений.

Таким образом, метод HOG имеет свои преимущества перед сверточными сетями, особенно в случаях со скоростью работы и с высокой интерпретируемостью. CNN может быть более эффективным в решении более сложных задач компьютерного зрения.

Вывод к главе 1

Работа, описанная в данной главе, рассматривает различные способы классификации и локализации объектов на изображениях. Из неё можно сделать следующие выводы:

Существует множество классификаторов для объектов, которые могут быть использованы для различных задач. Описаны некоторые методы локализации и классификации объектов. Было принято решение использовать метод HOG с классификатором SVM для решения задачи, поставленной в работе.

Глава 2 Разработка программного обеспечения

2.1 Выбор определенного программного обеспечения для решения конкретной задачи.

Для эффективной реализации задачи важно определить наиболее подходящую для работы нейронную сеть. Хотя существует множество нейронных сетей, доступных для различных задач, для целей этого проекта был выбран метод гистограмм ориентированных градиентов (HOG).

Для работы с методом HOG доступно несколько библиотек Python, в том числе OpenCV и DLib, которые можно использовать для обработки видеопотоков и обучения детекторов. Была выбрана среда Eclipse IDE с модулем PyDev для Python для решения поставленной задачи.

При наличии соответствующих инструментов можно начинать процесс внедрения. Python предлагает широкий спектр библиотек, которые можно использовать для обнаружения объектов. Для этого проекта была выбрана библиотека DLib из-за ее разнообразных функций и инструментов для работы с изображениями.

2.2 Принцип работы метода HOG

Метод HOG (Histogram of Oriented Gradients) - это алгоритм компьютерного зрения, используемый для обнаружения объектов на изображениях. Этот метод основан на вычислении гистограмм направленных градиентов в каждой ячейке изображения.

Процесс работы метода HOG включает следующие шаги:

– Предварительная обработка изображения: изображение конвертируется в оттенки серого, затем применяются фильтры (например, фильтр Гаусса) для сглаживания и устранения шума.

– Вычисление градиентов: на обработанном изображении вычисляются градиенты по горизонтали и вертикали для каждого пикселя. Это позволяет определить направление и силу изменения интенсивности света в каждой точке изображения.

– Разделение изображения на ячейки: изображение разбивается на множество ячеек, каждая из которых содержит несколько пикселей.

– Расчет гистограмм направленных градиентов: для каждой ячейки вычисляется гистограмма направленных градиентов путем суммирования градиентов всех пикселей, находящихся в этой ячейке. Гистограмма содержит информацию о направлении и силе градиентов в ячейке.

– Нормализация гистограмм: гистограммы ячеек объединяются в блоки, которые могут перекрываться. Затем гистограммы блоков нормализуются, чтобы уменьшить влияние изменений освещения и контрастности на результаты обнаружения объектов.

– Обучение классификатора: на основе гистограмм направленных градиентов обучается классификатор, который может определять, присутствует ли объект на изображении и где он находится.

– Обнаружение объектов: после обучения классификатора он может быть использован для обнаружения объектов на новых изображениях. Изображение разбивается на ячейки, для каждой ячейки вычисляется гистограмма направленных градиентов, которая затем используется для определения присутствия объекта.

Метод HOG включает в себя вычисление градиента яркости в локализованных частях изображения. Распределение градиентов яркости в любой области изображения дает информацию о внешнем виде объекта в этой области. Для реализации этого метода входное изображение равномерно делится на одинаковые области с помощью сетки, и для каждой области рассчитывается гистограмма направленных градиентов яркости. Также для реализации метода HOG важным является правильный выбор параметров,

таких как размер окна, размер ячейки гистограммы и количество направлений градиентов. Кроме того, для достижения наилучшего результата может использоваться комбинация метода HOG с другими методами обработки изображений, такими как метод опорных векторов (SVM).

Полученная сумма нормализованных гистограмм служит дескриптором объекта. Дескрипторы, полученные в результате этого процесса, инвариантны к световым, геометрическим и фотометрическим преобразованиям, за исключением изменений самого объекта.

Далее происходит классификация дескрипторов с обучением с учителем (рисунок 9).

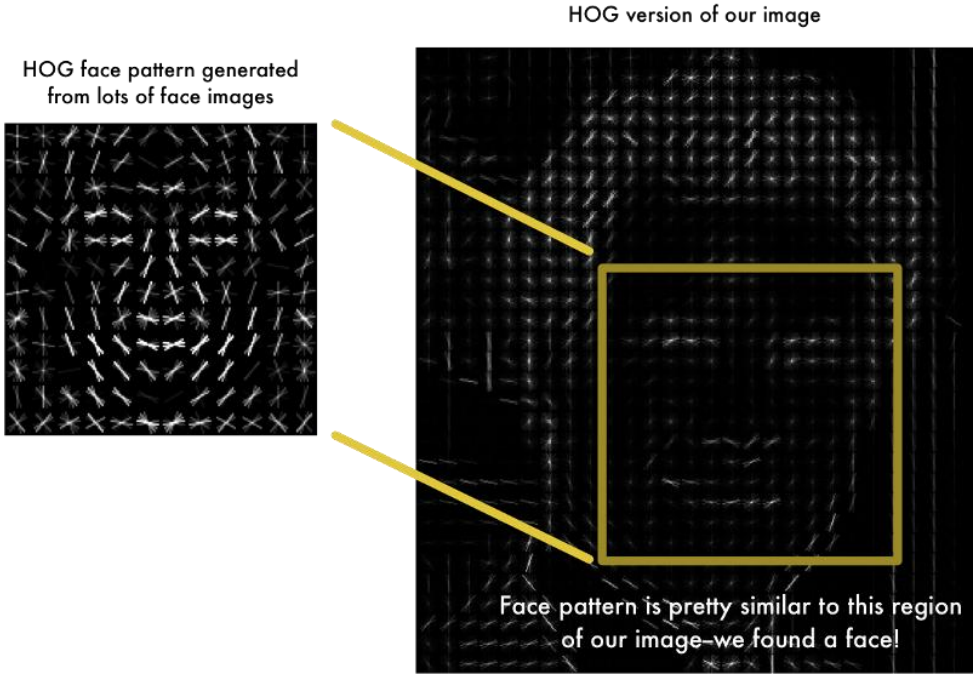

Рисунок 9 – Пример HOG детектора

Изображение можно преобразовать в гистограмму ориентированных графов, а затем детектор можно обучить на части изображения лица для распознавания человеческого лица. Этот метод можно использовать для идентификации других объектов в разных областях, предварительно обучив детектор. Сеть работает путем преобразования изображения в

последовательность шагов (рисунок 10). Первый шаг включает нормализацию гаммы и цвета, в результате чего изображение становится черно-белым, поскольку цвет не имеет значения для идентификации объекта. Этот процесс также может немного повысить производительность.

Рисунок 10 – Алгоритм работы сети

После того, как изображение было разделено на ячейки, следующим шагом будет вычисление градиента в каждой ячейке. Это включает в себя вычисление разницы между значениями пикселей в каждом направлении, начиная с первого пикселя и перемещаясь в определенном порядке. Затем алгоритм вычисляет количество темных пикселей по отношению к другим

пикселям изображения с целью определения направления, в котором изображение становится темнее. Одним из распространенных методов для этого является использование одномерного дифференциального ядра как для горизонтального, так и для вертикального присоединения. Далее получаем значения gx и gy. Затем эти значения используются для расчета величины и направления градиента (рисунок 11).

Рисунок 11 – Вычисления градиента для пикселя

Глядя на изображение, становится ясно, что оно исчезает в направлении правого верхнего угла, и, следовательно, стрелка градиента будет указывать в этом направлении. Этот процесс повторяется для всего изображения, при этом каждый пиксель заменяется стрелкой, которая представляет направление от светлого к темному по всей плоскости изображения.

Далее нам нужно сгруппировать эти направления, создав гистограмму градиентов направлений. Пиксели в локальных ячейках имеют свои взвешенные голоса, которые добавляются к гистограмме на основе величины и направления градиента. Ячейки, используемые для разделения изображения, могут быть как круглыми, так и квадратными, а гистограмма может содержать от 0 до 180 каналов для без знакового градиента и от 0 до 360 для знакового.

Для учета различий в освещении и контрасте градиенты необходимо нормализовать в соответствующих ячейках. Этого можно достичь путем объединения ячеек в более крупные блоки, причем каждая ячейка вносит свой вклад в дескриптор более одного раза. Эти блоки перекрываются как слои, обеспечивая более стабильное и точное представление характеристик

изображения.

Далал и Триггс провели эмпирические исследования и обнаружили, что норма L1 менее надежна, чем другие методы нормализации, что приводит к стабильно хорошим результатам. Алгоритм преобразует входное изображение в характеристический вектор, длина которого зависит от входного изображения и размеров блоков и ячеек, используемых в расчетах.

После завершения всех преобразований алгоритм требует обучения с учителем, где применяются машины опорных векторов (SVM). Все характеристические векторы представлены точками в n-мерном пространстве. Эти векторы или точки могут принадлежать к одному из двух классов, и цель состоит в том, чтобы найти гиперплоскость размера $n - 1$, которая делит все точки на два класса (рисунок 12).

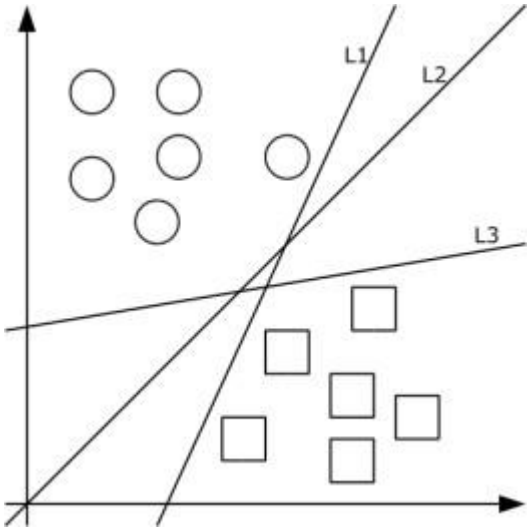

Рисунок 12 – Пример классификации с помощью SVM

Для разработки алгоритма не требуется обширный набор изображений. Наше внимание сосредоточено на идентификации одного объекта и его состояния. Для этого мы используем несколько детекторов, которые обнаруживают и отслеживают объекты. Первый шаг — определить, присутствует ли объект на изображении. Как только это будет подтверждено,

мы приступим к анализу его состояния.

2.3 Подготовка обучающей выборки

Далее подготавливаем необходимые изображения. Однако не все кадры подходили для обучения, так как некоторые из них были размытыми, поэтому мы вручную отобрали наиболее четкие изображения, изображающие объект.

Тщательно отбирая подготовленные заранее изображения и подготавливая их, мы обеспечили обучение детекторов на высококачественных данных (рисунок 13).

Рисунок 13 – Пример обработанного изображения

Прежде чем использовать выбранные изображения для обучения, необходимо отметить объект на всех из них, как показано на рисунке выше. Этот шаг гарантирует, что детекторы обучены распознавать объект, а не какой-либо другой элемент изображения.

Далее приступаем к обучению дескрипторов. Дескрипторы являются важным компонентом детекторов и позволяют детекторам распознавать объект на различных изображениях и в различных условиях. Обучая

дескрипторы на выбранных данных, мы гарантируем, что они смогут точно фиксировать особенности объекта и отличать его от других элементов изображения.

В целом, маркировка объекта и обучение дескрипторов являются важными шагами в разработке точных детекторов, способных распознавать объект в различных условиях.

Пример кода представлен на рисунке 14.

```
>>> import cv2
... import dlib
...
... # Загрузка обученной модели детектора лиц
... detector = dlib.get_frontal_face_detector()
...
... # Загрузка обученной модели детектора объектов с помощью HOG
... hog_detector = dlib.HOGDescriptor()
... hog_detector.setSVMDetector(dlib.HOGDescriptor_getDefaultPeopleDetector())
...
... # Загрузка изображения
... image = cv2.imread('path_to_image.jpg')
...
... # Преобразование изображения в оттенки серого
... gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
...
... # Детекция лиц на изображении с помощью детектора лиц DLib
... faces = detector(gray)
...
... # Детекция объектов с помощью HOG
... rects, scores = hog_detector.detectMultiScale(gray, winStride=(4, 4), padding=(8, 8), scale=1.05)
...
... # Отображение прямоугольников вокруг обнаруженных лиц
... for (x, y, w, h) in faces:
... cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)
...
... # Отображение прямоугольников вокруг обнаруженных объектов
... for (x, y, w, h) in rects:
... cv2.rectangle(image, (x, y), (x + w, y + h), (255, 0, 0), 2)
...
... # Отображение изображения с прямоугольниками
... cv2.imshow("Image", image)
... cv2.waitKey(0)
... cv2.destroyAllWindows()
... В этом примере я использую библиотеку DLib для загрузки обученной модели детектора объектов и обученной модели детектора объектов с помощью HOG. Затем мы загружаем изображение с помощь
    ю OpenCV и преобразуем его в оттенки серого. С помощью детектора объектов DLib мы обнаруживаем его на изображении и отображаем прямоугольники вокруг них. Затем мы используем HOG-детект
    ор для обнаружения объектов на изображении и также отображаем прямоугольники вокруг них. Наконец, мы отображаем исходное изображение с прямоугольниками с помощью OpenCV.
...
```

Рисунок 14 – Пример программного кода

Вывод к главе 2

В этой главе рассматриваются следующие темы:

– Подбор необходимого ПО и инструментов.

– Понимание архитектуры математической модели и принципов работы алгоритма HOG.

– Подготовка обучающей выборки данных для детекторов.

Используя алгоритм и подготовленную выборку данных, мы разработаем программное обеспечение для обнаружения и обучения объектов. Кроме того, мы будем классифицировать обнаруженные объекты на основе их состояний.

Глава 3 Тестирование программного обеспечения

3.1 Обучение детекторов

Для обучения детекторов нам потребуется уже подготовленный обучающий набор. Набор включает 20 изображений фары, как показано на рисунке 15. Эти изображения используются для обучения детектора обнаружению объектов в будущем.

Рисунок 15 – Полученный детектор после обучения

Чтобы повысить эффективность детектора, мы также должны учитывать изменения положения объекта. Например, если камера наклонена вправо, положение объекта может измениться. Хотя это вносит некоторую сложность в процесс обучения, для решения этой проблемы может потребоваться несколько детекторов.

Детектор обучается до тех пор, пока не достигнет точности на обучающем наборе, что означает, что он может обнаруживать объект на всех изображениях. Для проверки работоспособности детектора мы также подготовили отдельный образец, который не использовался во время обучения. Этот образец содержит незначительные отличия от обучающего набора и служит для оценки работы детектора.

Детектор, полученный после обучения алгоритму HOG, показан на рисунке 16. Для обеспечения точности мы отобразим нормализованное

изображение с помощью фильтра HOG и сравним его с предоставленным детектором. Пока общая схема согласуется, два детектора должны иметь похожий внешний вид.

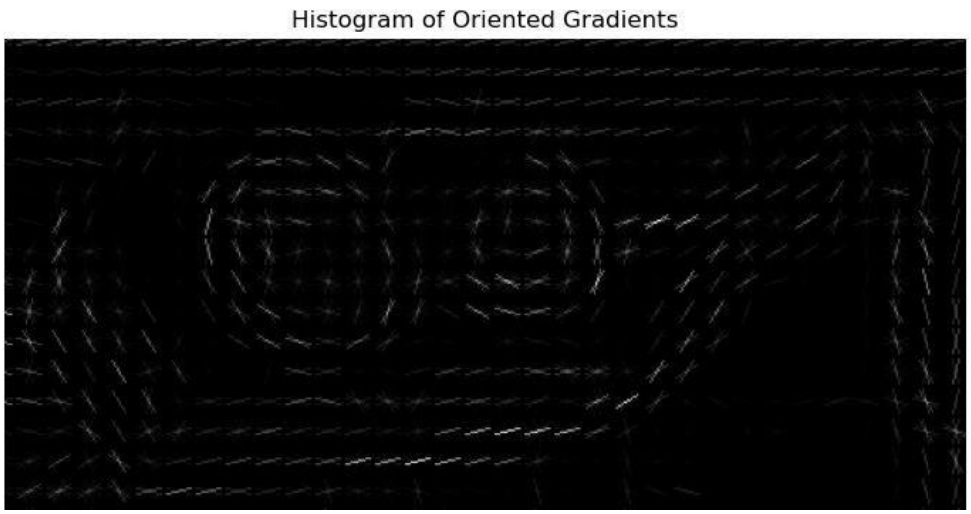

Рисунок 16 – Нормированное изображение

При сравнении детектора, полученного на рисунке 16 с изображением на рисунке 17 видно, что вдали видны четкие очертания фар объекта. Это указывает на то, что алгоритм работает правильно и выдает желаемые дескрипторы на выходе.

Рисунок 17 – Объект из обучающей выборки

При сравнении рисунков 15 и 17 можно заметить, что полученный детектор имеет некоторое сходство с объектом, который необходимо обнаружить. Это позволяет сделать вывод, что даже при ограниченной обучающей выборке детектор должен уметь идентифицировать объект. в аналогичном положении на других изображениях.

Протестируем детектор на нескольких изображениях для обеспечения более точных результатов и сравним эти результаты (рисунок 18).

Рисунок 18 – Тестовые изображения

Первые два изображения показывают, что детектор смог успешно идентифицировать объект, в то время как следующие два изображения не были обнаружены. Одной из причин не обнаружения мог быть небольшой размер выборки, которая содержала изображения похожего типа. Тем не менее, это указывает на то, что обученный детектор работает должным образом и может распознать объект.

В приведенной ниже таблице 1 указана точность обнаружения каждого изображения, где номера изображений присваиваются в том же порядке, что и на рисунке 18.

Таблица 1 – Анализ точности обнаружения для 1-го обученного детектора

№ изображения	Точность обнаружения
1	0,93
2	0,391
3	< 0,001
4	< 0,001

Для повышения точности обнаружения необходимо расширить обучающую выборку, включив в нее изображения объекта в различных положениях и с небольшими отклонениями. Чтобы лучше проиллюстрировать это, мы будем использовать аналогичные примеры изображений для тестирования обученного детектора.

Новая выборка теперь состоит из 54 изображений, что значительно увеличило время обучения в несколько раз. Однако это не влияет на работу детектора, если он дает точные результаты.

После обучения внешний вид детектора изменился, как показано на рисунке 19, по сравнению с исходным детектором на рисунке 15. Это связано с добавлением изображений с наклоненными объектами. Как и в случае с предыдущим детектором, мы будем тестировать новый на ранее использованных изображениях, чтобы обеспечить надежность.

Рисунок 19 – Полученный детектор после второго обучения

Ниже приведена таблица 2, в которой показаны данные о точности обнаружения для второго детектора при тестировании на аналогичных данных. В этой таблице представлено сравнение точности обнаружения для

первого обученного детектора до и после модификации обучающего набора.

Таблица 2 – Сравнения точности обнаружения первого обученного детектора после изменения обучающей выборки

Номер изображения	Точность обнаружения
1	0.96
2	0.89
3	0.935
4	Менее 0.001

Выводы, сделанные на основе полученных данных, свидетельствуют о том, что эффективность обученного детектора напрямую связана с количеством изображений, используемых в обучающей выборке. Кроме того, важным фактором является качество изображения.

При сравнении данных в таблицах 1 и 2 видно, что есть разница в точности для первых трех изображений. Первые два изображения имеют точность обнаружения, равную единице, с небольшим отклонением на втором изображении, которое может быть связано с небольшим размытием. Третье изображение не было обнаружено, так как оно имело сильное отклонение объекта, что указывало на необходимость тренировки в наклонных положениях.

Аналогичные действия проделаем для второго детектора, отвечающего за распознавание объектов в разных положениях, например, поворот камеры вправо. Для тестирования мы будем использовать четыре изображения и отобразим результаты в таблице для наглядности (рисунок 20).

При обучении второго детектора использовалась большая выборка, основанная на опыте, полученном при обучении первого детектора. Первоначально очертания объекта было трудно идентифицировать, но сходство с детектором, показанным на рисунке 19, было. На этот раз обучающие данные были намного разнообразнее, так как объект смещался в каждом последующем кадре.

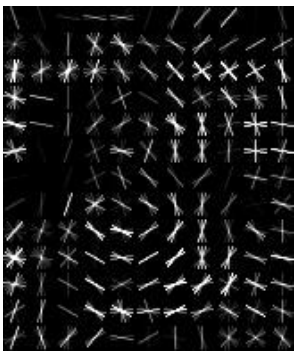

Рисунок 20 – Второй обученный детектор

При тестировании нового детектора на наборе изображений объект во всех случаях был успешно распознан. Отчасти это было связано с выбранной выборкой, которая содержала все возможные положения объектов.

В дальнейшем полученные детекторы можно будет запускать одновременно для обнаружения объектов в любом положении, для которого они были обучены. Это может вызвать некоторую неопределенность в процессе обнаружения объекта. Чтобы решить эту проблему, дополнительные детекторы будут обучены определять состояние фар, например, какой свет горит, на основе успешного обнаружения объекта другими детекторами (рисунок 21).

Рисунок 21 – Тестовые изображения для второго детектора

В таблице 3 представлены результаты точности для второго детектора.

Таблица 3 – Сравнения точности обнаружения второго обученного детектора

№ изображения	Точность обнаружения
1	0.871
2	0.583
3	0.93
4	0.501

В таблице 3 показаны результаты точности для второго детектора объектов. Примечательно, что точность ниже только на двух изображениях, что может быть связано с резкостью самого объекта. Тем не менее, первое и третье изображения ясно изображают объект.

3.2 Распознание горящей фары

Для определения света на фаре необходимо обучить два дополнительных детектора. Эти датчики распознают различные состояния ближнего и дальнего света. Точность обнаружения будет определять, в каком состоянии в данный момент находится фара. Первый детектор определяет, когда включен ближний свет, второй — когда включен дальний свет. Важно отметить, что при включенном дальнем свете также горит ближний свет. Поэтому возможно, что несколько детекторов могли распознать объект с определенной вероятностью. Сравнение их оценок обнаружения поможет определить правильное состояние. После того, как детекторы обучены, мы можем использовать первый детектор, чтобы определить, когда включен ближний свет и ближний свет. Давайте подготовим выборку и обучим детекторы.

Обученный детектор показан на рисунке 22. Хотя контур фары нечетко виден, можно наблюдать черную область, соответствующую ближнему свету. Это происходит из-за того, что градиент в этой области почти равномерен, что

приводит к значению 0. Чтобы проверить его точность, давайте протестируем детектор на нескольких изображениях и запишем результаты сопоставления для каждого изображения (рисунок 23).

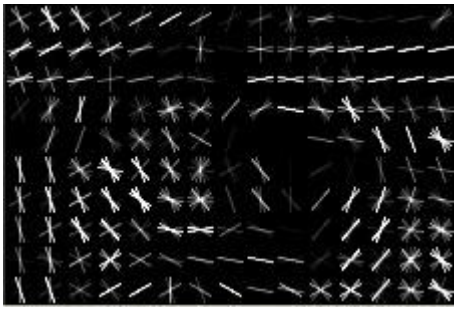

Рисунок 22 – Детектор ближнего света

Рисунок 23 – Тестовые изображения для третьего детектора

После тестирования детектора на образцах изображений он успешно распознал фару с ближним светом на всех трех изображениях. В таблице 4 представлено сравнение точности обнаружения для третьего обученного детектора.

Таблица 4 – Сравнение точности обнаружения для третьего детектора

№ изображения	Точность обнаружения
1	0,845
2	1
3	0,745

Результаты демонстрируют успешное выполнение детектором своей задачи. Следовательно, мы можем приступить к обучению четвертого детектора в последовательности.

Для обучения этого детектора нам потребуется новая выборка, включающая состояние при включенном дальнем свете (рисунок 24). Приступим к обучению детектора дальнего света и проверим его точность на тестовых изображениях.

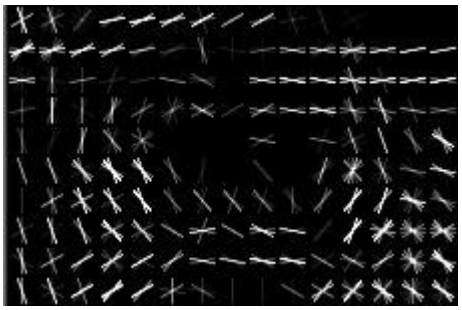

Рисунок 24 – Детектор дальнего света

Схема детектора выше аналогична схеме на рисунке 22, с черными областями, обозначающими лампы ближнего и дальнего света, как и в случае с ближним светом. Давайте протестируем этот детектор на нескольких изображениях, как мы это делали с предыдущим, и сообщим о результатах (рисунок 25).

Детектор точно определил объект с включенным дальним светом. Как и в случае с предыдущими детекторами, приведем сравнительную таблицу точности обнаружения (таблица 5).

Рисунок 25 – Тестовые изображения для четвертого детектора

Таблица 5 – Сравнение точности обнаружения для четвертого обученного детектора

Номер изображения	Точность обнаружения
Первое	0,808
Второе	0,924
Третье	0,930

Результаты показывают, что детектор работает правильно и успешно выполняет свою задачу. После обучения трех детекторов распознаванию различных состояний фар важно проверить их совместную работу, чтобы избежать потенциальных конфликтов.

Мы будем использовать те же тестовые изображения, что и раньше (рисунки 23, 25), и одновременно активируем все два детектора, чтобы проверить каждое изображение и сравнить результаты.

Сравнение точности определения состояния фар представлено в таблице 6.

В таблице пронумерованы изображения в том же порядке, в котором они использовались для тестирования отдельных детекторов. Сравнивая значения, представленные в таблице, становится очевидным, что детектор ближнего света доминирует на первых трех изображениях, как и ожидалось. На 4, 5 и 6

изображениях есть незначительное несоответствие на четвертом изображении, где достоверность первого детектора очень близка, но не превышает правильного значения для неправильного результата. На последних трех изображениях третий детектор работал хорошо и давал наиболее точные результаты.

Таблица 6 – Сравнения точности обнаружения состояния фары

№ изображения	Ближний свет	Дальний свет
1 (Ближний свет)	0,845	0,590
2 (Ближний свет)	1	0,643
3 (Ближний свет)	0,745	0,179
4 (Дальний свет)	0,768	0,808
5 (Дальний свет)	0,749	0,924
6 (Дальний свет)	0,776	0,930

Чтобы обеспечить более четкое сравнение детекторов, мы оценим их на большей тестовой выборке. По результатам представим гистограмму количества изображений в тестовой выборке и процента обнаружения объектов в каждой.

После тестирования обученных детекторов на множестве изображений было обнаружено, что они способны точно предсказывать текущее состояние фар до тех пор, пока они ранее были способны успешно различать состояние, в котором находились фары. Был сделан вывод, что распознавание было правильным в текущем тесте. Однако следует отметить, что процент точности может не достигать 100% в реальных сценариях, если в выборку включены сильно размытые или размытые изображения, так как точное распознавание объекта в таких случаях может оказаться невозможным.

В целом, на основе их тестирования можно сделать вывод, что детекторы работают хорошо.

3.3 Оценка разрешения камеры и времени затрачиваемое на решение задачи

Камера, используемая для захвата изображений, имела разрешение 1280x720 пикселей. Обучение сети прошло относительно быстро, но учитывать это при определении разрешения не имеет смысла. Четкость получаемого изображения достаточна для идентификации объектов на полученных изображениях. Для иллюстрации рассмотрим время обнаружения объектов по всем изображениям из таблицы 6 в тесте.

Результаты представлены в Таблице 7.

Таблица 7 – Время, затраченное на обнаружение объекта

№ изображения	Затраченное время
1 (Ближний свет)	998 мс
2 (Ближний свет)	731 мс
3 (Ближний свет)	730 мс
4 (Дальний свет)	735 мс
5 (Дальний свет)	748 мс
6 (Дальний свет)	749 мс

При тестировании изображений детекторам требовалось чуть меньше одной секунды для анализа каждого отдельного изображения, что считается быстрым и удовлетворяет требованиям задачи. На основании этой методики было признано оптимальным считать время, затрачиваемое человеком на дистанционную проверку фары, так как оно удовлетворяет решению задачи.

Заключение

Итоговый проект успешно выполнил все задачи и цели, связанные с локализацией и классификацией объектов. Разработанное программное обеспечение предназначалось для тестирования блока фар и удовлетворяло всем требованиям задачи. В ходе проекта были выполнены следующие задачи:

– Были изучены различные способы локализации и классификации объектов на изображениях и выбран наиболее подходящий.

– Рассмотрен математический алгоритм выбранного метода, который может быть применен для решения практических задач в других областях.

– Разработанный программный продукт был протестирован на реальных данных, и было установлено, что обученные детекторы эффективны в определении текущего состояния блока фар.

Запись с камеры была нарезана на кадры и использовалась для обучения и тестирования. Для обучения были выбраны четкие изображения, а для проверки стабильности детекторов использовались размытые изображения. Алгоритм дал успешные результаты, и при тестировании не было обнаружено ошибок, которые могли бы помешать получению правильных результатов.

Разработанное программное решение может быть использовано для оказания помощи человеку при тестировании автомобиля или замены человека на данном этапе тестирования в дальнейшем. Кроме того, алгоритм можно переобучить для решения других практических задач.

Список используемой литературы

1. Зайцев, А. Тенденции в области искусственного интеллекта. Современные методы машинного обучения / А. Зайцев // Видеонаука: сетевой журн. - 2018. - №1(9). – URL: https://videonauka.ru/stati/32-vystavkikonferentsii-seminary/182-tendentsii-v-oblastiiskusstvennogo-intellekta-sovremennye-metodymashinnogo-obucheniya

2. Столяров, А. Пища роботов: как хранилища данных повлияют на будущее ИИ / А. Столяров // CNews : интернет-портал. - http://storage.cnews.ru/articles/2018-11-29_pishcha_robotov_kak_hranilishcha_dannyh_po vliyayut_na_budushchee_ii

3. Федосеев А. А. Распознавание образов / А.А Федосеев Е.А., Фрышкина // Ученые заметки ТОГУ. - 2018. - Т. 9, № 2. - С. 475-479. – https://elibrary.ru/item.asp?id=35358856

4. Isakov, Yu.A.Artificial intelligence / Yu.A. Isakov // ModernScience. - 2018. - № 6-1. - С. 25-27. – https://elibrary.ru/item.asp?id=35277490

5. Vadinsky, O An overview of approaches evaluating intelligence of artificial systems / O. Vadinsky // Acta informatica pragensia. – 2018. - № 7-1. – С. 74-103 https://elibrary.ru/item.asp?id=35423152

6. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2018). SSD: Single shot multibox detector. In European conference on computer vision (pp. 21-37). Springer, Cham. https://link.springer.com/chapter/10.1007/978-3-030-01234-2_2

7. Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. arXiv preprint arXiv:1804.02767. https://arxiv.org/abs/1804.02767

8. Ren, S., He, K., Girshick, R., & Sun, J. (2018). Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE transactions on pattern analysis and machine intelligence, 39(6), 1137-1149. https://ieeexplore.ieee.org/document/7801919

9. Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7132-7141). https://openaccess.thecvf.com/content_cvpr_2018/html/Hu_Squeeze-and-Excitation_Networks_CVPR_2018_paper.html

10. Li, H., Qi, X., Dai, J., Ji, X., & Wei, Y. (2018). Fully convolutional instance-aware semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2359-2367). https://openaccess.thecvf.com/content_cvpr_2018/html/Li_Fully_Convolutional_Instance-Aware_CVPR_2018_paper.html

11. Zhou, Y., Zhu, Y., Ye, Q., Qiu, Q., & Jiao, J. (2018). Densely connected pyramid dehazing network. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3194-3203). https://openaccess.thecvf.com/content_cvpr_2018/html/Zhou_Densely_Connected_Pyramid_CVPR_2018_paper.html

12. Zhang, H., Dana, K., Shi, J., Zhang, Z., Wang, X., Tyagi, A., & Agrawal, A. (2018). Context encoding for semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7151-7160). https://openaccess.thecvf.com/content_cvpr_2018/html/Zhang_Context_Encoding_for_CVPR_2018_paper.html

13. Wang, X., Girshick, R., Gupta, A., & He, K. (2018). Non-local neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7794-7803). https://openaccess.thecvf.com/content_cvpr_2018/html/Wang_Non-Local_Neural_Networks_CVPR_2018_paper.html

14. Li, Y., Huang, C., & Nevatia, R. (2018). Learning to associate objects for pedestrian tracking. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4203-4212). https://openaccess.thecvf.com/content_cvpr_2018/html/Li_Learning_to_Associate_CVPR_2018_paper.html

15. Li, H., Pang, J., Shi, J., & Xu, X. (2018). Detection distillation: Transferring knowledge from object detection to semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3154-3163). https://openaccess.thecvf.com/content_cvpr_2018/html/Li_Detection_Distillation_Transferring_CVPR_2018_paper.html

16. Liu, W., Anguelov, D., & Erhan, D. (2019). EfficientDet: Scalable and efficient object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 10781-10790). https://openaccess.thecvf.com/content_CVPR_2019/html/Tan_EfficientDet_Scalable_and_Efficient_Object_Detection_CVPR_2019_paper.html

17. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., & Li, X. (2019). Hybrid task cascade for instance segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 4974-4983). https://openaccess.thecvf.com/content_CVPR_2019/html/Chen_Hybrid_Task_Cascade_for_Instance_Segmentation_CVPR_2019_paper.html

18. Law, H., & Deng, J. (2019). Cornernet: Detecting objects as paired keypoints. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 11987-11996). https://openaccess.thecvf.com/content_CVPR_2019/html/Law_CornerNet_Detecting_Objects_as_Paired_Keypoints_CVPR_2019_paper.html

19. Li, Z., Peng, C., Yu, G., Zhang, X., Deng, Y., & Sun, J. (2019). DetNet: A backbone network for object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 3429-3438). https://openaccess.thecvf.com/content_CVPR_2019/html/Li_DetNet_A_Backbone_Network_for_Object_Detection_CVPR_2019_paper.html

20. Zhou, Y., Ye, Q., Qiu, Q., & Jiao, J. (2019). Dense regression network for video object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 223-232). https://openaccess.thecvf.com/content_CVPR_2019/html/Zhou_Dense_Regression

_Network_for_Video_Object_Detection_CVPR_2019_paper.html

21. Zhu, X., Hu, H., Lin, S., & Dai, J. (2019). Deformable convnets v2: More deformable, better results. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 9308-9316). https://openaccess.thecvf.com/content_CVPR_2019/html/Zhu_Deformable_ConvNets_V2_More_Deformable_Better_Results_CVPR_2019_paper.html

22. Li, X., Chen, L. C., Hu, X., Zhang, Z., & Yang, J. (2019). Selective kernel networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 510-519). https://openaccess.thecvf.com/content_CVPR_2019/html/Li_Selective_Kernel_Networks_CVPR_2019_paper.html

23. Zhang, Y., Qi, T., Xiao, B., & Wang, J. (2019). Interleaved group convolutions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 6668-6677). https://openaccess.thecvf.com/content_CVPR_2019/html/Zhang_Interleaved_Group_Convolutions_CVPR_2019_paper.html

24. Gao, J., Yang, Z., & Nevatia, R. (2019). Dynamic zoom-in network for fast object detection in large images. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 2704-2713). https://openaccess.thecvf.com/content_CVPR_2019/html/Gao_Dynamic_Zoom-In_Network_for_Fast_Object_Detection_in_Large_Images_CVPR_2019_paper.html

25. Law, H., & Deng, J. (2019). CornerNet-Lite: Efficient Keypoint Based Object Detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (pp. 0-0). https://openaccess.thecvf.com/content_ICCVW_2019/html/LPCV/Law_CornerNet-Lite_Efficient_Keypoint_Based_Object_Detection_ICCVW_2019_paper.html

26. А.Г. Шабанов, А.В. Серебренников, А.А. Черкасов. "Нейросетевые модели для задачи распознавания объектов". Информационные технологии и вычислительные системы, 2019. Ссылка:

http://www.ittc.ru/2019/pdf/13.pdf

27. А.А. Панов, А.В. Кондаков. "Обучение глубоких нейронных сетей для распознавания объектов". Информационные технологии и вычислительные системы, 2018. Ссылка: http://www.ittc.ru/2018/pdf/14.pdf

28. И.В. Кузнецов, С.А. Кузнецов. "Сравнение методов распознавания объектов на основе нейронных сетей". Информационные технологии и вычислительные системы, 2020. Ссылка: http://www.ittc.ru/2020/pdf/12.pdf

29. А.В. Сергеев, А.В. Шпаковский. "Применение сверточных нейронных сетей для распознавания объектов". Материалы XIII Международной конференции "Интеллектуальные системы проектирования", 2019. Ссылка: http://ispranproceedings.elpub.ru/jour/article/view/139

30. И.А. Сергеев, А.В. Селищев. "Применение нейронных сетей для распознавания объектов на изображениях". Материалы Международной научно-технической конференции "Информационные технологии и системы", 2018. Ссылка: http://conf.nsc.ru/IT-2018/ru/paper/1042