Министерство науки и высшего образования Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого
Физико-механический институт
Высшая школа теоретической механики и математической физики

Работа допущена к защите
Директор ВШТМиМФ,
д. ф.-м. н., чл.-корр. РАН
А.М. Кривцов
«___» _____2024 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ
РАЗРАБОТКА СИСТЕМЫ РАСПОЗНАВАНИЯ ВИЗУАЛЬНЫХ ОБРАЗОВ С ПОМОЩЬЮ НЕЙРОСЕТЕВЫХ АЛГОРИТМОВ КАК ЭЛЕМЕНТА КОНТРОЛЯ КАЧЕСТВА НА ПРОИЗВОДСТВЕ

По направлению 01.04.03 «Механика и математическое моделирование»
Направленность 01.04.03_03 «Механика и цифровое производство»

Выполнил
студент гр.5040103/20301 Д.Л. Оленчук

Руководитель
профессор ВШТМиМФ, д.ф.-м.н. В.М. Иванов

Консультант
ассистент ВШТМиМФ Д.С. Перец

Консультант
по нормоконтролю Е.А. Хайбулова

Санкт-Петербург
2024

САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

Физико-механический институт

Высшая школа теоретической механики и математической физики

УТВЕРЖДАЮ

Директор ВШТМиМФ

А. М. Кривцов

«__»_____20__г.

ЗАДАНИЕ

на выполнение выпускной квалификационной работы

студенту Оленчук Дмитрий Леонидович, гр. 5040103/20301

1. Тема работы: Разработка системы распознавания визуальных образов с помощью нейросетевых алгоритмов как элемента контроля качества на производстве.

2. Срок сдачи студентом законченной работы: 30.05.2024.

3. Исходные данные по работе: актуальные научные публикации по теме работы, язык программирования Python, фреймворк для создания и обучения нейронных сетей Tensorflow, сервис разметки данных для компьютерного зрения Superwisely.

4. Содержание работы (перечень подлежащих разработке вопросов): сравнение существующих типов искусственных нейронных сетей, технологий машинного зрения и актуальных реализованных архитектур нейросетевых моделей, разработка структуры нейросети для распознавания предметов в видео-потоке, программирование нейросетевой модели, создание набора данных, обучение нейросетевой модели, оценка качества обучения с помощью метрик, корректировка модели в случае необходимости, тестирование модели на данных, не входивших в обучающий набор.

5. Перечень графического материала (с указанием обязательных чертежей): не предусмотрено.

6. Консультанты по работе: Перец Д.С. – руководитель направления аналитики Auto, Авито (ООО «Авито Тех»).

7. Дата выдачи задания 26.02.2024.

Руководитель ВКР _____ Иванов В.М., профессор ВШТМиМФ, д.ф.-м.н.

Задание принял к исполнению 26.02.2024

Студент _____ Оленчук Д.Л.

РЕФЕРАТ

На 48 с., 34 рисунка, 1 таблица

КЛЮЧЕВЫЕ СЛОВА: КОМПЬЮТЕРНОЕ ЗРЕНИЕ, АНАЛИЗ, РАСПОЗНАВАНИЕ, КАЧЕСТВО, МАШИННОЕ ОБУЧЕНИЕ, НЕЙРОННЫЕ СЕТИ.

Тема выпускной квалификационной работы: «Разработка системы распознавания визуальных образов с помощью нейросетевых алгоритмов как элемента контроля качества на производстве».

В данной работе разработана система распознавания образов в видеопотоке с помощью нейросетевых алгоритмов на основе записей компонентов изделий. Определён алгоритм обучения и оценки работы нейросетевой модели. Результатом работы алгоритма является заключение о соответствии или несоответствии набора компонентов, находящихся в поле зрения камеры, загруженной в память компьютера спецификации изделия. Проведен анализ результатов, полученных в ходе экспериментов.

ABSTRACT

48 pages, 34 figures, 1 table

KEYWORDS: COMPUTER VISION, ANALYSIS, RECOGNITION, QUALITY, MACHINE LEARNING, NEURAL NETWORKS.

The subject of the graduate qualification work is «The Development of a system for recognizing visual objects using neural network algorithms as a quality control element in production

This work presents a system for recognizing patterns in a video stream using neural network algorithms based on recordings of production components. The algorithm for training and evaluating the neural network model has been defined. The result of the algorithm's operation is a conclusion that set of components located within the camera’s view field complies or doesn’t comply with the product specification pre-loaded into the computer memory. An analysis of the results obtained during the experiments has been conducted.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время практически во все сферы жизни современного человечества входят информационные технологии в различных своих проявлениях.

В быту распространены голосовые помощники, домашние роботы, системы «умный дом», беспилотный транспорт, интернет-покупки, виртуальные развлечения.

В классическом образовании широко используются цифровые источники информации, технологии дистанционного обучения. Кроме того, благодаря IT появились новые альтернативные способы получения образования — онлайн-платформы.

В бизнесе активно используются технологии автоматизированного производства, компьютерное моделирование продукции и бизнес-процессов, машинное обучение для анализа больших, в т.ч. неструктурированных объёмов данных.

В военной сфера это связь и координация, беспилотная техника, виртуальные тренажёры и симуляторы.

В сельском хозяйстве осуществляется мониторинг полей с дронов, используется «умный» полив и др.

С одной стороны, в коммерческих сферах такие технологии повышают эффективность деятельности, с другой — они пока очень недёшевы, а потому доступны в основном крупному и среднему бизнесу.

В малом же бизнесе, как правило, нет возможности инвестировать в такие технологии, потому что из-за малого масштаба деятельности они никогда не окупятся. А малый бизнес, между тем, во многих странах является основой экономики.

Естественно, что по законам экономики они (технологии) со временем станут более доступными по стоимости, но некоторые их элементы можно внедрять за небольшую стоимость уже сейчас. И, возможно, благодаря такому

внедрению мелкими шагами, но в больших количествах, дорогие сейчас технологии масштабируются и становятся дешевле.

Одним из «слабых» мест малого бизнеса является отсутствие системного подхода к процессам, слишком широкий функционал сотрудников и как следствие существенное влияние т.н. «человеческого фактора»: люди ошибаются, контроль качества часто отсутствует либо недостаточен в силу того же «человеческого фактора». В итоге — низкая конкурентоспособность и снижение экономической эффективности.

Стабильный уровень качества продукции невозможно обеспечить без его контроля, и именно контроль качества требует абсолютной и постоянной внимательности от контролёра. В этом деле, как показывает практика, более успешны автоматизированные системы.

Разработка такой системы является целью данной работы, а именно программы, позволяющей решить такой вопрос контроля качества продукции, как применение именно тех комплектующих и в том количестве, которые предписаны технологией сборки. Идея программы заключается в том, что комплектующие помещаются в область проверки под видеокамеру, алгоритм распознаёт их, сопоставляет с предзагруженными технологическими данными и сообщает результат (всё верно или что нужно изменить). Таким образом на данном этапе производственного цикла обходится проблема невнимательности контролёра качества.

Целью данной работы является разработка компьютерной программы, собирающей данные с камеры и распознающей какие классы объектов (комплектующих) и в каком количестве присутствуют в поле зрения, для дальнейшего сравнения получившего списка со спецификацией изделия, для которого данные комплектующие были подобраны. Это позволит предотвратить использование неподходящих компонентов в собираемых изделиях.

В ходе исследования были поставлены и выполнены следующие задачи:

1. Изучить методы машинного обучения, позволяющие распознавать физические объекты на изображениях и видео.

2. Изучить данные о существующих в настоящее время решениях в данном направлении.

3. Разработать нейросетевую модель, позволяющую на видеосигнале, поступающем с камеры устройства, выделять предметы и распознавать их.

4. Произвести оценку качества распознавания.

5. Разработать дополнительный модуль программы, в который будут загружаться спецификации изделий, и выбранная спецификация будет сравниваться с полученными данными с камеры, после чего подводится итог о наличии или отсутствии ошибок.

6. Проверить применимость разработанной программы.

ГЛАВА 1. МАШИННОЕ ОБУЧЕНИЕ

1.1. Общая теоретическая информация

Машинное обучение (англ. Machine learning, сокр. ML) — это направление в области технологий искусственного интеллекта, использующее алгоритмы и данные таким образом, чтобы, постепенно повышая точность, приближать процесс обучения и принятия решений компьютерными алгоритмами к тому, как это делает человек.

У алгоритма машинного обучения можно выделить три основные части:

1. Принятие решения. В целом, цель использования машинного обучения - предсказание результата или классификация данных. Для этого алгоритм на основе имеющихся размеченных или неразмеченных данных выявляет закономерности.

2. Функция ошибки. Она оценивает, насколько точно модель машинного обучения предсказывает результат. Сравнивая полученный результат с заранее известным, вычисляется точность работы алгоритма.

3. Оптимизация модели. Если есть возможность приблизить результаты алгоритма к данным в обучающем наборе данных, веса меняются так, чтобы уменьшить расхождения между результатами, полученными моделью, и известными данными. Такой итеративный процесс оценки и оптимизации автономно корректирует веса до достижения порога точности.

Алгоритмы машинного обучения разделяют на 4 основные категории [5]:

- обучение с учителем;
- обучение без учителя;
- обучение с частичным привлечением учителя;
- обучение с подкреплением.

При обучении с учителем используются наборы размеченных данных, на которых модель учится точно предсказывать исходы [8]. По мере обработки

обучающих данных моделью, алгоритм корректирует веса до тех пор, пока результаты модели не достигнут соответствующего уровня, иначе может произойти переобучение или недообучение [12]. Здесь используются такие методы как наивный Байес, случайный лес, нейросети, линейная регрессия, логистическая регрессия и метод опорных векторов. На практике этот вид машинного обучения используется, в частности, для классификации входящей почты как спам.

Обучение без учителя имеет дело с неразмеченными данными (кластерами), его алгоритмы без участия человека самостоятельно выявляют неявные закономерности или классифицируют данные [9]. Самые распространённые методы обучения без учителя:

1. Кластеризация - группирует данные в кластеры, основываясь на их сходстве, например, K-Means, DBSCAN и другие.
2. Снижение размерности - снижение количества признаков в данных с сохранением наиболее важной информации. Это упрощает анализ и снижает вычислительную сложность. Примеры методов снижения размерности: метод главных компонент (PCA), t-распределение стохастического соседства (t-SNE), многомерное масштабирование (MDS).
3. Визуализация данных позволяет представить в двух- или трёхмерном виде характеристики и взаимосвязи (в частности, t-SNE и MDS).
4. Обнаружение аномалий выявляет объекты, существенно отличающиеся от остальных в выборке, что в свою очередь говорит о потенциальных ошибках или нетипичных ситуациях.

Методы: кластеризация на основе выбросов (DBSCAN, LOF) и алгоритмы, основанные на правилах.

Обучение с частичным привлечением учителя являет собой «золотую середину» между обучением с учителем и обучением без учителя. Применяется он в случаях, когда имеется большой набор неразмеченных данных или разметить его стоит неоправданно дорого. Делается частичная выборка из данных, либо размеченных, либо неразмеченных (во втором случае они

размечаются), и обучение модели производится на данных этой частичной выборки [13]. Примеры: методы на основе графов, с использованием самообучения, с использованием генеративных моделей.

Обучение с подкреплением — алгоритм, в котором модель, принимая решение, получает обратную связь в виде «поощрений» и «наказаний» о том насколько этот результат соответствует успешному. Цель в том, чтобы, накапливая «поощрения», научиться выбирать наиболее успешные решения. Примеры: обучение роботов, управление персонажами в компьютерных играх, управление портфелями на финансовых рынках, управление энергосистемами и др. Методы: Q-обучение, обучение с адаптивным поведением (Actor-Critic), стратегии Монте-Карло и др. Данный метод обучения детально описан в статье [7].

1.2. Искусственные нейронные сети

Искусственные нейронные сети — это метод машинного обучения, который называется глубоким обучением. Они представляют собой сети соединенных между собой узлов, которые имитируют работу нейронов в человеческом мозге. Каждый узел, или нейрон, принимает входные данные, выполняет над ними математические операции и передает результат следующему узлу. Вместе эти узлы образуют слои, а набор слоев формирует нейронную сеть [3].

Существует несколько типов искусственных нейронных сетей, каждый из которых предназначен для решения определенного класса задач. Некоторые из наиболее распространенных типов включают в себя:

- перцептрон
- нейронные сети прямого распространения
- свёрточные нейронные сети
- рекуррентные нейронные сети
- глубокие нейронные сети

Перцептрон - один из самых простых типов искусственных нейронных сетей, который играл ключевую роль в развитии области машинного обучения. Это модель искусственного нейрона, предложенная Фрэнком Розенблаттом в 1957 году. Он является простейшим видом многослойной нейронной сети, состоящей из одного или нескольких слоев нейронов, называемых перцептронами. Он обладает способностью обучаться с учителем. Пример перцептрона приведён на рис. 1.1.

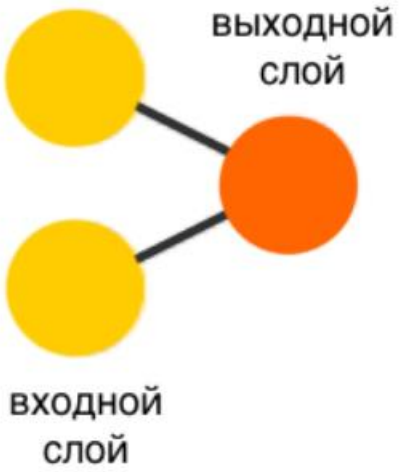


Рисунок 1.1. Графическое представление перцептрона

Перцептрон состоит из входного слоя, скрытых слоёв (при наличии) и выходного слоя [11]. Каждый нейрон входного слоя связан с каждым нейроном следующего слоя. У каждой связи есть свой вес, который определяет вклад входного сигнала в активацию нейрона следующего слоя. Каждый нейрон также имеет своё смещение (bias), который корректирует сумму взвешенных входов.

Обучение перцептрона происходит путём корректировки весов связей и смещений таким образом, чтобы минимизировать функцию потерь. Обычно для этого используется градиентный спуск или его вариации.

Перцептроны широко используются в задачах классификации, распознавания образов и прогнозирования. Они могут быть применены в

различных областях, таких как распознавание рукописных цифр, диагностика болезней, финансовый анализ, анализ текста и многие другие.

Нейронные сети прямого распространения (FFNN) — это класс искусственных нейронных сетей, в которых информационный поток движется только в одном направлении, от входных узлов к выходным без циклических обратных связей. Это означает, что данные проходят через сеть только один раз, без обратного распространения ошибки или рекуррентных связей.

FFNN состоят из одного или нескольких скрытых слоев, входного слоя и выходного слоя. Каждый слой содержит набор нейронов, и каждый нейрон связан с нейронами предыдущего и следующего слоя с помощью весовых коэффициентов. Входные данные подаются на входной слой нейронной сети. Каждый нейрон в следующем слое вычисляет взвешенную сумму входов с использованием весовых коэффициентов и применяет к этой сумме нелинейную активационную функцию. Этот процесс повторяется для всех слоев сети до выходного слоя, который возвращает окончательный результат.

Обучение нейронных сетей прямого распространения обычно осуществляется путем минимизации функции потерь с использованием методов градиентного спуска или его вариаций. Это процесс настройки весовых коэффициентов сети таким образом, чтобы минимизировать разницу между предсказанными и истинными значениями.

Существенным элементом FFNN является использование нейронов с сигмовидной функцией активации. На рис.1.2 представлен нейрон со входами (x_1, x_2, …, x_n), с соответствующими весовыми коэффициентами (w_1, w_2, …, w_n), смещением b и функцией активации f. Выходом является сумма смещения со взвешенной суммой входов. Данные нейронные сети являются основой для компьютерного зрения [18]. На рис. 1.3. представлен пример структуры FFNN.

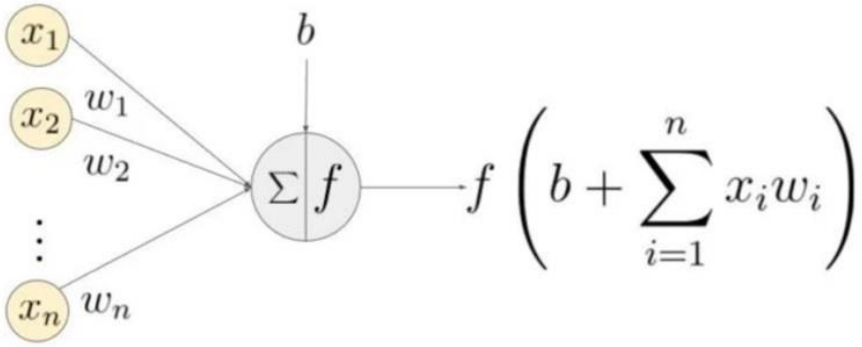


Рисунок 1.2. Пример нейрона

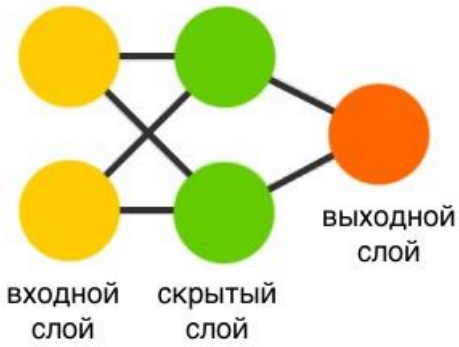


Рисунок 1.3. Графическое представление FFNN

Свёрточные нейронные сети (CNN) — это класс глубоких нейронных сетей, специализирующейся на обработке и анализе изображений. Они состоят из нескольких слоев, включая свёрточные, пулинговые и полносвязные слои.

Свёрточные слои используют операцию свёртки для извлечения признаков из входного изображения. Каждый нейрон в свёрточном слое соединён с небольшой областью входного изображения, называемой receptive field, и применяет к ней фильтр (ядро) для выделения различных признаков, таких как границы, углы и текстуры [21].

Пулинговые слои уменьшают размерность изображения, сохраняя наиболее важные признаки. Это позволяет уменьшить количество параметров и улучшить инвариантность к масштабированию и трансляции.

Полносвязные слои представляют собой обычные нейронные слои, в которых каждый нейрон связан с каждым нейроном предыдущего слоя. Входные данные, поступающие в этот слой, умножаются на матрицу весов, после чего добавляется вектор смещения [22]. Пример свёрточной нейронной сети приведён на рис. 1.4.

Свёрточные нейронные сети широко используются в различных областях:

- обнаружение и классификация объектов на изображениях и видео
- в медицинской диагностике используются для автоматической интерпретации изображений с целью выявления заболеваний и патологий
- распознавание рукописного текста и др. [2].

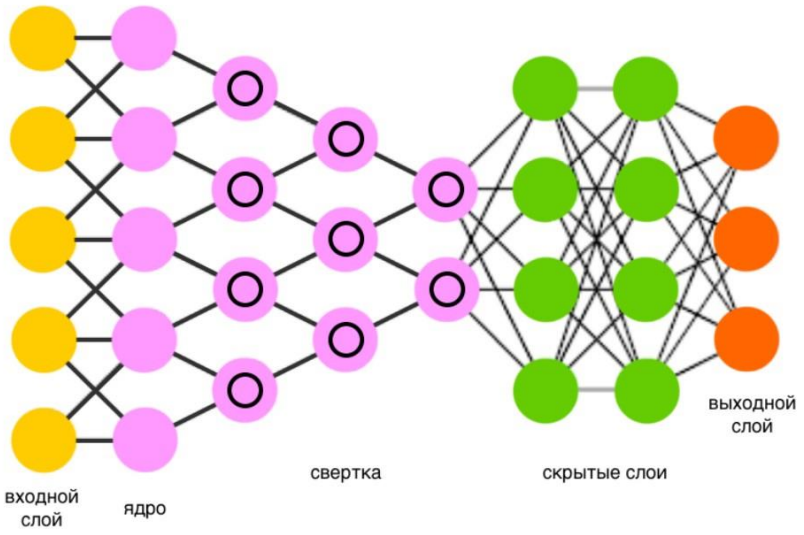


Рисунок 1.4. Графическое представление CNN

Рекуррентные нейронные сети (RNN) — это класс нейронных сетей, способных работать с последовательными данными, сохраняя внутреннее состояние для обработки последующих входных данных. Эти сети обладают способностью анализировать данные, учитывая их последовательную природу, что делает их мощным инструментом в области обработки текста, временных рядов, аудио- и видеоанализа, а также других областей, где важна учетная последовательность данных.

Основной принцип работы рекуррентной нейронной сети заключается в том, что она обрабатывает входные данные поэлементно, сохраняя состояние между последовательными входами. Это достигается за счет обратной связи, когда выход нейронной сети на предыдущем временном шаге используется как вход на следующем временном шаге [17].

Структура RNN включает в себя один или несколько слоев рекуррентных нейронов. Входные данные подаются на каждом временном шаге, а выходы могут быть получены после обработки всей последовательности или после каждого временного шага. Пример RNN можно увидеть на рис. 1.5.

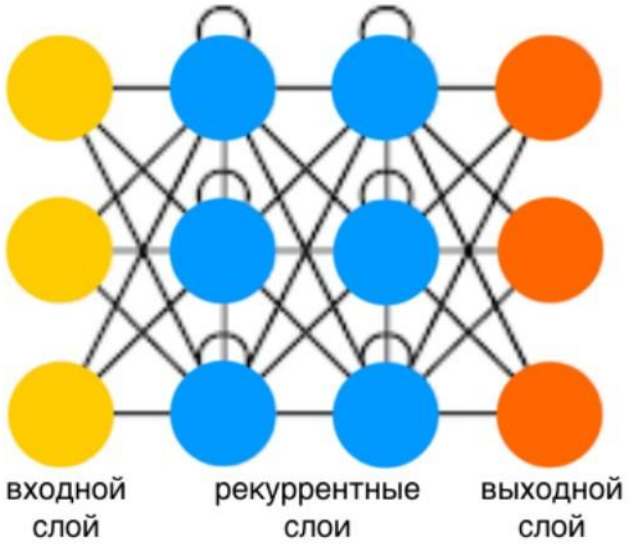


Рисунок 1.5. Графическое представление RNN

RNN нашли широкое применение в следующих областях:

- анализ текста, машинный перевод, генерация текста и другие задачи, связанные с естественным языком.
- прогнозирование временных рядов, анализ финансовых данных, управление временными рядами и т.д.
- распознавание речи, аудиоанализ, синтез речи и т.д.
- анализ движения, распознавание жестов, обработка видео и другие задачи, связанные с анализом видеоданных.

ГЛАВА 2. МАШИННОЕ ЗРЕНИЕ

2.1. Определение и задачи

Машинное зрение (или компьютерное зрение) — это область искусственного интеллекта и информатики, занимающаяся разработкой алгоритмов и систем для автоматического анализа и интерпретации визуальных данных [16]. Основной целью машинного зрения является создание систем, которые могут понимать и интерпретировать информацию, содержащуюся в изображениях или видеопотоках, аналогично тому, как это делает человеческое зрение.

Основные задачи машинного зрения:

- Обнаружение объектов: определение наличия объектов в изображении или видеопотоке и их классификация по типам.
- Распознавание объектов: идентификация конкретных объектов или категорий объектов на изображениях.
- Локализация объектов: определение местоположения объектов в изображении с использованием ограничительных рамок или других методов.
- Сегментация изображений: разделение изображения на несколько сегментов или областей, часто для выделения и анализа объектов и структур.
- Отслеживание объектов: наблюдение и отслеживание движения объектов в последовательности изображений (видеопоток).
- Распознавание лиц: идентификация и верификация лиц в изображениях или видеопотоках.
- Оптическое распознавание символов (OCR): преобразование изображений текста в машинно-читабельный текст.

2.2. Обзор методов идентификации объектов

Задача по распознаванию объекта на входном сигнале обычно делится на два этапа, прежде чем можно получить итоговый результат. Первый этап состоит в классификации, где программа получает на вход изображение или поток изображений (видео) с искомым объектом и предсказывает тип или класс данного объекта. На выходе получается метка класса, которому принадлежит объект. Вторым этапом является локализация, где на вход поступают те же данные, но программа определяет наличие на входном изображении или видеопотоке объектов конкретного класса и указывает их местоположение ограничительной рамкой. На выходе получаются данные того же типа, что и на входе, но объекты искомых классов выделяются ограничительными рамками.

Таким образом, распознавание объектов – это одновременно и задача классификации, и локализации экземпляров объекта, будь то изображение или видеопоток [14].

Для решения данной задачи разработаны различные методы, в том числе R-CNN, Fast R-CNN, Faster R-CNN, YOLO, U-Net и другие.

R-CNN (Region-based Convolutional Neural Networks) — это метод обнаружения объектов в изображениях, который был представлен в 2014 году Россом Гиршиком, Соломоном Шингом и Шенноном Фэрлингхемом [10]. R-CNN был первым методом, который успешно применил глубокие нейронные сети к задаче обнаружения объектов. Этот метод стал важным прорывом в области компьютерного зрения и затем был дальше улучшен и развит.

Исторически R-CNN был разработан в ответ на проблему того, что традиционные методы обнаружения объектов, такие как методы на основе признаков Хогга и Haar, не показывали высокую точность на сложных датасетах, таких как PASCAL VOC и ImageNet. Другие методы, такие как Deformable Parts Models (DPM), имели ограничения в эффективности и точности.

R-CNN предложил использовать сверточные нейронные сети для извлечения признаков из изображений, а затем применять классификаторы и

регрессоры для обнаружения и локализации объектов. Он состоит из нескольких основных этапов:

Предварительная обработка изображения: Входное изображение подается на вход свртоочной нейронной сети (CNN), чтобы извлечь признаки. CNN принимает входное изображение I и преобразует его в признаковое представление $f(I)$.

Генерация пропозалов (Region Proposals): Алгоритм генерации пропозалов создает прямоугольные области, которые могут содержать объекты [20]. Эти области называются "пропозалами". Алгоритм генерации пропозалов возвращает список $R=\{r_1,r_2,...,r_n\}$, где каждый r_i представляет собой ограничивающую рамку (bounding box) в форме координат (x,y,w,h), которая предположительно содержит объект.

Извлечение признаков: Каждая ограничивающая рамка r_i пропускается через CNN для извлечения признаков: $f(r_i)$.

Классификация: Извлеченные признаки $f(r_i)$ подаются на вход классификатору, который выдает вероятность принадлежности к каждому классу: $p(c|r_i)$, где c - класс, r_i - ограничивающая рамка.

Регрессия: регрессор определяет класс и точное положение каждого объекта: (x',y',w',h') = regressor($f(r_i)$), где (x,y,w,h) - изначальные координаты и размеры, а (x',y',w',h') - скорректированные значения.

Плюсы: R-CNN точно локализует объекты на изображении и демонстрирует хорошую точность.

Минусы: Она медленная в работе из-за необходимости обработки большого числа регионов на изображении, а также требует значительного объема памяти и вычислительных ресурсов при обучении.

Fast R-CNN (Fast Region-based Convolutional Network) является алгоритмом глубокого обучения, разработанным для задачи обнаружения объектов в изображениях. Он представляет собой улучшение предыдущей модели, R-CNN (Region-based Convolutional Network), и предлагает значительные улучшения в скорости и эффективности обучения.

Основные особенности Fast R-CNN включают в себя:

1. Использование всего изображения для обучения свёрточной нейронной сети (СНС): В отличие от R-CNN, где для каждого предложенного региона создается отдельный набор признаков, Fast R-CNN использует всё изображение для получения карты признаков с помощью СНС. Это позволяет избежать повторного вычисления признаков для перекрывающихся регионов и существенно ускоряет процесс.

2. RoI (Region of Interest) Pooling слой: после получения карты признаков для всего изображения, Fast R-CNN использует RoI Pooling слой для извлечения фиксированного размера признаков из предложенных регионов интереса (RoI). Это позволяет сети работать с регионами различных размеров и масштабов, приводя их к единому виду.

3. Мультизадачное обучение: Fast R-CNN выполняет одновременно две задачи - классификацию регионов и коррекцию их границ (регрессия ограничивающих рамок). Это достигается благодаря специализированным полносвязным слоям, которые используют извлеченные признаки для выполнения обеих задач.

4. Улучшенный алгоритм обучения: Fast R-CNN использует метод обучения, основанный на мини-пакетах (mini-batch training), где каждый мини-пакет содержит множество RoI из разных изображений. Это позволяет более эффективно использовать вычислительные ресурсы и ускоряет процесс обучения.

Fast R-CNN использует сверточные нейронные сети для извлечения признаков изображения и методы региональной предобработки для обнаружения объектов. Вот основные шаги алгоритма Fast R-CNN:

1. Извлечение признаков: изображение подается на сверточную нейронную сеть, такую как VGG, ResNet и т. д., для извлечения признаков. Это позволяет модели автоматически извлекать важные черты изображения.

2. Получение областей предложений (RoIs): используется алгоритм Selective Search для генерации областей, предполагаемых наличием объектов.

3. Подготовка RoIs: Каждая область предложения проходит через процесс ROI Pooling (Region of Interest Pooling), который преобразует разные размеры RoIs в фиксированный размер с учетом извлеченных признаков.

4. Классификация и регрессия: Обработанные RoIs подаются на полносвязный слой, который выполняет классификацию объектов и регрессию, чтобы уточнить координаты ограничивающего прямоугольника (bounding box) для каждого объекта.

5. Обучение и оптимизация: Модель обучается на размеченных данных, используя функцию потерь, такую как многоклассовая логистическая потеря для классификации и потеря регрессии для координат bounding box.

Плюсы: Fast R-CNN более быстрая и эффективная по сравнению с R-CNN.

Минусы: все еще медленнее по сравнению с более поздними моделями.

Faster R-CNN (Faster Region-Based Convolutional Neural Network) — это модификация архитектуры R-CNN, которая была разработана для более быстрого и эффективного обнаружения объектов на изображениях. Она впервые была представлена в статье "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks" авторства Shaoqing Ren, Kaiming He, Ross Girshick и Jian Sun в 2015 году.

Faster R-CNN была представлена как развитие предыдущих методов обнаружения объектов, таких как R-CNN и Fast R-CNN. Основным автором архитектуры Faster R-CNN является Kaiming He, который известен своими работами в области компьютерного зрения и глубокого обучения.

Основная идея Faster R-CNN заключается в том, чтобы полностью интегрировать процесс обнаружения объектов и обнаружения областей (Region Proposal) в одну модель.

Принцип работы:

Faster R-CNN состоит из двух основных компонентов: сверточной нейронной сети (CNN) для извлечения признаков и модуля Region Proposal Network (RPN) для создания областей-кандидатов [15].

1. Сначала на вход подается изображение, которое проходит через свёрточные слои CNN и выделяет признаки объектов на изображении.

2. Затем модуль RPN генерирует области-кандидаты, которые вероятно содержат объекты. Эти области-кандидаты создаются с использованием якорных прямоугольников на разных масштабах и соотношениях сторон.

3. Области-кандидаты затем проходят через процесс классификации и точной регрессии, чтобы определить тип объекта и уточнить его координаты на изображении.

4. Наконец, на выходе получается список найденных объектов с их классами и координатами.

Плюсы:

- Высокая точность обнаружения объектов.
- Скорость работы позволяет применять модель в реальном времени.
- Полная интеграция процесса обнаружения объектов и обнаружения областей-кандидатов делает архитектуру более эффективной и легко масштабируемой.

Минусы: Она требует больше вычислительных ресурсов для обучения из-за наличия двух нейронных сетей: для генерации регионов и для классификации.

YOLO (You Only Look Once) — это метод глубокого обучения для обнаружения объектов на изображениях. Он был разработан Джозефом Редмондом, Сантьяго де Ла Сигуэра, Сэндиасом Кириллом и Райаном Фарреллом и впервые представлен в статье "You Only Look Once: Unified, Real-Time Object Detection" в 2016 году.

История создания YOLO связана с поиском эффективного и быстрого метода для обнаружения объектов на изображениях. Ранее методы обнаружения объектов, такие как R-CNN и его вариации, требовали выполнения нескольких шагов, включая генерацию областей-кандидатов и их классификацию. Это было не только трудоемким, но и неэффективным с точки зрения скорости обработки изображений в реальном времени.

YOLO был создан для того, чтобы обеспечить единый и быстрый процесс обнаружения объектов. Основная идея YOLO заключается в том, чтобы применять глубокую нейронную сеть к изображению в едином проходе, предсказывая сразу и координаты объектов, и их вероятности принадлежности к различным классам.

Принцип работы YOLO состоит из нескольких шагов:

1. Входное изображение: Изображение подается на вход нейронной сети.
2. Прямой проход (Forward Pass): Изображение проходит через предварительно обученную глубокую нейронную сеть, которая состоит из сверточных слоев, слоев объединения (pooling), и полносвязных слоев.
3. Разделение изображения на сетку (Grid Cell Division): Изображение разделяется на фиксированное количество ячеек (grid cells), которые покрывают всю область изображения.
4. Предсказание боксов (Bounding Box Prediction): для каждой ячейки сетки модель делает прогнозы о наличии объекта и определяет ограничивающие рамки (bounding boxes) для обнаруженных объектов. Каждая ограничивающая рамка характеризуется координатами прямоугольника и уверенностью (confidence score) в том, что внутри рамки находится объект.
5. Классификация объектов (Object Classification): кроме того, для каждой ограничивающей рамки модель делает предсказание о классе объекта, который она содержит.
6. Подавление немаксимумов (Non-maximum Suppression): для уменьшения количества ложных срабатываний и повышения точности модель использует алгоритм подавления немаксимумов. Этот алгоритм удаляет лишние ограничивающие рамки, оставляя только самые уверенные прогнозы для каждого объекта.

Математическая модель YOLO включает в себя ряд функций активации, сверточных операций, функций потерь и других операций, которые применяются к изображению и помогают сети сделать прогнозы о наличии и классификации объектов.

U-Net — это архитектура нейронной сети, разработанная для сегментации изображений, то есть для выделения объектов на изображениях путем разделения их на пиксели и классификации каждого пикселя как принадлежащего к объекту или фону. U-Net была впервые представлена в статье "U-Net: Convolutional Networks for Biomedical Image Segmentation", опубликованной в 2015 году Ольфом Ронненбергером и др.

U-Net была разработана для решения задач сегментации биомедицинских изображений, таких как снимки МРТ и сканирование тканей. Однако ее архитектура стала широко применяться и в других областях, таких как обработка изображений, сегментация снимков спутников и других видов изображений.

U-Net имеет особенную архитектуру, которая включает в себя сверточные слои (encoder) для извлечения признаков и слои расширения (decoder) для восстановления пространственной информации. Его основной принцип заключается в том, чтобы обеспечить точную сегментацию объектов, сохраняя при этом пространственную информацию.

Архитектура U-Net состоит из следующих ключевых компонентов:

1. Сверточные блоки (Encoder): эти блоки состоят из нескольких сверточных слоев, которые извлекают признаки из входного изображения и уменьшают его размерность.

2. Пулинг (Pooling): В процессе кодирования размер изображения уменьшается с помощью операции пулинга (например, максимальное пулинга), что помогает увеличить эффективность и скорость обработки.

3. Связи пропуска (Skip Connections): эти связи позволяют передавать информацию о низкоуровневых признаках с энкодера к декодеру, что помогает восстановить пространственную информацию в процессе декодирования.

4. Сверточные блоки с увеличением (Decoder): эти блоки увеличивают размерность изображения и восстанавливают пространственную информацию с использованием сверточных операций и операций объединения (upsampling).

5. Конкатенация (Concatenation): Результаты декодера объединяются с результатами связей пропуска для более точного восстановления пространственной информации.

Математическая модель U-Net включает в себя операции свертки, пулинга и объединения, а также функции активации (например, ReLU), функции потерь (например, кросс-энтропия) и другие компоненты, которые помогают сети извлекать признаки из изображений и делать точные прогнозы о принадлежности каждого пикселя к объекту.

U-Net широко используется в медицинском и научном образовании для сегментации биомедицинских изображений, таких как снимки МРТ и сканирование тканей. Однако его архитектура также нашла применение в других областях, таких как анализ изображений, обработка изображений и компьютерное зрение.

2.3. Выбор метода идентификации объектов в видеопотоке

Проанализировав все рассмотренные выше методы и учитывая поставленную задачу одновременной идентификации нескольких предметов, как одинаковых, так и разных, в видеопотоке, был сделан вывод о целесообразности выбора архитектуры U-Net по следующим причинам:

1. Архитектура сегментации: U-Net была специально разработана для задач сегментации изображений, что делает ее эффективной в идентификации и выделении объектов на изображениях и в видеопотоках. Ее архитектура позволяет точно определять границы объектов и выделять их на изображении с высокой точностью.

2. Скорость и эффективность: в отличие от архитектур R-CNN, Fast R-CNN и Faster R-CNN, которые требуют сложной последовательной обработки каждого объекта, U-Net может одновременно обрабатывать все объекты на изображении без необходимости повторного сканирования. Это делает U-Net

более эффективной в сегментации видеопотоков с большим количеством объектов.

3. Обработка изображений разного размера и формы: U-Net способна обрабатывать изображения разного размера и формы благодаря своей архитектуре, которая включает в себя связи пропуска. Это позволяет ей эффективно работать с разнообразными объектами на изображении, включая объекты разного размера и формы.

4. Простота и универсальность: U-Net имеет относительно простую архитектуру и легко настраивается для различных типов данных и задач. Она также может быть использована для различных типов сегментации, не только для видеопотоков, но и для статических изображений, медицинских изображений и других типов данных.

5. Точность и качество: U-Net известна своей высокой точностью и качеством сегментации, что делает ее предпочтительным выбором для задач, требующих точного определения объектов на изображении или в видеопотоке.

Ниже приведена таблица сравнения важнейших характеристик для выбора архитектуры.

Таблица 2.1 Сравнительные характеристики архитектур по 5-балльной шкале

Архитектура	Вычисли-тельная эффектив-ность	Подходит для задач сегмен-тации	Простота настройки	Точность распознава-ния объектов сложных форм	Сум-ма баллов
R-CNN	1	2	2	3	8
Fast R-CNN	3	3	3	3	12
Faster R-CNN	4	3	3	4	14
YOLO	5	2	4	3	14
U-Net	3	5	5	5	18

ГЛАВА 3. РЕАЛИЗАЦИЯ МЕТОДА

Решение реализовано на языке программирования Python с использованием фреймворка TensorFlow.

В целом, задачу распознавания объектов на изображениях можно разделить по группам: классификация (определение, к какому из классов относится объект), локализация (определение местоположения объекта на изображении и нахождение его координаты) и сегментация (определение того, какая часть изображения относится к одному или другому классу). В рамках данной работы была осуществлена мультиклассовая сегментация, то есть определено и выделено несколько различных объектов на одном изображении, при этом в данной архитектуре классификация и локализация не реализуются как отдельные процессы, т.к. решаются как неотъемлемая часть мультиклассовой сегментации.

3.1. Архитектура U-Net

На рис. 3.1. схематично представлена структура U-Net.

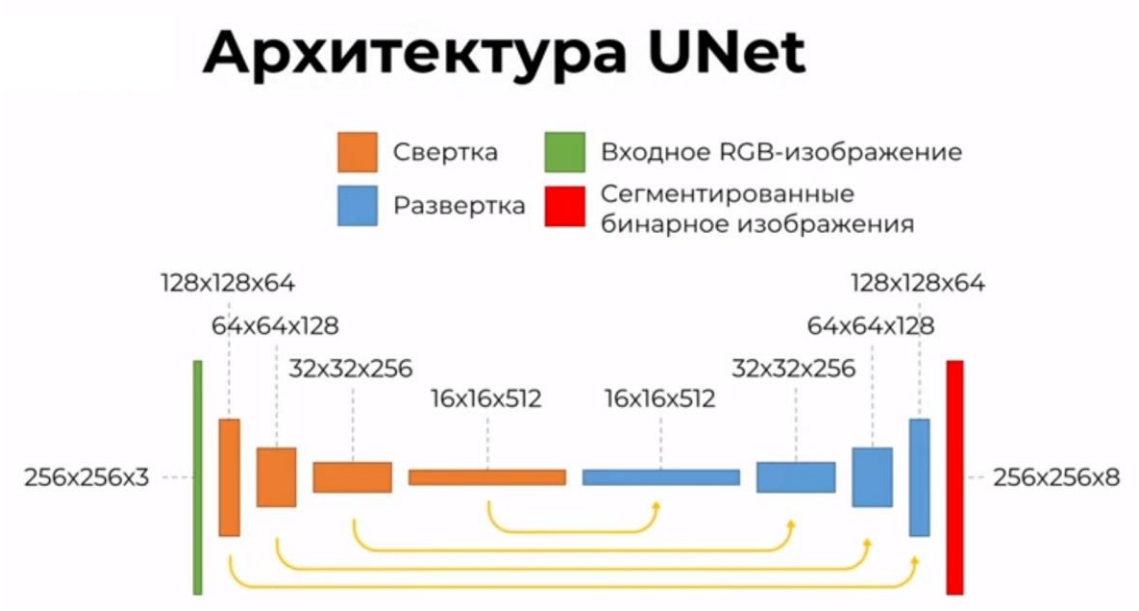


Рисунок 3.1. Графическое представление архитектуры U-Net

Как было упомянуто выше, используется архитектура U-Net, которая предполагает выполнение следующих шагов:

- формирование набора данных для обучения нейросетевой модели,
- построение модели нейронной сети,
- обучение нейросети на подготовленном наборе данных,
- тестирование обученной нейросети.

3.2. Формирование набора данных и разметка

Для формирования набора данных произведена видеосъёмка нескольких комплектующих, использующихся в производстве, с последующим извлечением из него изображений в формате jpg. Фон выбран однотонным, отличающимся цветом от всех предметов. Съёмка проводилась под разными углами с изменением расстояние от камеры до объектов сцены. Это повышает вариативность изображений, извлечённых в дальнейшем из видео. На видео присутствовали объекты различных цветов, форм и размеров (см. рис.3.2).

Рисунок 3.2. Состав комплектующих для обучающего набора данных

Длительность видео составила 38 секунд. С помощью утилиты FFmpeg из него было извлечено 1338 кадров и сохранено в качестве простых изображений. Так как разметка данных производилась самостоятельно вручную, то было подготовлено небольшое количество изображений (42 шт.).

Для разметки выбранных изображений выбран облачный сервис SUPERWISELY, так как он предоставляет инструмент SmartTool, с помощью которого можно провести разметку качественно и быстро (см. рис. 3.3).

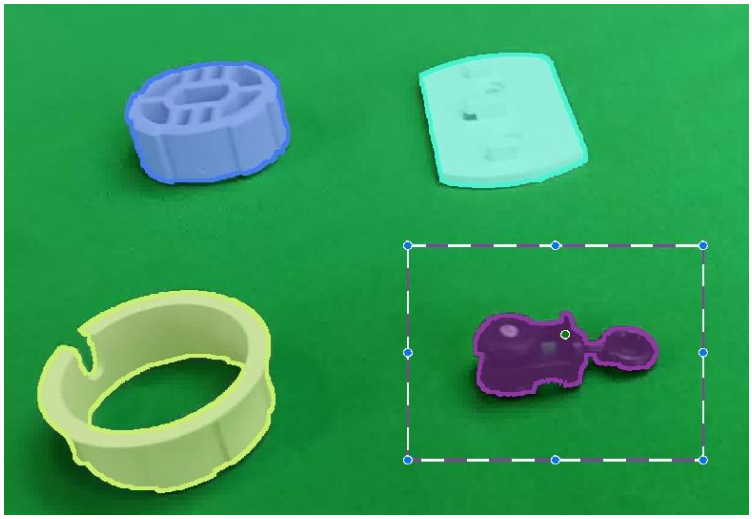


Рисунок 3.3. Разметка данных в сервисе Superwisely

Данный сервис в качестве результата разметки (аннотации) изображений позволяет получить монохромные изображения предметов, в которых пиксели объектов каждого класса имеют различные значения. Так как на сцене представлено всего ЧЕТЫРЕ объекта, то и диапазон значений находится в интервале от 1 до 4. Поэтому, если открыть эти монохромные изображения, то размеченные на них объекты глазом различить практически невозможно. Но если программно открыть эти картинки, то можно убедиться, что все данные на местах, это видно на рис. 3.4.

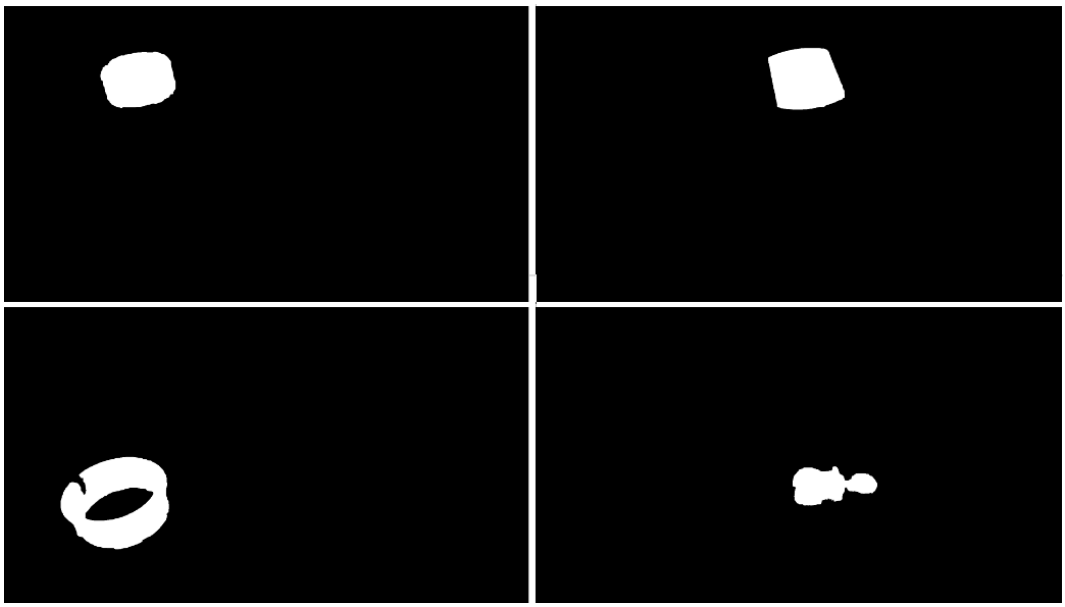


Рисунок 3.4. Маски размеченных объектов в формате «png»

3.3. Формирование нейросетевой модели

Следующим этапом настраивается входной конвейер данных с помощью фреймворка TensorFlow. В нём доступен инструмент, с помощью которого можно организовать сложные входные потоки данных. Например, есть такие методы, как последовательная или параллельная обработка, кэширование, предвыборка, векторизация данных и другие. Набор этих инструментов позволяет создать оптимизированный входной конвейер данных. Таким образом, можно балансировать между потреблением ресурсов процессора, оперативной памяти или хранилища.

В данной работе загружен небольшой объем данных с диска и их аугментация, т.е. искусственное увеличение набора данных, проводилась на лету. Объем свободной оперативной памяти позволяет разместить все промежуточные данные в ней. Таким образом, взаимодействие с диском, самым медленным звеном, сведено к минимуму.

Далее был сформирован входной конвейер данных TensorFlow по схеме, изображённой на рис. 3.5.

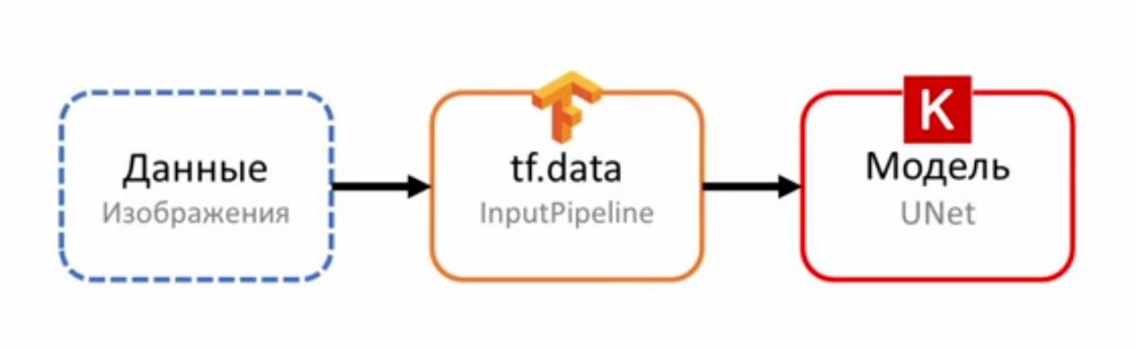


Рисунок 3.5. Схема входного конвейера данных

В самом начале объявлены переменные, задающие число классов, т.е. всех объектов, присутствующих на сцене, и плюс один класс для обозначения заднего фона. Заданы различные цвета, с помощью которых будут подсвечиваться объекты на изображениях. Размер входного изображения, которое будет использоваться в нейронной сети, задан как 256x256, и размер выходного изображения, которое будет формироваться после финальной обработки данных - 1920x1080 (96 dpi). Математически этот процесс представляет собой следующее.

Пусть X — это входное изображение размером $H \times W \times C$, где H и W - высота и ширина изображения соответственно, а C - количество каналов.

Пусть F — это ядро (фильтр) свертки размером $h_f \times w_f \times C \times K$, где h_f и w_f - высота и ширина ядра, C - количество каналов во входном изображении, K - количество фильтров.

Тогда операция транспонированной свертки принимает вход X и выполняет следующие действия:

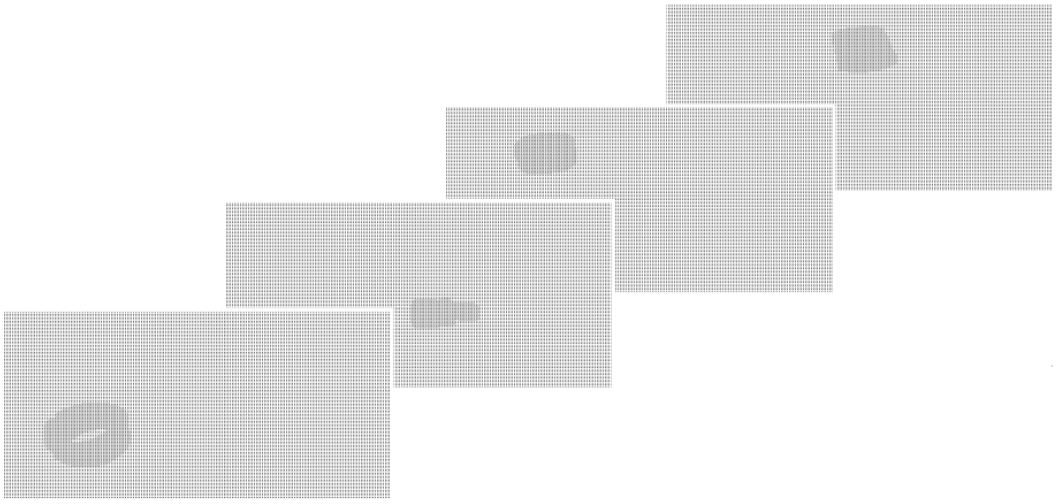
- Применяется свёртка между входом X и ядром F с использованием тех же методов паддинга (добавление в данном случае нулевых элементов вокруг существующих данных) и шага, что и в обычной свертке.
- Результат этой операции - выходное изображение большего размера.

После этого определяются две функции.

Первая, «load_image», принимает на вход путь к изображению. С помощью нескольких специальных функций TensorFlow файл загружается и

преобразуется в необходимый формат, изменяется размер, нормализуются значения. Аналогичным образом с помощью этой же функции загружаются маски. Изображения с масками представляют собой одномерные массивы, в которых регионы с объектами соответствующих классов помечены различными значениями от 1 до n. Маска представляет собой бинарное изображение, которое указывает на области изображения, соответствующие конкретному объекту или классу объектов. Обычно маска состоит из пикселей, где каждый пиксель имеет значение либо 0, либо 1, где 1 обозначает принадлежность объекту, а 0 — фону.

Представления различных классов объектов разделены по нескольким каналам. Таким образом, вместо одномерного массива с маской получится многомерный массив, глубина которого соответствует количеству классов объектов, и в каждом слое этого массива содержатся только нули или единицы, показывающие отсутствие или наличие объекта на изображении (см. рис. 3.6). В результате формируются бинарные многоканальные изображения, размерность которых равна количеству классов объектов.



3.6. Многомерный массив

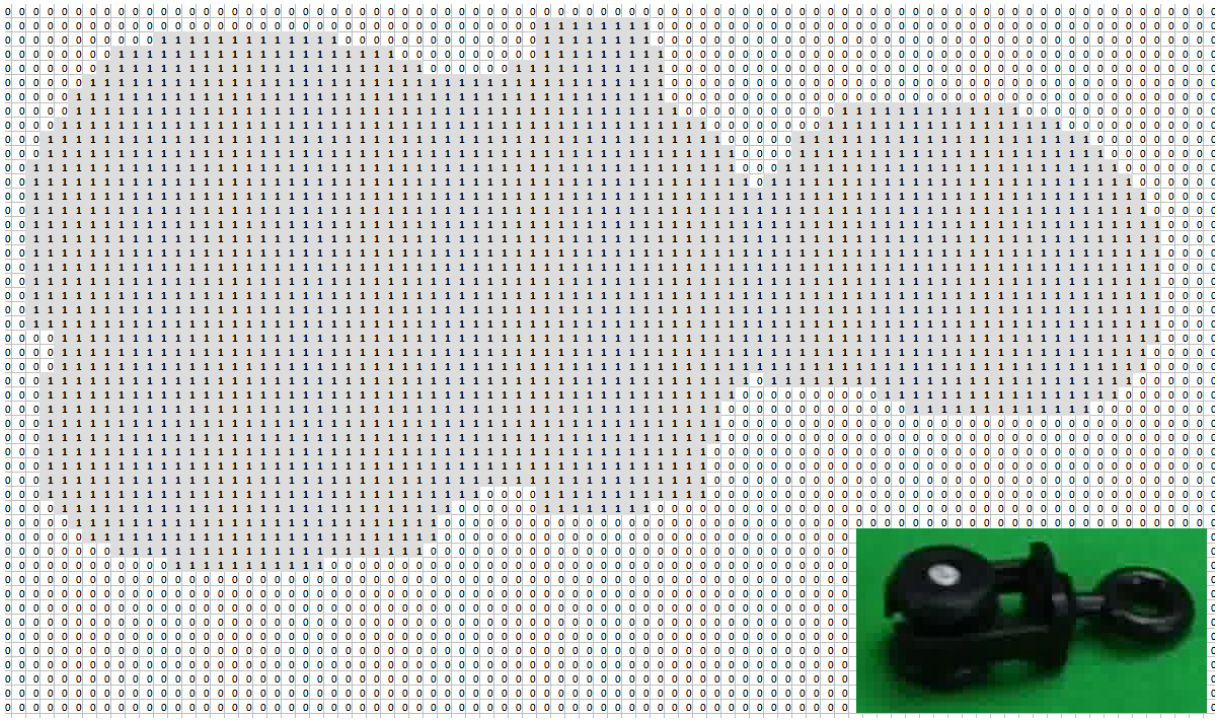


Рисунок 3.7. Один из слоёв многомерного массива и фото соответствующего предмета

Следующая функция называется «augmentate_images», она принимает на вход трехканальное RGB изображение и многоканальную маску, сформированную на предыдущем этапе. Она проводит с ними различные преобразования, которые представляют собой аугментацию данных или искусственное увеличение набора данных [19]. Эти преобразования способствуют увеличению вариативности выборки, и это в конечном итоге положительно сказывается на качестве работы нейросетевой модели. Для этого применяется функция «central_crop», которая извлекает центральный фрагмент со случайным значением масштаба и применяется случайное отражение изображения по горизонтали. И в конце этого конвейера устанавливается выходной размер изображений и масок.

После того, как функции обработки определены, загружаются данные с диска. Важно, чтобы изображению соответствовала нужная маска.

Далее формируются наборы данных из изображений и масок, они объединяются для параллельной обработки. С помощью функции «load_images» данные загружаются в память. с помощью функции «repeat» объем данных увеличивается в 50 раз простым копированием, и ко всему объему данных применяется функция аугментации. Теперь каждое изображение в этом наборе уникально. Благодаря такому методу небольшой объем данных, размеченных вручную, искусственно увеличился в 50 раз.

Затем с помощью функций «take» и «skip» этот набор данных делится на обучающий и тестовый, при этом они кэшируются в памяти. Устанавливается размер пакета, которым обучается нейроннаю сеть.

На этом подготовка входного конвейера данных TensorFlow завершена.

Существует несколько способов обработки изображений нейронными сетями. Например, если необходимо обработать большое изображение в высоком разрешении, то оно разделяется на несколько более мелких изображений и обрабатывается каждое из них. При этом увеличивается время обработки.

В том случае, если нужно обработать относительно небольшое изображение, и не важна детализация полученного результата, то можно уменьшить изображение настолько, чтобы целиком его пропустить через нейронную сеть.

Последний подход и был применён. Оригинальные изображения в разрешении Full HD масштабируются до установленного размера 256x256 пикселей. Это реализуется с использованием бикубической интерполяции:

$$f(x) = a \cdot x^3 + b \cdot x^2 + c \cdot x + d, \quad (3.1)$$

где a, b, c, и d - коэффициенты, которые определяются на основе значений пикселей в окружающих областях.

После преобразования в размер 256x256 они подаются на вход нейронной сети, а на выходе получаются многоканальные сегментированные изображения. Снижение детализации необходимо для упрощения и ускорения процесса.

Нейросетевая модель состоит из сужающейся первой части и расширяющейся второй. Они называются энкодер-декодер (см. рис. 3.8).

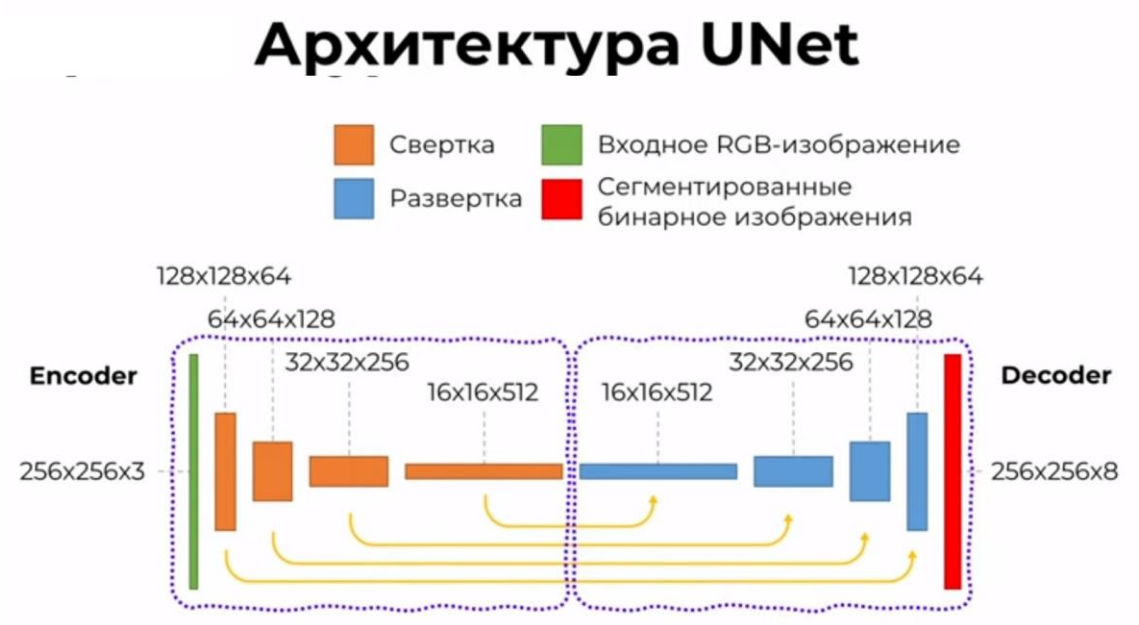


Рисунок 3.8. Графическое представление энкодера-декодера

Первая часть — это свёрточная сеть, состоящая из комбинации свёрточных слоев, слоев пакетной нормализации и активационной функции LeakyReLU (см. рис. 3.9).

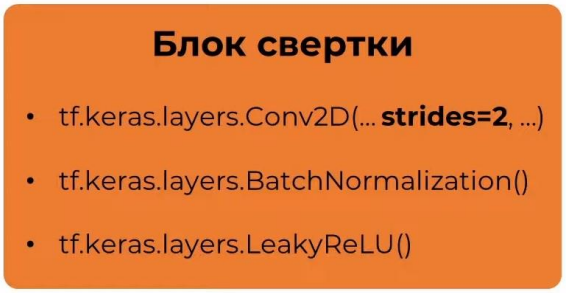


Рисунок 3.9. Блок свёртки нейронной сети

Вместо pooling-слоёв для уменьшения размерности, как в оригинальной архитектуре, используется параметр «strides» в самом свёрточном слое.

Вторая часть нейронной сети состоит из комбинации Transpose свёрточных слоев, слоев пакетной нормализации, DropOut слоев и активационной функции ReLU (см. рис.3.10).

Рисунок 3.10. Блок развёртки нейронной сети

DropOut слои являются инструментом для предотвращения переобучения сети, а также в целом положительно сказывается на процессе её обучения. Слои Batch (слои пакетной нормализации) применяются для того, чтобы повысить производительности сети и сделать её работу стабильнее. Свёрточные Transpose-слои применяются для обратной операции свёртки и для увеличения размерности выходных данных.

Особенностью свёрточной архитектуры U-Net является применение подхода «Skip connections», изображённого на рис.3.11.

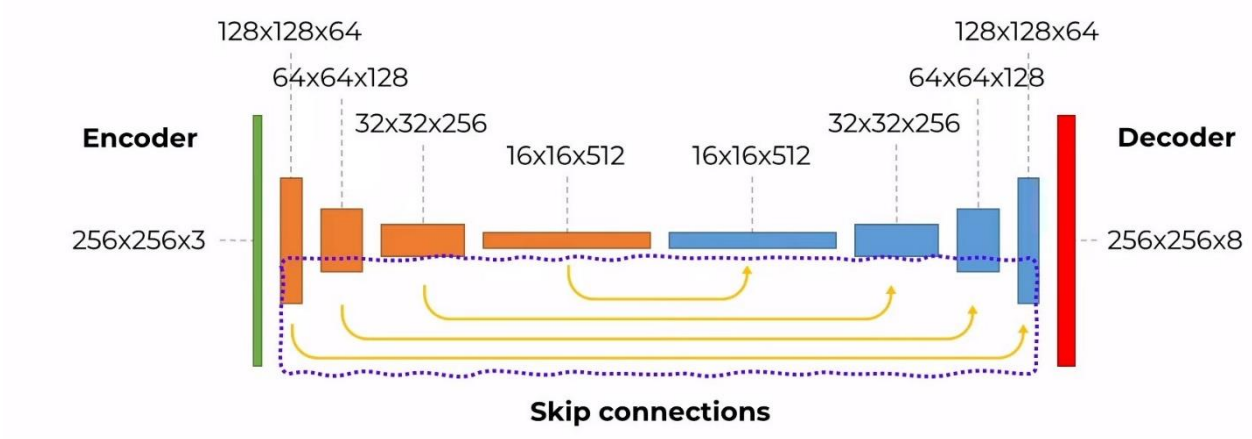


Рисунок 3.11. Графическое представление «Skip connections»

Сейчас это стандартный инструмент во многих свёрточных архитектурах. Эта техника часто применяется в глубоких архитектурах и положительно сказывается на сходимости модели. Благодаря этим межслоевым соединениям, информация низкого уровня совместно используется входными и выходными слоями, а в архитектуре U-Net эти соединения используются для передачи признаков, полученных в энкодере, в декодер. Это помогает восстановить пространственную информацию, утерянную в результате операции свёртки.

В целом, «Skip connections» позволяют использовать ранее извлечённые признаки в декодере и стабилизируют процесс обучения и сходимости нейронной сети.

Для дальнейшей реализации нейросетевой модели на Python используется библиотека Keras, являющейся частью фреймворка TensorFlow. Она содержит все необходимые для этого модули. Для программирования сети используются следующие функции.

Первая функция «input_layer» задает входной слой нейронной сети и устанавливает размер входных данных.

Функция «downsample_block» описывает блоки, которые формируют энкодер. Она задает метод инициализации весовых коэффициентов, включает свёрточный слой, добавляет слои пакетной нормализации и устанавливает активационную функцию.

Функция «upsample_block» помогает формировать декодер нейронной сети, но вместо функции свертки используется противоположная ей функция Transpose. Также в этом блоке предусмотрена возможность добавления DropOut слоев.

Последняя функция «output_layer» задает выходной слой размерностью, соответствующей количеству классов объектов на изображении, и использует сигмоидную активационную функцию [4].

Затем с помощью этих функций создаётся массив «downsample_stack», представляющий энкодер, и «upsample_stack», представляющий декодер. Такая структура данных позволяет достаточно просто реализовать «skip connections».

С помощью этих двух циклов соединяются блоки энкодера и декодера друг с другом, и с помощью операции конкатенации реализуются межслоевые соединения. В результате создаётся модель, в которой указываются входные и выходные слои.

3.4. Функция потерь и метрики

Алгоритмы глубокого обучения используют метод стохастического градиентного спуска. Это метод для нахождения минимума или максимума целевой функции [1]. Целевая функция — это математическое представление того, насколько нейронная сеть делает свою работу хорошо (см. рис.3.12).

Рисунок 3.12. Градиентный спуск и целевая функция

Для решения задачи сегментации могут применяться различные функции или их комбинации. Одними из самых популярных являются функции Binary cross-entropy и Dice (см. рис.3.13).

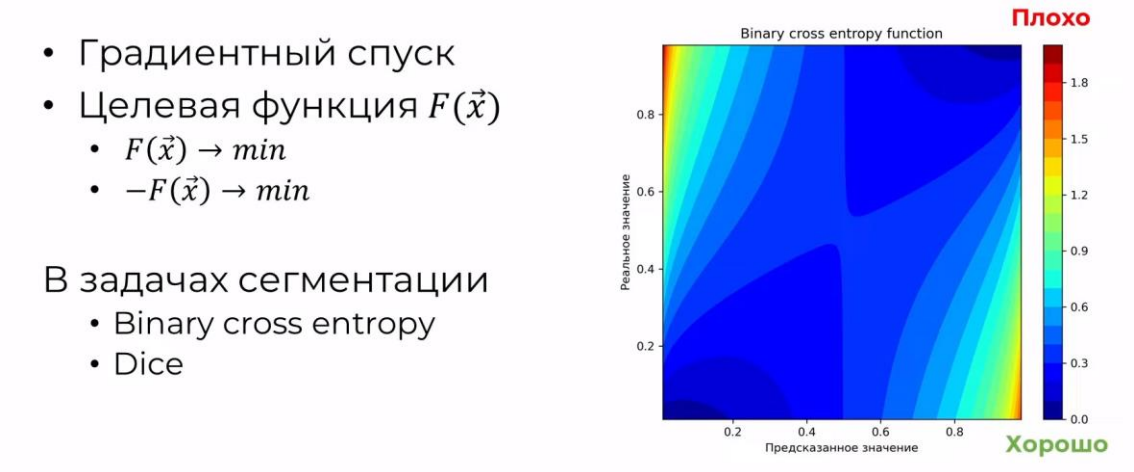


Рисунок 3.13. Бинарная кросс-энтропия и Dice

Если нейронная сеть дает ответ, близкий к правильному, то значение функции потерь будет маленьким, в противном случае — большим. Верный выбор функции потерь влияет на скорость и качество обучения модели [6].

В данной работе использована комбинация функций Binary cross-entropy и Dice, математическое представление которых дано на рис. 3.14.

Первая даёт хорошую сходимость модели, обучение проходит более стабильно при сбалансированном наборе данных. Вторая очень хорошо показывает себя в задачах сегментации, но из-за своей природы обеспечивает не самую лучшую сходимость.

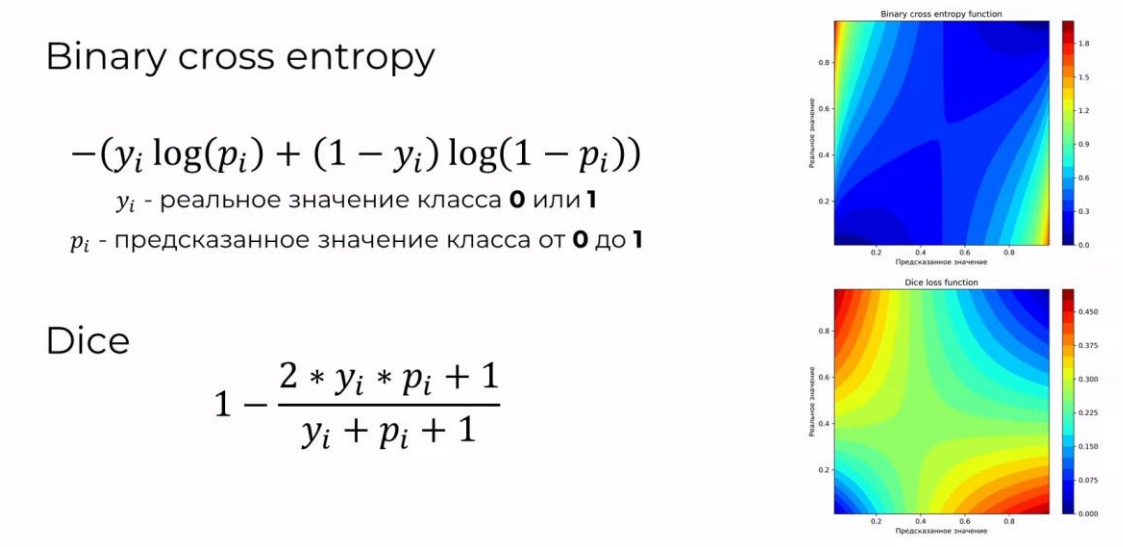


Рисунок 3.14. Математическое представление Binary cross-entropy и Dice

Этот выбор помогает справиться с плохо сбалансированными наборами данных и добиться приемлемых результатов обучения.

Для оценки точности работы нейронной сети использовался упомянутый выше коэффициент Dice и функция потерь (см. рис.3.15).

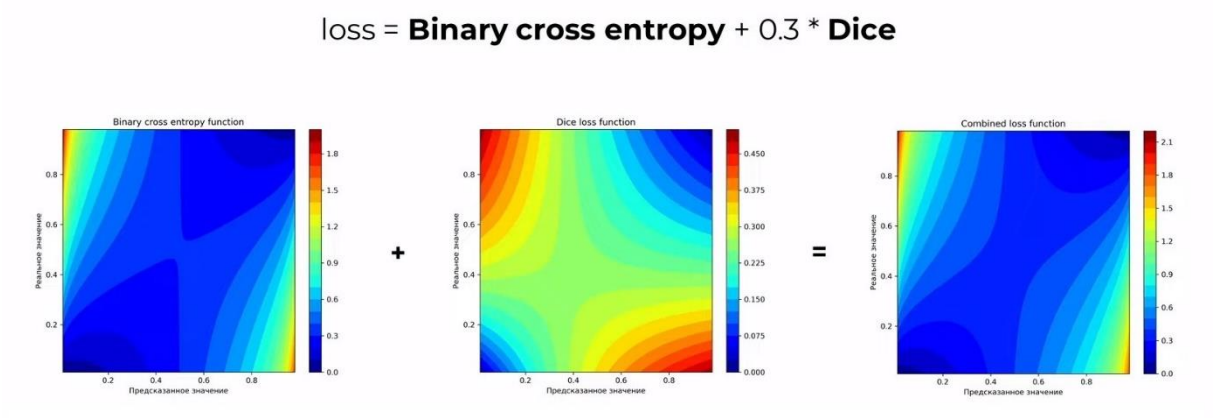


Рисунок 3.15. Функция потерь

На уровне программного кода это реализуется следующим образом.

Определяется функция «multi_class_metric». Она принимает 2 параметра, один из них — это ответ нейронной сети, а второй — результат, который должен получиться на самом деле: то, чему старается научиться нейронная сеть. Метрика дает оценку точности результатов работы нейронной сети. Функция «unstack» распаковывает многоканальные изображения маски, и с помощью цикла находится среднее значение коэффициента Dice по всем классам объектов, которые присутствуют на сцене. $y_i + p_i + 1$

Функция потерь (см. рис.3.16) реализуется через «dice_mc_loss» (dice multi-class loss):

$$1 - \frac{2y_i p_i + 1}{y_i + p_i + 1}, \tag{3.2}$$

где y_i - количество истинно-положительных пикселей для класса i на изображении, p_i - количество предсказанных положительных (true positive) пикселей для класса i на изображении. Добавление единицы в числитель и

знаменатель помогает избежать деления на ноль в случае, если в изображении отсутствуют как истинно-положительные, так и предсказанные пиксели для данного класса.

А комбинированная функция «dice_bce_mc_loss» (dice binary cross-entropy multi-class loss) состоит из суммы двух функций: стандартная функция бинарной кросс-энтропии берётся из TensorFlow, а влияние функции dice уменьшается с помощью коэффициента 0,3.

3.5. Обучение нейронной сети

Когда все компоненты готовы, нейронную сеть можно обучать. Модель с выбранными функциями потерь компилируется с метрикой. В качестве алгоритма используется Adam.

Алгоритм оптимизации Adam использует два основных момента: момент градиента первого порядка (обычно называемый "момент") и момент градиента второго порядка (обычно называемый "экспоненциально убывающее скользящее среднее квадрата градиента").

Пусть θ - параметры модели, gt - градиент функции потерь по параметрам θ на итерации t, mt - оценка первого момента (момента), vt - оценка второго момента.

Алгоритм Adam выполняет следующие шаги на каждой итерации t:

1. Вычисление градиента gt по параметрам θ.
2. Обновление экспоненциально убывающего среднего первого момента: $m_t = \beta_1 m_t - 1 + (1 - \beta_1) \cdot g_t$, где β_1 - параметр, обычно выбираемый близким к 1.
3. Обновление экспоненциально убывающего среднего второго момента: $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$, где β_2 - параметр, обычно выбираемый близким к 1
4. Коррекция смещения первого момента: $\widehat{m_t} = \frac{m_t}{1-\beta_1^t}$
5. Коррекция смещения второго момента: $\widehat{v_t} = \frac{v_t}{1-\beta_2^t}$

6. Обновление параметров модели: $\theta_{t+1} = \theta - \frac{\eta}{\sqrt[2]{\widehat{v_t}+\epsilon}} \cdot \widehat{m_t}$, где η - learning rate, ϵ - небольшое число для численной стабильности (обычно около 10^{-8}).

Затем запускается обучение нейронной сети. Чтобы с большой вероятностью достичь оптимального уровня обученности, было решено запустить обучение на 25 эпох и принимать решение об остановке исходя из текущих значений метрик. После того, как обучение завершится, модель сохраняется в файл. На рис. 3.16 изображён результат сегментации изображений после первой итерации обучения.

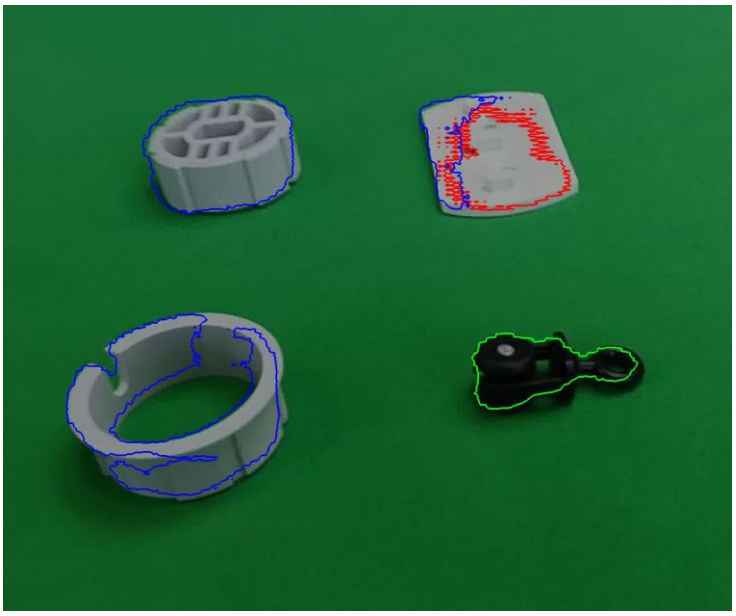


Рисунок 3.16. Сегментация после итерации 1

Контуры объектов в основном определяются не верно. Соответственно, классы объектов, тоже.

На второй и третьей итерациях уже различимы очертания объектов с небольшими неточностями, классы определяются верно (см. рис.3.17, 3.18).

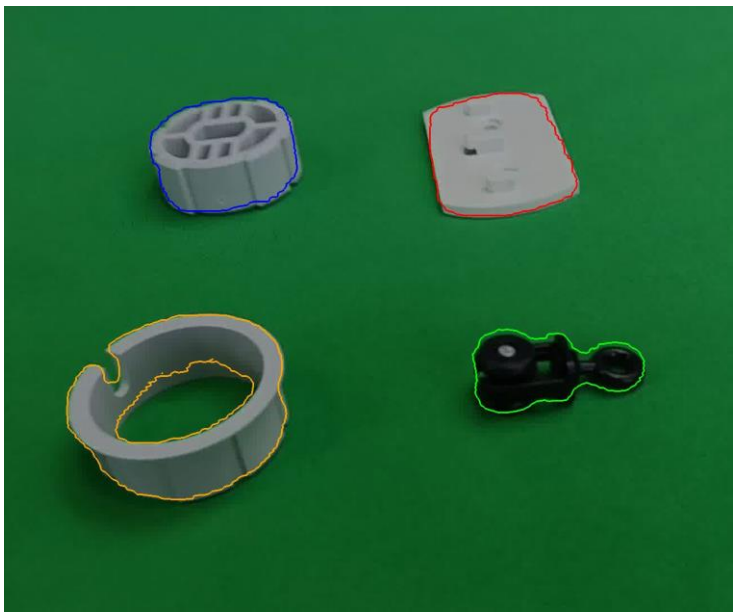


Рисунок 3.17. Сегментация после итерации 2

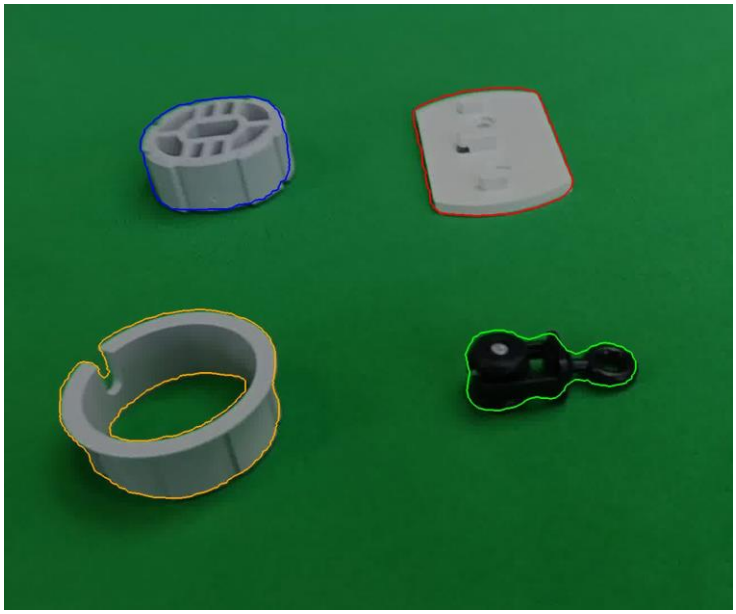


Рисунок 3.18. Сегментация после итерации 3

На 3 и 4 эпохах обучения (см. рис.3.18, 3.19), все классы в большинстве случаев определяются верно, а также происходит незначительное увеличение точности работы модели, при этом границы объектов определяются все лучше и лучше.

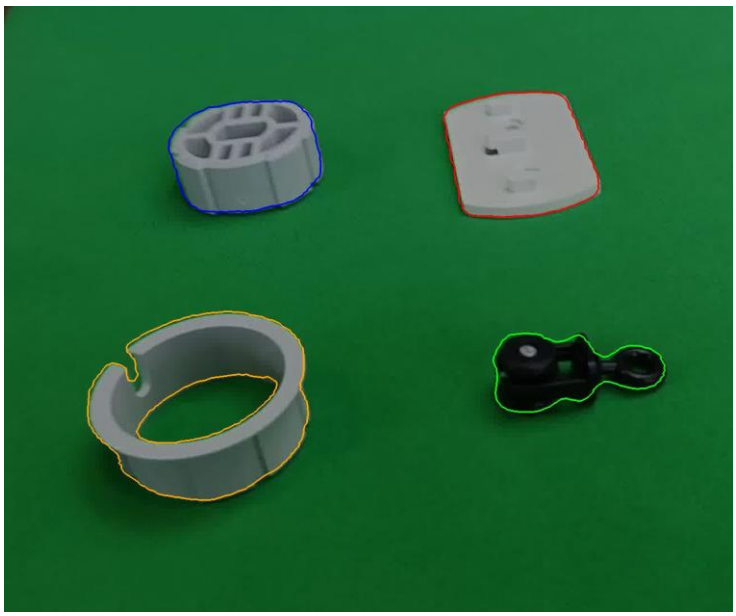


Рисунок 3.19. Сегментация после итерации 4

После 4-й итерации изменений в определении контуров и классов практически нет, это видно на рис. 3.20, 3.21, 3.22.

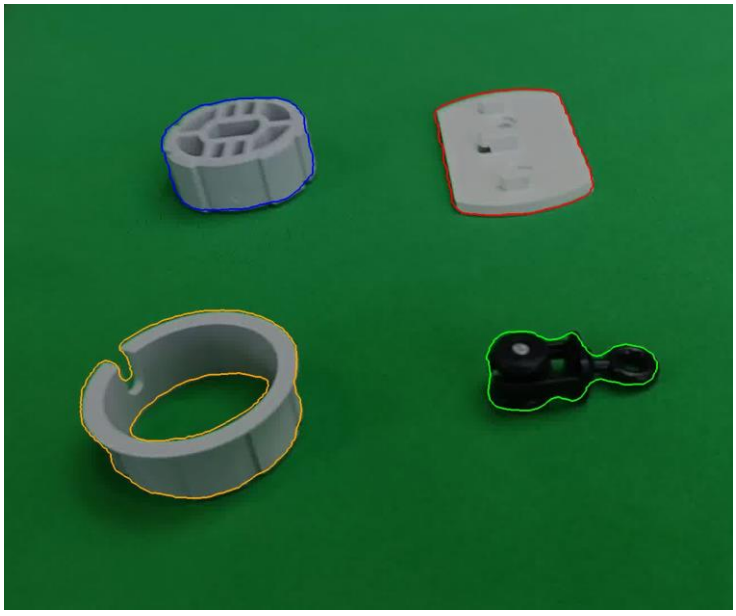


Рисунок 3.20. Сегментация после итерации 5

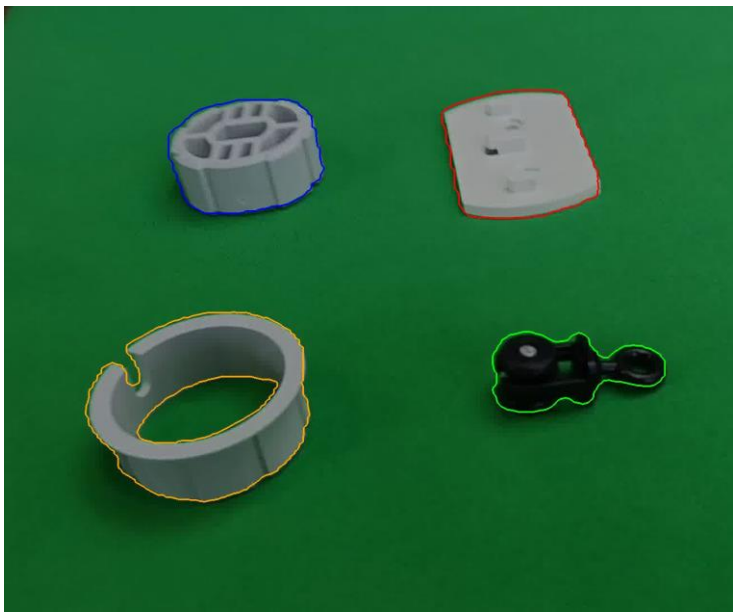


Рисунок 3.21. Сегментация после итерации 10

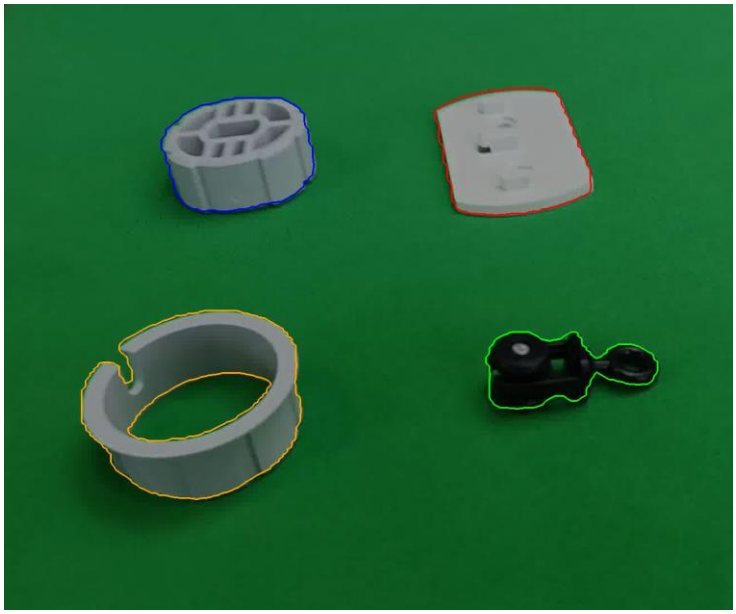


Рисунок 3.22. Сегментация после итерации 15

При этом, если посмотреть на качество обучения модели, используя метрики dice и loss (функция потерь) на обучающих и валидационных данных, то на графиках прослеживается та же тенденция (см. рис. 3.23).

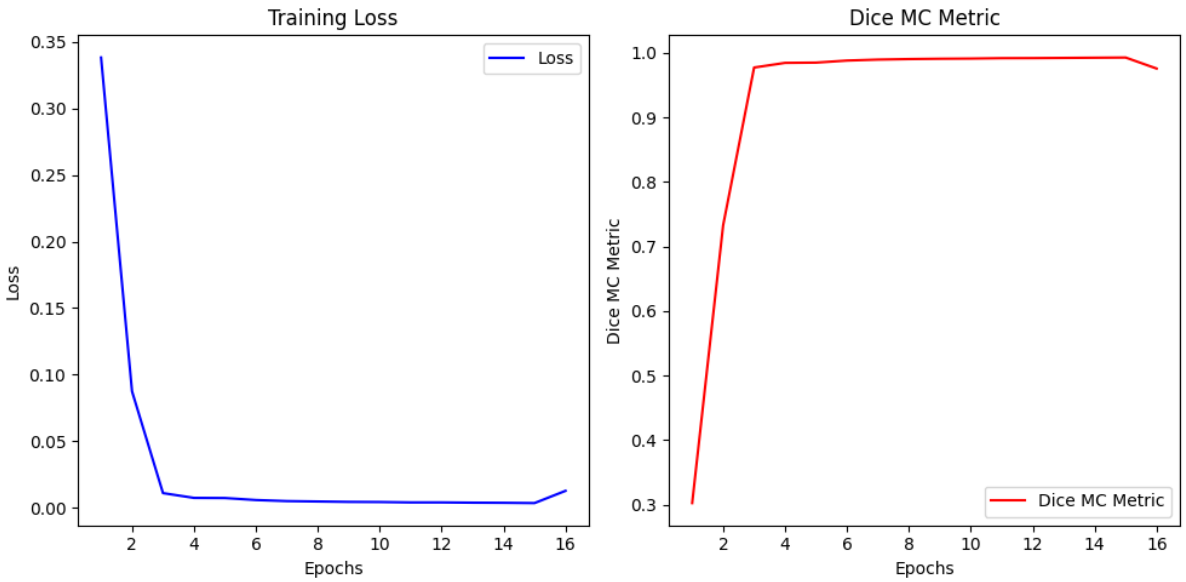


Рисунок 3.23. Графики метрик Dice и Loss

Такой перегиб графиков в районе 4-й эпохи, за которым они (графики) становятся практически горизонтальными, указывает на следующее:

- Быстрое улучшение модели: в начале обучения модель быстро учится и адаптируется к обучающим данным, что приводит к резкому улучшению значения метрики и снижению функции потерь.
- Сходимость: после резкого улучшения кривая стабилизируется, что может свидетельствовать о достижении оптимальных параметров модели или сходимости алгоритма оптимизации.
- Насыщение обучения: горизонтальность графика после резкого улучшения может указывать на то, что модель достигла своего потенциального предела в данном наборе данных, и дальнейшее обучение не приводит к значительному улучшению.

Кроме того, после 15 итерации обучения происходит ухудшение метрик, что говорит о переобученности модели. Поскольку метрики отслеживались непрерывно в процессе обучения, то сразу после их ухудшения, обучение было остановлено.

На основе визуальной оценки качества определения контуров объектов и отнесения их к классам, а также после анализа метрик, был сделан вывод о том, что оптимального уровня обученности нейросетевая модель достигает на 4–5 итерации и дальнейшее обучение не имеет существенного смысла, а после 15 итерации даже вредно.

Далее с помощью уже обученной нейронной сети обрабатывался видеофайл. Из видео извлекаются все кадры и каждый из них обрабатывается нейросетью. Общее количество получившихся изображений 1338. Важно отметить, что 42 из них принимали непосредственное участие в обучении нейросетевой модели. В большинстве случаев изображения обрабатываются правильно, качество выделения границ объектов удовлетворительное (см. рис.3.24).

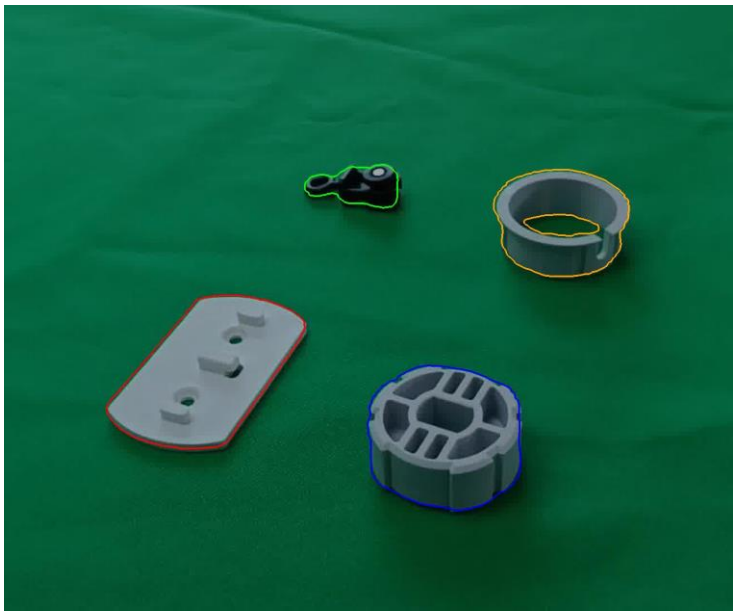


Рисунок 3.24. Сегментация на исходном видео вне обучающей выборки

Далее на вход нейронной сети подано видео, в котором изменено положение предметов на сцене. Таких изображений и ракурсов нейронная сеть раньше не видела. Часть объектов определяется верно, часть - с ошибками, что продемонстрировано на рис.3.25.

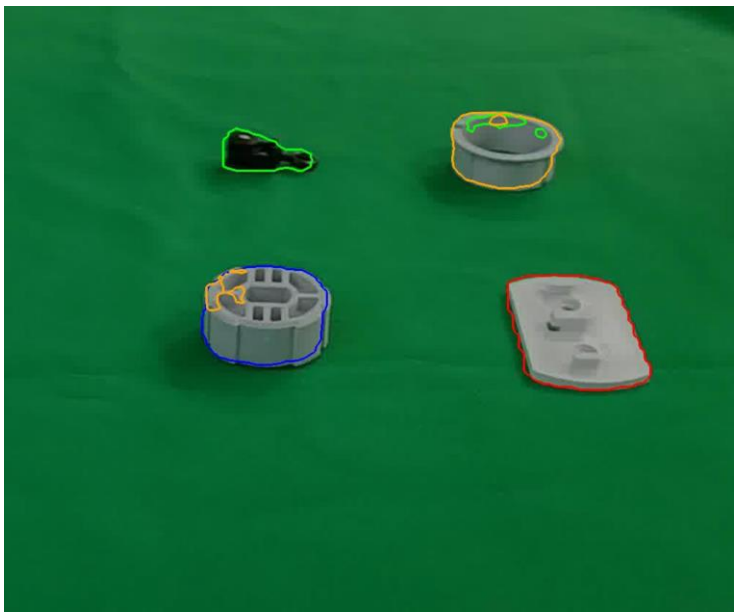


Рисунок 3.25. Сегментация на данных нового для нейросети видео

Чтобы исправить ситуацию, в обучающий набор добавлены несколько размеченных вручную изображений из нового видео. При этом нейронная сеть обучается не с нуля, достаточно продолжить обучение ранее сохраненной модели. В результате дообучения, объекты определяются корректно (см. рис.3.26).

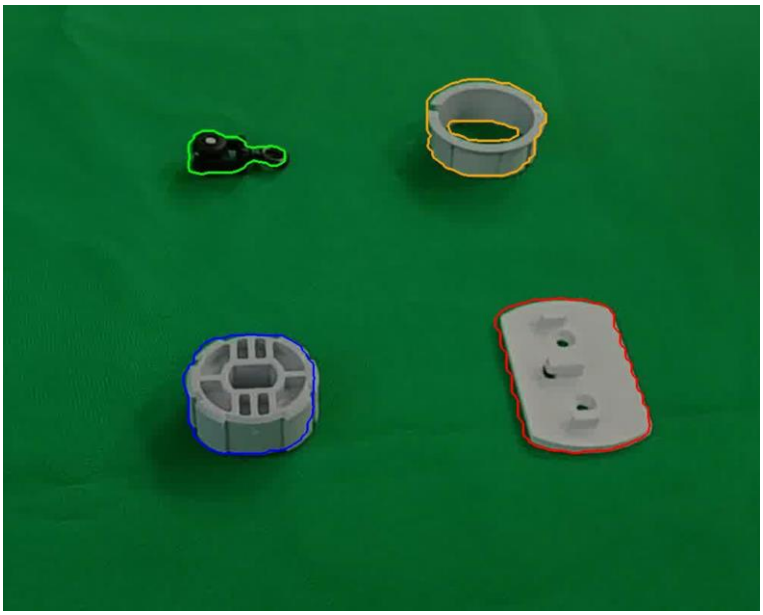
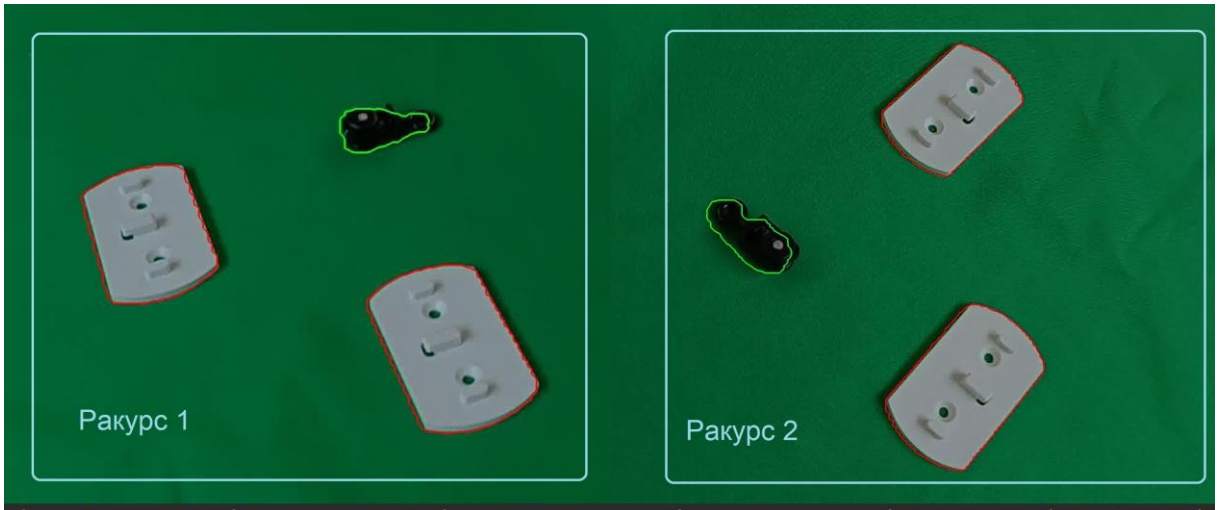


Рисунок 3.26. Сегментация на данных нового для нейросети видео после дообучения

Далее, после дообучения модели, в нейронную сеть были поданы три разных видео, в которых количество предметов было меньше, равно и больше, чем в обучающем наборе. При этом некоторых классов предметов в них не было, а предметы отдельных классов присутствовали в количестве большем, чем 1 шт., то есть не равном тому, что было в обучающем наборе.

Также был написан и включен в алгоритм программный модуль, содержащий спецификации, которые сопоставлялись с результатами распознавания моделью.

В первом видео было 3 предмета. В спецификацию намеренно были поданы ложные данные для проверки корректности работы модуля. Результаты представлены на рис.3.27.



Наименование	Цвет контура	Кол-во в спец.	Кол-во факт.	Добавить	Убавить
Адаптер	голубой	1	0	1	0
Бегунок	зелёный	2	1	1	0
Кольцо	оранжевый	0	0	0	0
Пластина	красный	1	2	0	1

РЕЗУЛЬТАТ: есть ошибки

Рисунок 3.27. Результаты распознавания видео с тремя предметами и ложной спецификацией

Модель верно определила классы и количество экземпляров, а также были верно рассчитаны расхождения относительно ложной спецификации.

Во втором видео общее количество предметов 4, как в обучающем наборе, но другой состав. Спецификация соответствовала действительности.

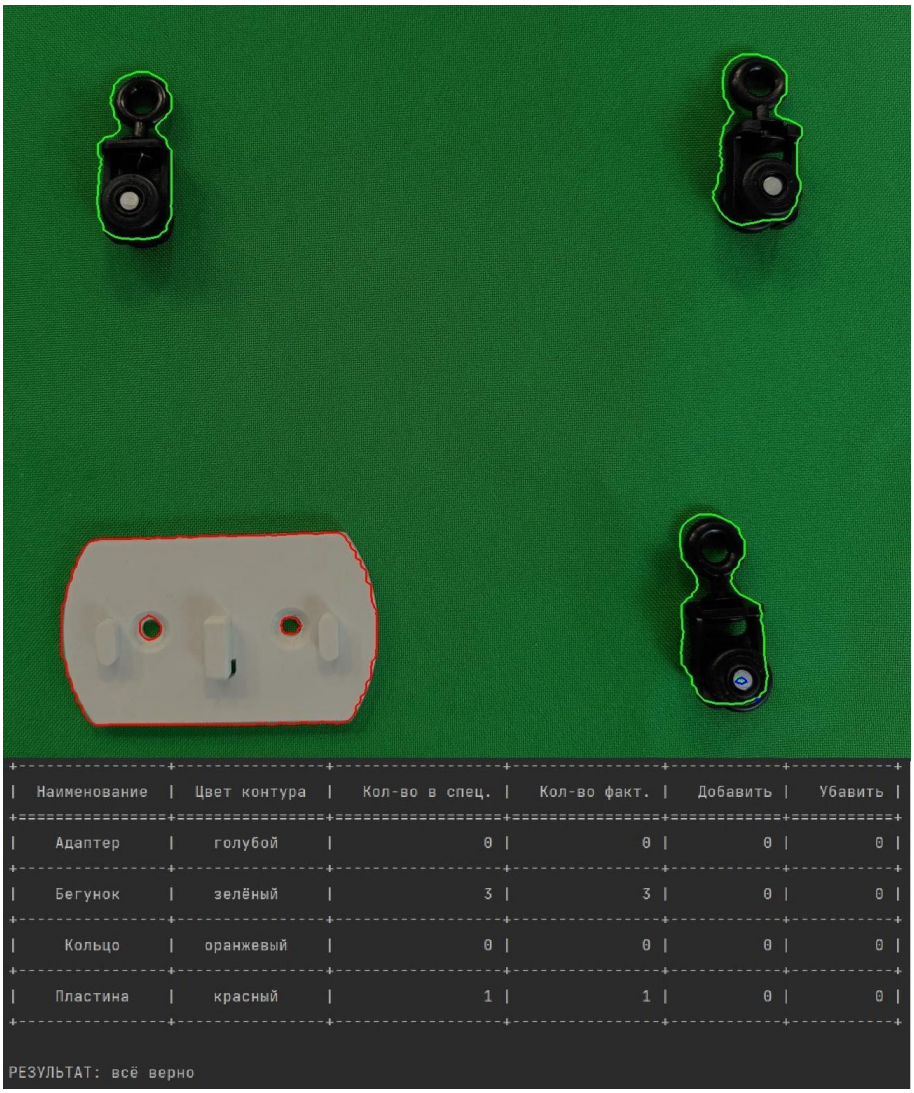


Рисунок 3.28. Результаты распознавания видео с четырьмя предметами и верной спецификацией

Все четыре объекта и количества экземпляров классов определились правильно, проверка на соответствие спецификации пройдена (см. рис. 3.28).

В третьем видео на сцене было 7 предметов трёх классов. Спецификация соответствовала реальному составу комплекта.

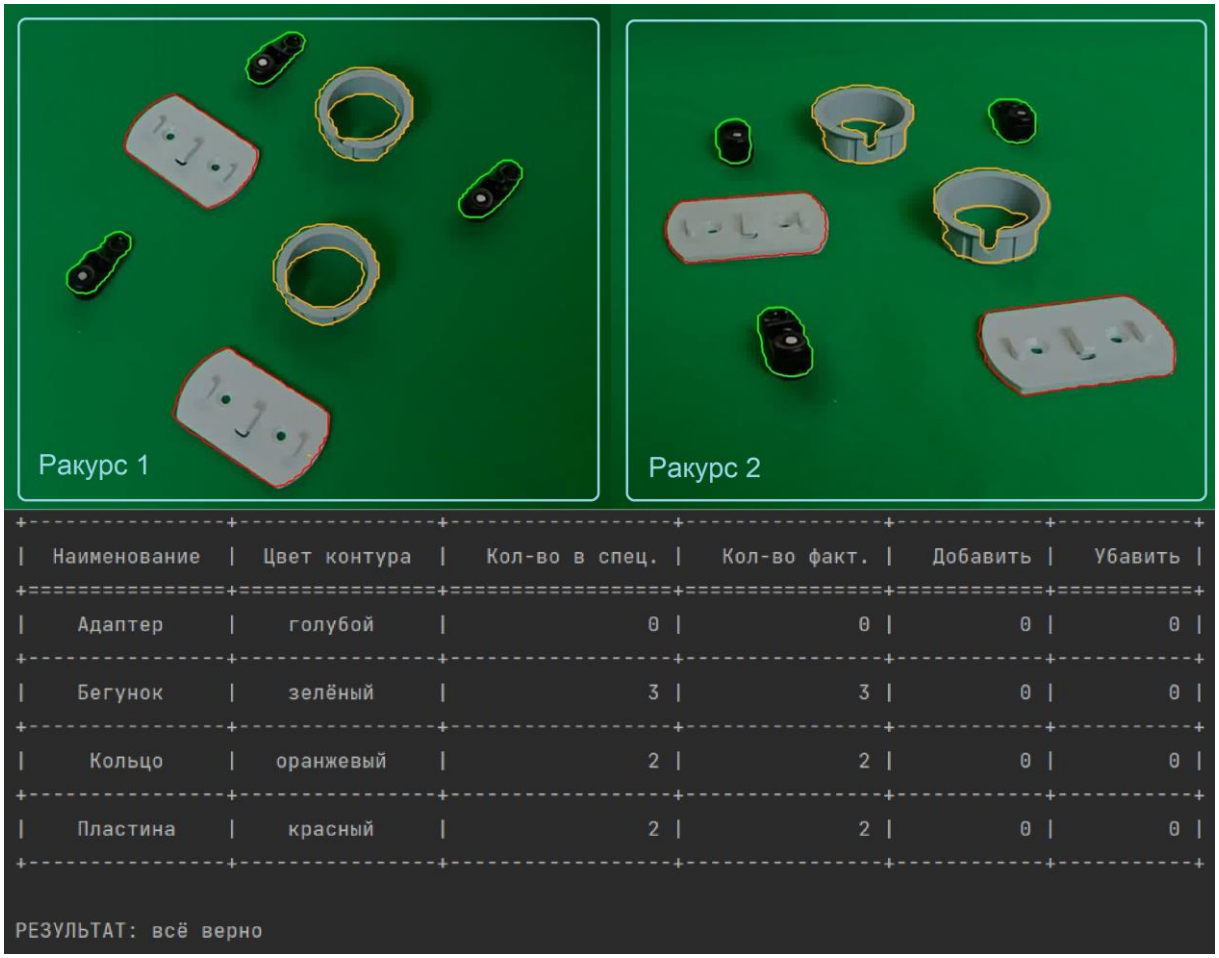


Рисунок 3.29. Результаты распознавания видео с семью предметами и верной спецификацией

Все семь предметов распознаны и корректно распределены по классам. Выявлено соответствие спецификации, как видно на рис.3.29.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы была изучена литература на темы, связанные с машинным обучением, машинным зрением и со структурами искусственных нейронных сетей.

Также были изучены имеющиеся методы и проанализирована их эффективность в областях, связанных с решением поставленных задач.

Итоговым результатом данной работы стала компьютерная программа со следующим функционалом:

- загрузка размеченных наборов данных о предметах, класс и количество которых необходимо определять
- обучение и дообучение нейросетевой модели на дополняемых наборах данных
- контроль качества распознавания моделью предметов с помощью набора метрик
- загрузка спецификаций изделий, в которых указаны классы и количества предметов, которые нужно распознать и сверить со спецификацией.

При разработке программы была использована свёрточная нейронная сеть с архитектурой U-Net. Такой подход имеет преимущества, делающие его оптимальным выбором метода решения поставленной задачи:

- позволяет точно определять границы объектов разных форм и размеров, и выделять их с высокой точностью
- высокая скорость работы позволяет одновременно обрабатывать все объекты без повторного сканирования
- простота архитектуры и относительно лёгкая настройка для разных типов данных и задач
- высокое качество сегментации и точность определения объектов

В ходе выполнения задачи была обучена модель распознавания некоторых комплектующих солнцезащитного оборудования, а также написан программный

модуль, содержащий в себе данные спецификаций и сверяющий полученные от нейросети данные с соответствующей спецификацией.

Несмотря на положительные результаты, данный подход имеет некоторые решаемые недостатки и ограничения.

1. Ограничение по размеру объектов: текущая модель U-Net имеет тенденцию искажать или упрощать очертания мелких объектов на изображении. Это может привести к утере деталей и недостаточному распознаванию мелких объектов.

Решение:

- использование более мелких фильтров: замена некоторых свёрточных слоев в модели более мелкими фильтрами может помочь распознавать более мелкие объекты на изображении
- использование многошаговой сегментации: для улучшения точности сегментации объектов можно использовать методы многошаговой сегментации, включая постобработку сегментированных изображений и уточнение контуров объектов
- использование более мощного компьютера для реализации предыдущих двух пунктов.

2. Качество работы модели существенно зависит от качества набора данных, на которое в свою очередь влияют:

- качества съёмки (освещение, оборудование и др.)
- качества разметки данных (точнее размечены данные - выше качество)

Решение:

- съёмку проводить в тех же условиях и тем же оборудованием, которые будут использоваться в реальных рабочих условиях
- выполнить все возможные преднастройки программного обеспечения для разметки, чтобы оператор имел минимальное влияние на качество.

3. Ограниченность количества объектов, которые должна распознавать система, вызванная трудоёмкостью разметки.

Решение: интеграция программы с сервисом разметки.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Гольдберг Й. Нейросетевые методы в обработке естественного языка / Йоав Гольдберг : перевод с английского А. А. Слинкина. – М.: ДМК Пресс, 2019. – 282 с.: ил. – С. 48-52. – ISBN 978-5-97060-754-1
2. Лекун, Я. Как учится машина: Революция в области нейронных сетей и глубокого обучения / Ян Лекун : перевод с французского Е. Арсеновой. – М.: Альпина ПРО, 2021. – С. 226-230. – ISBN 978-5-907394-29-2
3. Рассел, С. Искусственный интеллект: современный подход, 4-е издание, том 3. Обучение, восприятие и действие. / Стюарт Рассел, Питер Норвиг : перевод с английского А. В. Слепцова – СПб. : ООО "Диалектика", 2022 – 640 с. – с. 167-169. – ISBN 978-5-907365-27-8 (рус., том 3)
4. Рашка, С. Python и машинное обучение: машинное и глубокое обучение с использованием Python, scikit-learn и TensorFlow 2 / Себастьян Рашка, Вахид Мирджалили : перевод с английского и ред. Ю. Н. Артеменко. – 3-е изд – СПб. : ООО "Диалектика", 2020. – 848 с.: ил. – С. 609-625. – ISBN 978-5-907203-57-0 (рус.)
5. Сохина, С. А., Немченко С. А. Машинное обучение. Методы машинного обучения // СОВРЕМЕННАЯ НАУКА МОДЕРНИЗАЦИОННЫХ ПРОЦЕССОВ: ПРОБЛЕМЫ, РЕАЛИИ, ПЕРСПЕКТИВЫ / Сборник научных статей по материалам IV Международной конференции (5 января 2021 г., г. Уфа) / – Уфа: Изд. НИЦ Вестник науки, 2021. – С. 165-168.
6. Чернавин, П. Ф. Машинное обучение на основе задач математического программирования / П. Ф. Чернавин, Д. Н. Гайнанов, В. Н. Панкращенко [и др.] – М. : Наука, 2021. – 128 с. – С. 42-48. – ISBN 978 5 02 040908 8.
7. Botvinick M. et al. Reinforcement learning, fast and slow // Trends in cognitive sciences. – 2019. – Т. 23. – №. 5. – pp. 408-422.
8. Dridi S. Supervised Learning-A Systematic Literature Review. – 2021.
9. Dridi S. Unsupervised Learning - A Systematic Literature Review. – 2021.

10. Girshick R. et al. Rich feature hierarchies for accurate object detection and semantic segmentation // Proceedings of the IEEE conference on computer vision and pattern recognition. – 2014. – pp. 580-587.
11. Han S. H. et al. Artificial Neural Network: Understanding the Basic Concepts without Mathematics // Dementia and Neurocognitive Disorders. – 2018. – T. 17. – №. 3. – pp. 83-89.
12. Liu B., Liu B. Supervised learning. – Springer Berlin Heidelberg, 2011. – pp. 63-132.
13. Reddy Y., Viswanath P., Reddy B. E. Semi-supervised learning: A brief review // Int. J. Eng. Technol. – 2018. – T. 7. – №. 1.8. – 81 p.
14. Redmon J. et al. You only look once: Unified, real-time object detection // Proceedings of the IEEE conference on computer vision and pattern recognition. – 2016. – pp. 779-788.
15. Ren S. et al. Faster r-cnn: Towards real-time object detection with region proposal networks // Advances in neural information processing systems. – 2015. – T. 28.
16. Rosenfeld A. Computer vision: basic principles // Proceedings of the IEEE. – 1988. – T. 76. – №. 8. – pp 863-868.
17. Salehinejad H. et al. Recent advances in recurrent neural networks // arXiv preprint arXiv:1801.01078. – 2017.
18. Sazli M. H. A brief review of feed-forward neural networks // Communications Faculty of Sciences University of Ankara Series A2-A3 Physical Sciences and Engineering. – 2006. – T. 50. – №. 01.
19. TensorFlow in Action THUSHAN GANEGEDARA MANNING SHELTER ISLAND. – USA : Manning Publications Co. – 2022. – pp 194-203. – ISBN: 9781617298349.
20. Uijlings J. R. R. et al. Selective search for object recognition //International journal of computer vision. – 2013. – T. 104. – pp 154-171.

21. Wang C. Y. et al. CSPNet: A new backbone that can enhance learning capability of CNN //Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. – 2020. – pp 390-391.
22. Wu J. Introduction to convolutional neural networks //National Key Lab for Novel Software Technology. Nanjing University. China. – 2017. – Т. 5. – №. 23. –pp 495.