Федеральное государственное образовательное бюджетное учреждение  
высшего образования

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ**

|  |
| --- |
| **РОССИЙСКОЙ ФЕДЕРАЦИИ»** |
| Факультет информационных технологий и анализа больших данных |

Департамент анализа данных и машинного обучения

Выпускная квалификационная работа

на тему: «Разработка программного приложения для идентификации повреждений плодов яблок»

Направление подготовки:01.03.02 Прикладная математика и информатика

Профиль: Анализ данных и принятие решений в экономике и финансах

|  |  |  |
| --- | --- | --- |
|  | Выполнил студент учебной группы ПМ18-1 | |
|  | | Федоров Дмитрий Андреевич |
|  | |  |
|  | | Научный руководитель работы к.т.н., доцент |
|  | | Хорт Дмитрий Олегович |
|  | |  |

**ВКР соответствует предъявляемым требованиям:**

Руководитель Департамента анализа данных и машинного обучения, к.т.н., доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Д.А. Петросов

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_2023 г.

Москва 2023

Содержание

[1. Введение 4](#_Toc135548227)

[2. Описание проблемы 7](#_Toc135548228)

[3. Анализ существующих решений 13](#_Toc135548229)

[3.1 Определение повреждений яблок с использованием Multi-Scale Dense Classification Network 13](#_Toc135548230)

[3.1.1 Набор данных 13](#_Toc135548231)

[3.1.2 Алгоритм 15](#_Toc135548232)

[3.1.3 Результаты 17](#_Toc135548233)

[3.2 Определение повреждений яблок с при помощи анализа нескольких признаков (Multi-feature Fusion) 19](#_Toc135548234)

[3.2.1 Описание оборудования 19](#_Toc135548235)

[3.2.2 Определение цвета яблок 20](#_Toc135548236)

[3.2.3 Определение размера яблок 21](#_Toc135548237)

[3.2.4 Определение формы яблок 22](#_Toc135548238)

[3.2.4 Определение повреждений 23](#_Toc135548239)

[3.2.5 Результаты 23](#_Toc135548240)

[3.3 Выявление дефектов яблок на основе FCM-NPGA и многомерного анализа изображений 24](#_Toc135548241)

[3.3.1 Набор данных 24](#_Toc135548242)

[3.3.2 Алгоритм 25](#_Toc135548243)

[3.3.3 Результаты 26](#_Toc135548244)

[4. Разработка модели 27](#_Toc135548245)

[4.1 Теоритечесая справка 27](#_Toc135548246)

[4.1.1 Предобработка изображений 27](#_Toc135548247)

[4.1.2 Определение объектов 30](#_Toc135548248)

[4.1.3 YOLO модель 35](#_Toc135548249)

[4.1.4 Классификация изображений 37](#_Toc135548250)

[4.2 Описание датасетов 41](#_Toc135548251)

[4.2.1 Датасет для определения яблок 41](#_Toc135548252)

[4.2.2 Датасет для определения болезней яблок 43](#_Toc135548253)

[4.3 Разработка модели 45](#_Toc135548254)

[4.3.1 Определение яблок на фото 45](#_Toc135548255)

[4.3.2 Классификация болезней 50](#_Toc135548256)

[5. Заключение 53](#_Toc135548257)

[Литература 56](#_Toc135548258)

[Приложения 58](#_Toc135548259)

[Приложение 1 58](#_Toc135548260)

[Приложение 2 61](#_Toc135548261)

# 1. Введение

С развитием компьютерных алгоритмов и рынка технологий для сельскохозяйственной отрасли повышаются и ожидания качества и стандартов производимых продуктов, а следовательно, и потребность в быстром и точном определении характеристик пищевых продуктов и их классификация. В данный момент большинство процессов по контролю качества продуктов во время их выращивания и на стадии сбора выполняются в полу-ручном или полностью ручном формате, что занимает большое количество времени и подвержено ошибкам, связанным с человеческим фактором, невнимательностью. Компьютерное зрение представляет собой одну из возможностей для автоматизации процесса контроля качества, которая не требует значительных изменений в процессе выращивания фруктов и овощей. Анализ и обработка изображений находят широкое применение в процессах сельскохозяйственной промышленности. Проведенные исследование и тесты подтверждают потенциал использования компьютерного зрения для выявления повреждений на плодах и задач их классификации.

В ходе данной работы будет описана и рассмотрена проблема автоматического определения повреждений и болезней плодов яблок с помощью технологий компьютерного зрения, существующие разработки и их применение.

Сельскохозяйственная отрасль входит в десятку ведущих отраслей, в которых применение компьютерного зрения может принести значительный эффект. Потенциал внедрения данной технологии был исследован и признан еще до начала 21 века. Автоматизация процесса контроля качества выращиваемых плодов стала более возможной в последние года из-за создания новых алгоритмов и методов обработки изображения, которые позволяют упростить процесс внедрения и сделать его дешевле.

В ходе проведенных исследований были выявлены основные принципы и шаги обработки изображений:

* Low-level processing (низкоуровневая обработка) – первый уровень обработки изображений, который включает в себя получение изображения от камер и перевод его в цифровую форму и предварительную обработку, которая приводит изображения к единому формату, убирая шумы, искажения и корректируя цветовые уровни.
* Intermediate-level processing (промежуточная обработка) – уровень обработки, на котором происходит выделение объекта на изображении. От точности выполнения промежуточной обработки зависит точность выполнения дальнейшего определения повреждений. Сегментация и выделение объектов может быть осуществлена 3 разными способами: пороговая сегментация, сегментация на основе краев и сегментация на основе областей. Результатом промежуточной обработки является выделение плода на изображении.
* High-level processing (высокоуровневая обработка) – последний уровень обработки, в котором происходит распознавание поврежденных плодов и их классификация. Высокоуровневая обработка включает в себя использование алгоритмов глубокого обучений и статистических методов.

В каждом из 3 типов обработки идет непрерывное взаимодействие с базой знаний в которой хранятся необходимые материалы (коэффициенты моделей, примеры для обучения и классификации) для выполнения точной классификации. Для повышения качества автоматизации процесса контроля качества плодов базу знаний необходимо пополнять материалами и периодически пересчитывать коэффициенты моделей.

***Рисунок 1. Общий алгоритм обработки изображений***

Diagram

Description automatically generated

# 2. Описание проблемы

На данный момент яблочная промышленность является одной из самых быстро развивающихся среди всех фруктов в мире.

Качество выращиваемых яблок напрямую определяет возможности для дальнейшего развития яблочной промышленности. В то же время различные повреждения и болезни яблок, и несвоевременное их предотвращение становится одним из ключевых показателей в качестве плодов и доходности бизнеса по выращиванию яблок, что также напрямую наносит вред для развития всей сельскохозяйственной отрасли. В связи с этим решение проблемы определения и распознавания повреждений и болезней яблок поможет повысить качество конченого продукта и внесет положительное влияние на отрасль и инвестиции в нее.

В настоящее время определение поврежденных плодов выполняется в ручном или полуавтоматическом режиме и сильно зависит от человеческого фактора. В настоящее время автоматизация используется при сортировке собранных плодов по весу и форме.

Одной из основных проблем автоматизированного распознавания поврежденных плодов является внешнее сходство некоторых болезней и их стадий, что может привести к большим расхождениям в результатах и несвоевременному устранению проблем, которое отразится на общем качестве урожая фруктов и нанесет ущерб бизнесу. Решением этой проблемы становится использование нейронных сетей и алгоритмов глубокого обучения (Deep learning) для повышения точности результатов. Параллельное внедрение технологий IoT (Internet of Things – интернет вещей) помогает постоянно контролировать качество выращиваемых плодов, что уменьшает количество затрачиваемого человеком времени и сокращает влияние человеческого фактора на процесс контроля качества яблок.

Как было описано ранее, задача определения повреждений на плодах подразделяется на 3 основных подзадачи:

* Сбор и первичная обработка изображений
* Выделение плодов яблок на изображении
* Определение повреждений плода

В ходе данной работы основными элементами исследования станут выделение плодов яблок на изображении и последующее определение повреждений на них. В качестве данных для обучения модели будут использоваться готовые и промаркированные наборы данных.

Далее будут рассмотрены несколько примеров существующих решений и принцип их работы. На основании приведенных примеров будет сформирован финальный план создания и тестирования модели.

Практическое применение модели, определяющей повреждённые плоды имеет две основных области применения:

* Идентификация поврежденных плодов на производстве (в садах) для своевременного реагирование и устранения причин болезни \ вырубка больного дерева и тд
* Классификация плодов в распределительных центрах магазина для оценки качества и сверки с заявленным в договоре

Одними из самых распространенных и наиболее опасных для экономики производства яблок являются следующие болезни и повреждения:

1. Физические повреждения – плоды яблок поврежденные при помощи физического воздействия погоды, птиц или других вариантов. Физические повреждения не несут за собой серьезных последствий для всего урожая или конкретных деревьев, так как нанесены точечно и не способно распространиться.

***Рисунок 2. Физическое повреждение яблока***



1. Парша (apple scab) – болезнь приводит к образованию темных пятен или повреждений на листьях, плодах, а иногда и на молодых веточках. Пораженные растения могут преждевременно сбрасывать плоды. Поражаются все виды яблонь, хотя некоторые сорта обладают большей устойчивостью. Яблоневая парша встречается везде, но наиболее сильно проявляется в местах с прохладным и влажным климатом. Это заболевание может привести к большим потерям урожая и поэтому имеет экономическое значение для производителей яблок.

***Рисунок 3. Парша***



1. Плодовая гниль (ring rot) - болезнь характеризуется слегка впалыми поражениями с чередующимися коричневыми кольцами на инфицированных плода. В последние семьдесят лет высокая заболеваемость яблоневой плодовой гнилью постепенно становится серьезной проблемой в плодоводстве. С плодовой гнилью трудно бороться, поскольку патоген может заражать плоды на стадии роста и вызывать гниль во время хранения. Инфекция вызывает потери до 50% до сбора урожая и еще 79% теряется при хранении на восприимчивых сортах яблони.

***Рисунок 4. Плодовая гниль***



1. Пятнистый грибок или чернь (blotch) - распространенное заболевание, вызываемое различными грибками в течение всего сезона плодоношения. Эта проблема ограничивается кожицей яблока. Яблоки с таким заболеванием безопасны для употребления в пищу, в остутствии аллергии на плесень, поэтому для многих чернь может не представлять достаточно серьезной угрозы для здоровья. Симптомы яблоневой пятнистости обычно проявляются в виде неровных участков размером в 6 мм. или больше на поверхности зараженных плодов. Цвет может быть мутным или сажистым, часто поверхность яблока кажется оливково-зеленой. Обычно небольшие участки объединяются вместе, образуя более крупные, некруглые пятна на кожице.

***Рисунок 5. Пятнистый грибок или чернь***



# 3. Анализ существующих решений

## 3.1 Определение повреждений яблок с использованием Multi-Scale Dense Classification Network

В примере будет рассмотрено применение Multi-Scale convolutional нейронная сеть, которая используется для определения 11 различных видов заболеваний на основании анализа изображений плодов и листьев яблонь.

В работе были использованы следующие подходы:

* Для дополнения набора данных был использован метод Cycle-GAN, который позволил сгенерировать изображения двух болезней на поверхности здоровых яблок.
* Multi-scale Dense нейронная сеть была использована для определения заболеваний.

### 3.1.1 Набор данных

Изображения, используемые в данной работе, состоят из изображений плодов яблок, собранных в полевых условиях, поэтому количество изображений плодовой гнили намного меньше, чем изображений других типов болезней. Такое распределение изображений не подходит для качественного обучения нейронной сети. Для решения этой проблемы был использован алгоритм Cycle-GAN, который предназначен для переноса стиля изображения путем обучения на двух различных типах. Работу алгоритма можно описать следующим образом:

1. Формируется 2 пространства изображений со здоровыми плодами и плодами с болезнями (X и Y);
2. Алгоритму задается задача по преобразованию изображения из пространства X в пространство Y;
3. Алгоритм преобразует изображение с использованием генератора F(x);
4. Для определения принадлежит ли изображение к пространству Y алгоритм использует дискриминатор DY.

***Рисунок 5. Алгоритм создания повреждения на яблоке***

Graphical user interface, diagram

Description automatically generated

Cycle-GAN был обучен на большом количестве изображений, которые был предварительно обработаны по яркости, вращению и зеркальному отображению. После обучения алгоритм был применен для создания изображений с плодовой гнилью.

Полученные изображения, были предоставлены экспертам по болезням, которые отобрали 500 изображений каждого типа болезни, которые могли быть использованы в качестве обучающих образцов. Эти сгенерированные изображения были добавлены в набор данных.

### 3.1.2 Алгоритм

Для определения болезней на изображения со сложным фоном увеличивается количество слоев в нейронной сети. Однако, такой подход ведет к тому, что с увеличением глубины слоев нейронная сеть теряет особенности изображения из нижних слоев, что приводит к неадекватному использованию признаков изображения. В связи с этим, когда сеть достигает определенной глубины увеличение числа слоев не может продолжать улучшать ее возможности.

В качестве основы были выбраны сети Inception-V$ и Inception-ResNet-V2, так как семейство сетей inception показало высокую производительность в задачах классификации. Идеи Multi-Scale соединения и DenseNet были использованы для построения новых моделей и повышения их производительности.

Структура сетей Inception показана на рисунке 6, где Stem - базовая структура извлечения признаков, состоящая из множества операций свертки и объединения. Модуль Inception-X изучает особенности изображения с помощью архитектуры передачи признаков.

***Рисунок 6. Структура сетей Inception***

Diagram

Description automatically generated

Один из основных механизмов модуля Inception показан на рисунке 7. Существует два основных вклада структуры Inception:

1. Свертка 1×1 для повышения и понижения размерности;
2. Cвертка и агрегирование по нескольким измерениям одновременно.

Inception-ResNet-V2 вводит модуль ResNet по сравнению с Inception-V4, что еще больше улучшает производительность сети.

***Рисунок 7. Модуль Inception***

Diagram

Description automatically generated

DenseNet использовалась для объединения признаков разной глубины путем установления плотных связей между передними и задними слоями, тем самым достигая повторного использования признаков.

В качестве примера взяты четыре модуля Inception-A в Inception-V4, показанные на рисунке 8. Выход первого блока Inception обрабатывается функцией H1 и затем используется в качестве входа второго блока Inception. Выход второго блока Inception каскадируется с выходом предыдущего блока после работы функции H2. Их объединение является входом третьего блока Inception. Выход третьего блока Inception каскадируется с выходом двух предыдущих блоков Inception после выполнения функции H3 и затем принимается в качестве входа четвертого блока Inception. Наконец, выходы всех блоков Inception объединяются в качестве входа последующей сети.

***Рисунок 8. Пример работы модулей Inception***

A screenshot of a computer

Description automatically generated with low confidence

Multi-scale соединение позволяет функциям сети разной глубины функционировать в окончательном обучении, что еще больше улучшает использование функций нижнего слоя.

### 3.1.3 Результаты

Результаты определения различных заболеваний приведены в таблице 1. Точность1 - результат модели Multi-scale Dense Inception-V4, Точность2 - результат модели Multi-scale Dense Inception-ResNet-V2.

Из результатов, приведенных в таблице 1, видно, что самая высокая точность определения у здоровых плодов яблони. Поскольку различие между общим заболеванием и серьезными заболеваниями не очевидно, точность результата определения. Кроме того, признаки изображений серой пятнистости яблони и ржавчины кедровой яблони имеют определенное сходство, поэтому среди тестовых образцов выдается несколько неточных результатов диагностики. Однако, объединив результаты различных категорий, две новые модели показали превосходные результаты и могут быть использованы на практике.

***Таблица 1. Результаты применения Multi-scale Dense Inception алгоритмов***

Table

Description automatically generated

## 3.2 Определение повреждений яблок с при помощи анализа нескольких признаков (Multi-feature Fusion)

Работа описывает разработку устройства по обнаружению дефектов плодов и их сортировке в полевых условиях с использованием алгоритмов нейронных сетей. Для оценки яблок были выбраны 4 характеристики: цвет, форма, дефекты поверхности и размер. Были разработаны алгоритмы для определения и различения этих четырех признаков при помощи компьютерного зрения и других алгоритмов. После определения четыре признака были объединены в одну модель для классификации на 3 категории: яблоки первого сорта, яблоки второго сорта и яблоки других сортов.

### 3.2.1 Описание оборудования

Оборудование для сортировки яблок состояло из механизма наведения, транспортировочного устройства, устройства обнаружения, исполнительного механизма и устройства сбора, а камеры были установлены для обзора яблока с трех сторон:

***Рисунок 9. Устройство оборудования по сортировке яблок***

Diagram, engineering drawing

Description automatically generated

***Рисунок 10. Алгоритм работы оборудования по сортировке яблок***

Diagram

Description automatically generated

### 3.2.2 Определение цвета яблок

Для определения цвета использовалась модель цветового пространства HSV. Модель способна различать оттенки, насыщенность и значение яркости цвета яблока.

Оттенок менее всего подвержен влиянию освещения и наиболее подходит для определения различных цветов яблок. Для преобразования изображения в пространство HSV использовалась библиотека OpenCV. Для получения диапазона от 0 до 180 значения оттенка было поделено на 2. Для анализа из изображения были извлечены красный, желтый и фоновый компоненты.

Для финальной оценки и описания цвета яблока было использовано соотношение количества красных пикселей в оттенке к общему количеству пикселей в области яблока.

### 3.2.3 Определение размера яблок

Для определения размера яблока была использована минимальная окружность яблока, полученная с изображения с верхней камеры. На основе минимальной окружности был рассчитан максимальный диаметр поперечного сечения яблока.

### 3.2.4 Определение формы яблок

Для оценки формы яблок использовались индекс округлости и индекс формы.

Индекс округлости () высчитывался по формуле:

– площадь яблока на изображении (количество пикселей);

– периметр яблока на изображении (количество пикселей на окружности).

Чем ближе значение индекса к единице, тем более круглой считается форма.

Индекс формы () высчитывался по формуле:

– длина яблока по вертикали;

– длина яблока по горизонтали.

Значение индекса интерпретируются следующим образом: 0.8-0.9 – сплющенное, 0.9-1.0 – круглое, более 1 – вытянутое, продолговатое.

### 3.2.4 Определение повреждений

Определение повреждений выполнялось с помощью TensorFlow и модели SSD MultiBox.

Для обучения использовался, промаркированный при помощи инструмента LabelImg toolbox, набор данных, загруженный в сеть классификации MobeleNetV2.

Разработанная модель показала точность в 95% на тестовой выборке.

### 3.2.5 Результаты

Для проверки работоспособности оборудования было выбрано 510 яблок. Яблоки были вручнуб классифицированы, а после переданы для классификации при помощи оборудования. Результаты автоматического отбора яблок приведены в таблице 2.

***Таблица 2. Результаты алгоритма, анализирующего несколько факторов***

Table

Description automatically generated

Дальнейшее улучшение результатов возможно при применении вращающего устройства в оборудовании по классификации для получения более точной информации о яблоке.

## 3.3 Выявление дефектов яблок на основе FCM-NPGA и многомерного анализа изображений

Для определения дефектов яблок в работе был предложен алгоритм, основанный на сегментации при помощи Fuzzy C-means и нелинейного генетического алгоритма (NPGA), и последующего анализа изображений.

При обработке изображение было очищено и улучшено при помощи дробного дифференцирования: с изображения был удален шум и краевые точки. После обработки изображение сегментировалось и после сегментации определялись повреждения плодов.

***Рисунок 11. Алгоритм FCM-NGPA и многомерного анализа***

Diagram

Description automatically generated

### 3.3.1 Набор данных

В качестве выборки в работе были использованы 2000 яблок, приобретенных на рынке. 1200 яблок были не повреждены, а остальные 800 имели различные повреждения и были использованы для тестирования модели.

Отобранные яблоки были сфотографированы в специальных условиях, для облегчения определения яблока на изображении и повышении качества самого изображения.

Полученные изображения были обработаны при помощи алгоритма дробных вычислений, что позволило удалить шум изображений и более четко выделить границы яблока.

### 3.3.2 Алгоритм

Для сегментации поверхности яблока и выделения дефектов использовался алгоритм FCM. Однако использование только этого алгоритма является неточным, так как результаты могут быть подвержены сильному искажению из-за различий в освещении или ракурсе снимка. Для повышения точно был применен генетический алгоритм нелинейного программирования (NPGA), который позволил более четко выделить детали изображения.

***Рисунок 12. Сегментация изображения с применением FCM-NPGA***

A picture containing indoor, fruit, different, plant

Description automatically generated

После сегментации изображений был применен алгоритм многомерного анализа изображений и места повреждений были выделены на изображении.

***Рисунок 13. Сегментация изображения с применением FCM-NPGA и многомерного анализа***

A picture containing indoor, different

Description automatically generated

### 3.3.3 Результаты

Результаты эксперимента показали, что метод FCM-NPGA хорошо справляется с сегментацией изображений с выраженными геометрическими характеристиками, а метод многомерного анализа позволяет доопределить эти сегменты. Общая точность эксперимента составила 98%, что означает, что в 98% случаев дефекты на изображении были выделены верно.

# 4. Разработка модели

## 4.1 Теоритечесая справка

Ниже будет приведен набор методов и алгоритмов обработки изображений, который будет использоваться в разработке модели по определению болезней плодов яблок по фото

### 4.1.1 Предобработка изображений

Предварительная обработка изображений является важнейшим этапом в разработке приложений компьютерного зрения. Он включает в себя подготовку необработанных изображений к дальнейшему анализу путем применения различных методов для повышения их качества, нормализации характеристик и снижения вычислительной сложности. Этот этап жизненно важен для достижения точной и эффективной обработки изображений, распознавания объектов и алгоритмов машинного обучения. Далее мы опишем важность предварительной обработки изображений, ее цели и наиболее распространенные методы, используемые для подготовки изображений для приложений компьютерного зрения.

Цели предварительной обработки изображений:

1. **Уменьшение шума:** Изображения, полученные в реальных условиях часто содержат шум, который может негативно повлиять на работу алгоритмов компьютерного зрения. Основной целью предварительной обработки изображений является уменьшение шума при сохранении важных характеристик изображения. Благодаря минимизации шума алгоритмы могут лучше обнаруживать и анализировать необходимую информацию на изображении.
2. **Улучшение изображения:** Улучшение общего вида и качества изображения - еще одна важная цель предварительной обработки изображений. Методы улучшения изображения могут улучшить видимость объектов и особенностей на изображении, облегчая алгоритмам компьютерного зрения их обнаружение и анализ.
3. **Извлечение признаков:** Основной задачей приложений компьютерного зрения является распознавание и интерпретация особенностей изображения. Предварительная обработка изображения может упростить процесс извлечения признаков путем выделения определенных признаков, нормализации изображения и удаления нерелевантной информации.
4. **Нормализация изображения**: Изображения, полученные в различных условиях или с помощью различных устройств, могут иметь значительные различия в плане яркости, контрастности и распределения цветов. Нормализация этих свойств в наборе данных необходима для обеспечения стабильной работы алгоритмов компьютерного зрения.
5. **Вычислительная эффективность:** Предварительная обработка изображений позволяет снизить вычислительную сложность задач компьютерного применя методы, которые упрощают данные изображения без потери важных характеристик.

Распространенные методы предварительной обработки изображений:

1. **Преобразование в градации серого:** Преобразование цветного изображения в градации серого упрощает данные изображения за счет уменьшения количества цветовых каналов, что делает его более эффективным с вычислительной точки зрения для многих задач компьютерного зрения.
2. **Гистограммная эквализация:** Эта техника корректирует распределение интенсивности изображения для улучшения его контрастности, что облегчает алгоритмам обнаружение особенностей и паттернов.
3. **Гауссово размытие:** Гауссовское размытие - это метод подавления шума, который сглаживает изображение с помощью гауссовского ядра. Она позволяет эффективно уменьшить шум, сохраняя при этом детали изображения.
4. **Изменение размера изображения:** Изменение размера изображений помогает алгоритмам компьютерного зрения обрабатывать их эффективно и точно. Это особенно важно для моделей глубокого обучения, которые часто требуют фиксированных размеров входных данных.
5. **Вращение и масштабирование изображений:** Предварительная обработка изображений может включать вращение и масштабирование изображений для обеспечения согласованной ориентации и масштаба, что позволяет лучше выделять и сравнивать признаки.
6. **Обнаружение краев:** Методы определения краев, такие как операторы Канни или Собеля, могут использоваться для идентификации и подчеркивания границ объектов на изображении. Это может улучшить работу алгоритмов распознавания и сегментации объектов.
7. **Увеличение данных:** Увеличение данных подразумевает создание новых обучающих примеров путем применения различных преобразований к существующим изображениям, таких как вращение, переворачивание и сдвиг. Это может помочь повысить производительность и надежность моделей машинного обучения за счет предоставления более разнообразных обучающих данных.

Предварительная обработка изображений является важным этапом в разработке точных и эффективных приложений компьютерного зрения. Применяя различные методы, такие как подавление шума, улучшение изображения и извлечение признаков, мы можем улучшить качество и согласованность данных изображения, что в конечном итоге приведет к повышению производительности при решении последующих задач. Поскольку компьютерное зрение продолжает развиваться и находить новые применения, важность эффективной предварительной обработки изображений будет только расти, что делает ее необходимым навыком для исследователей и практиков в этой области.

### 4.1.2 Определение объектов

Обнаружение объектов - это фундаментальная задача компьютерного зрения, целью которой является идентификация и локализация интересующих объектов на изображениях или видео. Она широко используется в различных приложениях, включая автономные транспортные средства, робототехнику, видеонаблюдение и понимание изображений. В последние годы был достигнут значительный прогресс в разработке алгоритмов обнаружения объектов, что привело к повышению точности и производительности в реальном времени. Далее будут представлены основные методы и цели, связанны с алгоритмами обнаружения объектов, и их применение в компьютерном зрении.

Цели и характеристики алгоритмов обнаружения объектов:

1. **Локализация:** Одной из основных целей алгоритмов обнаружения объектов является точная локализация объектов на изображении. Локализация включает в себя определение пространственных координат ограничивающей рамки, которая плотно охватывает каждый обнаруженный объект.
2. **Классификация:** Наряду с локализацией, алгоритмы обнаружения объектов направлены на точную классификацию обнаруженных объектов. Классификация присваивает каждому объекту метку, указывающую его категорию или тип, например, "автомобиль", "человек" или "собака".
3. **Надежность:** Алгоритмы обнаружения объектов должны быть устойчивы к изменениям внешнего вида объекта, включая изменения масштаба, поворота, освещенности и окклюзии. Надежный алгоритм может точно обнаруживать и классифицировать объекты в различных условиях и сохранять высокую производительность в реальных приложениях.
4. **Производительность в реальном времени:** Многие приложения компьютерного зрения, такие как автономные транспортные средства и робототехника, требуют, чтобы алгоритмы обнаружения объектов работали в режиме реального времени. Способность быстро и эффективно обрабатывать изображения или видеокадры имеет решающее значение для обеспечения своевременной реакции системы.
5. **Масштабируемость:** Алгоритмы обнаружения объектов должны быть масштабируемыми, чтобы обрабатывать большие массивы данных и множество классов объектов. Масштабируемость важна для обучения и развертывания моделей обнаружения объектов в различных приложениях с разными требованиями.

Основные методы обнаружения объектов:

1. Традиционные методы: Ранние методы обнаружения объектов опирались на признаки, созданные вручную, и подходы со скользящим окном. Эти методы, такие как алгоритм Виолы-Джонса и гистограмма ориентированных градиентов (HOG), были успешны в обнаружении объектов в определенных сценариях, но им не хватало надежности и масштабируемости, необходимых для сложных реальных приложений.
2. Методы, основанные на глубоком обучении: С появлением глубокого обучения, конволюционные нейронные сети (КНС) значительно улучшили производительность алгоритмов обнаружения объектов. CNN могут изучать иерархические характеристики из необработанных данных изображения, что приводит к созданию высокодискриминативных и надежных представлений объектов.
   1. Двухэтапные детекторы: Двухэтапные детекторы, такие как R-CNN, Fast R-CNN и Faster R-CNN, сначала генерируют предложения регионов, которые потенциально могут содержать объекты, а затем классифицируют и уточняют координаты ограничивающей рамки для каждого предложения. Эти методы достигают высокой точности, но могут быть вычислительно дорогими.
   2. Одноэтапные детекторы: Одноэтапные детекторы, такие как YOLO (You Only Look Once) и SSD (Single Shot MultiBox Detector), выполняют обнаружение объектов за один проход по сети, устраняя необходимость в генерации предложений регионов. Эти методы быстрее и больше подходят для применения в реальном времени, но они могут быть менее точными, чем двухэтапные детекторы, особенно для маленьких объектов или объектов с большими изменениями соотношения сторон.
3. Одноэтапные детекторы с уточнением якоря: Последние достижения в области обнаружения объектов включают одноэтапные детекторы с уточнением якоря, такие как RetinaNet и EfficientDet. Эти методы сочетают скорость одноэтапных детекторов с точностью двухэтапных детекторов за счет использования методов уточнения якоря и функций потерь, которые устраняют дисбаланс между классами переднего и заднего плана.

Применение алгоритмов обнаружения объектов:

1. Автономные транспортные средства: Алгоритмы обнаружения объектов имеют решающее значение для того, чтобы автономные транспортные средства могли воспринимать и понимать окружающую обстановку. Они помогают распознавать другие транспортные средства, пешеходов, дорожные знаки и препятствия, обеспечивая безопасную и эффективную навигацию на дороге.
2. Робототехника: В робототехнике алгоритмы обнаружения объектов облегчают выполнение таких задач, как распознавание объектов, манипулирование ими и навигация. Эти алгоритмы позволяют роботам эффективно взаимодействовать с окружающей средой, выполняя такие задачи, как захват предметов, сортировка предметов и обход препятствий.
3. Здравоохранение: Алгоритмы обнаружения объектов все чаще используются в здравоохранении, например, при анализе медицинских изображений, где они могут идентифицировать и локализовать конкретные анатомические структуры, повреждения или аномалии на медицинских изображениях, помогая в диагностике и планировании лечения.
4. Розничная торговля: Алгоритмы обнаружения объектов могут применяться в розничной торговле для анализа поведения покупателей, контроля уровня запасов и выявления случаев воровства. Они также могут использоваться в автоматизированных системах касс, где алгоритмы идентифицируют и подсчитывают товары, приобретенные покупателями.
5. Сельское хозяйство: В сельском хозяйстве алгоритмы обнаружения объектов могут использоваться для идентификации и анализа сельскохозяйственных культур, оценки их здоровья и обнаружения вредителей или болезней. Эта информация может помочь фермерам оптимизировать стратегии управления сельскохозяйственными культурами, что приведет к повышению урожайности и снижению потерь.
6. Производство: Алгоритмы обнаружения объектов могут применяться в производственных процессах для проверки продукции на наличие дефектов или обеспечения правильной сборки. Они также могут использоваться для управления роботизированными системами при отборе и размещении изделий на производственных линиях.
7. Беспилотники: Алгоритмы обнаружения объектов используются в беспилотниках для различных целей, таких как наблюдение за дикой природой, осмотр инфраструктуры или воздушное наблюдение в районах, пострадавших от стихийных бедствий. Они помогают беспилотникам распознавать и отслеживать интересующие их объекты, избегая при этом препятствий.

### 4.1.3 YOLO модель

YOLO ("You Only Look Once") - это революционная модель обнаружения объектов, представленная Джозефом Редмоном и др. в их работе 2016 года. Модель YOLO предлагает значительный отход от традиционных методов обнаружения объектов, выполняя задачи обнаружения и классификации объектов за один проход по сети. Этот уникальный подход позволяет работать в режиме реального времени, сохраняя при этом высокий уровень точности, что делает YOLO популярным выбором для различных приложений компьютерного зрения.

Модель YOLO представляет собой конволюционную нейронную сеть (CNN), которая получает входное изображение и делит его на сетку ячеек. Каждая ячейка отвечает за предсказание фиксированного количества ограничивающих рамок и вероятностей классов. Затем модель объединяет эти прогнозы для получения окончательных результатов обнаружения объектов.

Архитектура модели YOLO состоит из нескольких конволюционных слоев, за которыми следуют полностью связанные слои, выдающие окончательные прогнозы. Функция потерь, используемая для обучения модели YOLO, сочетает потери на локализацию, классификацию и уверенность, что позволяет модели научиться предсказывать точные граничные рамки и вероятности классов.

Ключевые особенности модели YOLO:

1. Производительность в реальном времени: Однопроходная архитектура модели YOLO позволяет значительно ускорить обнаружение объектов по сравнению с традиционными двухэтапными детекторами, такими как R-CNN и его разновидности. Такая производительность в реальном времени очень важна для приложений, требующих своевременного реагирования, таких как автономные транспортные средства и робототехника.
2. Комплексное обучение: В отличие от многих других моделей обнаружения объектов, которые полагаются на многоступенчатый конвейер, YOLO обучается из конца в конец. Это делает процесс обучения более простым и эффективным, поскольку нет необходимости в точной настройке отдельных компонентов модели.
3. Глобальный контекст: Модель YOLO использует все изображение во время обучения и прогнозирования, что позволяет ей улавливать глобальный контекст и делать более точные прогнозы. Такой глобальный взгляд помогает YOLO обнаруживать объекты, которые могут быть частично закрыты или иметь схожий с окружающим вид.
4. Обобщение: Было показано, что YOLO хорошо обобщает новые классы объектов и сцены. Это делает ее универсальной моделью, которая может быть адаптирована к различным задачам компьютерного зрения с минимальными изменениями.

YOLOv5 - это неофициальная версия YOLO, разработанная командой Ultralytics. Она включает в себя многочисленные усовершенствования и оптимизации, такие как обрезка модели и квантование, что позволяет уменьшить размер модели и ускорить время вывода. YOLOv5 завоевала популярность благодаря простоте использования и отличной производительности в реальных приложениях.

### 4.1.4 Классификация изображений

Классификация изображений - это фундаментальная задача в области компьютерного зрения и машинного обучения, целью которой является присвоение предопределенных категорий или меток изображениям на основе их визуального содержания. Основная цель классификации изображений - научить компьютер воспринимать и интерпретировать изображения подобно тому, как это делает человек, что позволит им принимать интеллектуальные решения и выполнять задачи на основе информации, извлеченной из изображений. Классификация изображений имеет множество применений, включая распознавание объектов, понимание сцен, анализ медицинских изображений, распознавание лиц и автономную навигацию.

До появления методов глубокого обучения классификация изображений в значительной степени зависела от методов извлечения признаков вручную и алгоритмов машинного обучения. Ручное извлечение признаков подразумевает разработку алгоритмов для идентификации и извлечения отличительных признаков из изображений, таких как края, углы, текстуры и цвета. Затем эти признаки используются в качестве входных данных для классификаторов машинного обучения, таких как опорные векторные машины (SVM), k-Nearest Neighbors (k-NN) или Random Forests, для классификации изображений.

Некоторые широко используемые методы ручного извлечения признаков включают масштабно-инвариантное преобразование признаков (SIFT), гистограмму ориентированных градиентов (HOG) и локальные бинарные паттерны (LBP). Хотя эти традиционные методы могут достичь приемлемой производительности в определенных задачах классификации изображений, они часто не могут хорошо обобщать более сложные задачи и крупномасштабные наборы данных.

Внедрение методов глубокого обучения, в частности конволюционных нейронных сетей (КНС), произвело революцию в области классификации изображений, значительно повысив точность и эффективность моделей. CNN - это класс моделей глубокого обучения, специально разработанных для обработки данных, похожих на сетку, таких как изображения. Они состоят из нескольких слоев, включая конволюционные слои, слои объединения и полностью связанные слои, которые учатся автоматически извлекать иерархические признаки из входных изображений и использовать их для классификации.

Ключевое преимущество CNN перед традиционными методами заключается в их способности извлекать признаки непосредственно из необработанных значений пикселей, устраняя необходимость в ручном извлечении признаков. Этот сквозной процесс обучения позволяет CNN улавливать более сложные паттерны и представления, что приводит к превосходной производительности в различных задачах классификации изображений.

С момента появления CNN было предложено и разработано несколько архитектур, причем каждая итерация улучшала предыдущие с точки зрения точности и эффективности. К наиболее популярным и влиятельным архитектурам CNN относятся:

1. LeNet-5: Разработанный Янном Лекуном и его командой в конце 1990-х годов, LeNet-5 считается пионером современных CNN. Она использовалась в основном для распознавания цифр в рукописных почтовых индексах и продемонстрировала потенциал CNN в задачах классификации изображений.
2. AlexNet: В 2012 году Алекс Крижевский, Илья Суцкевер и Джеффри Хинтон предложили архитектуру AlexNet, которая показала революционные результаты в конкурсе ImageNet Large Scale Visual Recognition Challenge (ILSVRC). AlexNet значительно превзошла традиционные методы и проложила путь для дальнейшего развития архитектур CNN.
3. VGG: Архитектура VGG, разработанная группой Visual Geometry Group в Оксфордском университете, исследовала идею использования более глубоких сетей с меньшими конволюционными фильтрами. VGG продемонстрировала, что увеличение глубины сети может привести к улучшению производительности в задачах классификации изображений. VGG-16 и VGG-19, по количеству слоев в сети, являются двумя популярными вариантами этой архитектуры.
4. Inception (GoogLeNet): Архитектура Inception, также известная как GoogLeNet, была предложена исследователями из Google. В ней была введена концепция модулей Inception, которая позволила более эффективно использовать вычислительные ресурсы за счет использования параллельных сверток с различными размерами фильтров. Эта архитектура достигла самой высокой производительности на конкурсе ILSVRC 2014.
5. ResNet: Остаточные сети (ResNet) были предложены исследователями из Microsoft Research и ввели понятие остаточных связей или пропущенных связей. Эти связи позволили обучать гораздо более глубокие сети за счет облегчения проблемы исчезающего градиента. Архитектуры ResNet, такие как ResNet-50, ResNet-101 и ResNet-152, достигли высочайшей производительности в различных задачах классификации изображений и соревнованиях.
6. DenseNet: Плотные конволюционные сети (DenseNet) были предложены в качестве расширения ResNet, где каждый слой соединен со всеми последующими слоями в плотном блоке. Такая архитектура позволяет более эффективно использовать параметры сети и улучшить градиентный поток во время обучения.
7. MobileNet: MobileNets - это семейство облегченных архитектур CNN, разработанных для эффективного развертывания на мобильных устройствах и встроенных системах. В них используются глубинные сепарабельные свертки, которые значительно сокращают количество параметров и вычислительные затраты без снижения производительности.
8. EfficientNet: EfficientNet - это семейство архитектур CNN, которые масштабируются комплексно, одновременно учитывая глубину, ширину и разрешение. Архитектура EfficientNet была получена с помощью поиска нейронной архитектуры (NAS) и достигла самой высокой производительности на различных эталонах классификации изображений с меньшим количеством параметров и меньшими вычислительными затратами по сравнению с другими популярными архитектурами.

## 4.2 Описание датасетов

### 4.2.1 Датасет для определения яблок

Для тренировки модели определения яблок использовался набор данных “MinneApple: A Benchmark Data-set for Apple Detection and Segmentation”. Новый набор данных для продвижения передового опыта в области обнаружения, сегментации и подсчета фруктов в садах. Хотя в последнее время наблюдается значительный интерес к решению этих проблем, отсутствие единого набора данных затрудняет сравнение результатов.

Этот набор данных позволяет проводить прямые сравнения, предоставляя большое разнообразие изображений высокого разрешения, полученных в фруктовых садах. изображений, полученных в садах, вместе с аннотациями фруктов на деревьях, сделанными человеком. Фрукты маркируются с помощью полигональных масок для каждого объекта, что помогает в точном обнаружения, локализации и сегментации объектов. Кроме того, набор данных предназначен для подсчета кластеризованных фруктов.

Для построения модели из общего датасета был выделен набор яблок:

1. 563 фото для тренировки модели (80%)
2. 150 фото для валидации модели (20%)

***Рисунок 14. Структура папок датасета для определения яблок***

Text

Description automatically generated

***Таблица 3. Запись данных о фото в стандарте воспринимаемым YOLO***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| class\_id | x\_center\_norm | y\_center\_norm | width\_norm | height\_norm |
| 0 | 0.47 | 0.44 | 0.26 | 0.37 |
| 0 | 0.72 | 0.72 | 0.23 | 0.39 |
| 0 | 0.80 | 0.26 | 0.23 | 0.40 |

* class\_id - Целое число, представляющее класс объекта (например, 0 для apple, 1 для damaged\_apple).
* x\_center\_norm - Нормированная x-координата центра ограничивающей рамки относительно ширины изображения.
* y\_center\_norm - Нормированная координата y центра ограничивающей рамки относительно высоты изображения.
* width\_norm - Нормированная ширина ограничивающей рамки относительно ширины изображения.
* height\_norm - Нормированная высота ограничивающей рамки относительно высоты изображения.

***Рисунок 15. Примеры фото из датасета***

 A picture containing fruit, variety

Description automatically generated

### 4.2.2 Датасет для определения болезней яблок

Для определения болезней яблок использовался датасет из 502 фото. В датасете были собраны фото как яблок в полях и садах, так и яблок в производственных условиях, где качество и точность съемки значительно отличается. Фото яблок в датасете были распределены по 4 категориям яблок:

* Чернь – 146 фото
* Плодовая гниль – 148 фото
* Парша – 111 фото
* Нормальное яблоко – 97 фото

Для целей построения, валидации и тестирования датасета был разделен в следующих пропорциях: обучение – 80%, валидация– 10%, тестирование – 10%.

***Рисунок 16. Структура папок датасета для определения яблок***

Text

Description automatically generated

***Рисунок 17. Примеры фото датасета для классификации болезней***

 A fruit from a tree

Description automatically generated with low confidence

A fruit on a tree

Description automatically generated with low confidence 

## 4.3 Разработка модели

### 4.3.1 Определение яблок на фото

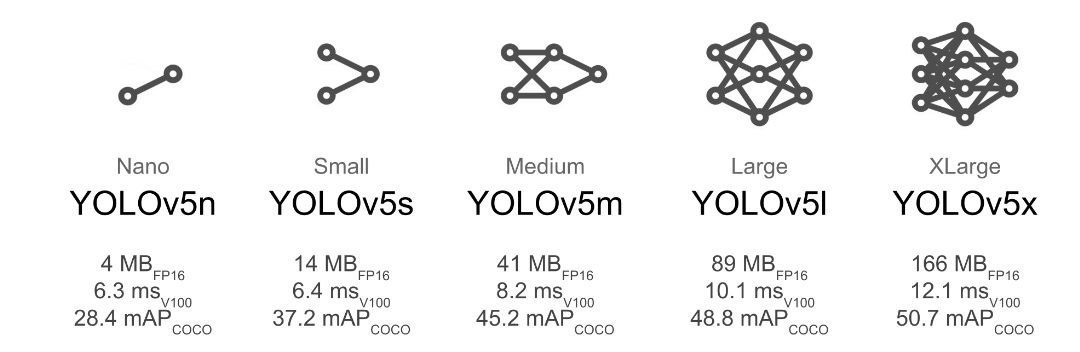
Для задачи определения яблока на фото была выбрана модель YOLOv5, так как она обладает такими преимуществами, как:

* Производительность в реальном времени: YOLO разработана для быстрой обработки изображений, что делает ее подходящим для приложений реального времени.
* Одновременное обнаружение: YOLO использует другой подход по сравнению с традиционными методами обнаружения объектов. Вместо того, чтобы делить изображение на регионы и выполнять классификацию и локализацию отдельно, YOLO предсказывает вероятности классов и координаты ограничивающей области напрямую. Это позволяет YOLO обнаруживать несколько объектов за один проход.
* Одноэтапная архитектура: YOLO имеет одноступенчатую архитектуру. Такая простота приводит к созданию более эффективной и оптимизированной модели, снижая вычислительные затраты и требования к памяти.
* Open Source модель, что означает, что ее можно без ограничений использовать как в исследовательских, так и в коммерческих целях.

Перечисленные выше преимущества позволять сравнительно быстро обучить модель, а также эффективно применять ее в реальном мире.

Для обучения модели использовался датасет описанный в главе 4.2.1. После подготовки датасета была использована предобученная модель YOLOv5s, так как предполагется, что модель будет использоваться в полевых или производственных условиях и необходимо сохранить скорость работы алгоритма:

***Рисунок 18. Виды моделей YOLOv5***



В качестве архитектуры, YOLOv5 использует сверточную нейронную сеть (CNN) для извлечения особенностей из входного изображения. Архитектура разработана таким образом, что она постепенно уменьшает пространственные размеры, одновременно увеличивая глубину карт признаков.

YOLOv5 предсказывает координаты ограничительных рамок, используя предопределенные формы, называемые якорными рамками. Это набор размеров, которые модель использует в качестве отправной точки для предсказания фактических ограничительных рамок. Модель обучается предсказывать смещение от этих якорных ящиков вместо предсказания абсолютных координат, что делает процесс обучения более стабильным.

Модель обучается минимизировать функцию потерь, которая состоит из трех частей:

* потери на локализацию - неточность в координатах ограничительных ячеек;
* потери на объектность - ошибки в предсказании того, содержит ли определенная ячейка сетки объект или нет;
* потери на классификацию - ошибки в предсказании класса.

***Рисунок 19. Анализ датасета сгенерированный в ходе обучения модели YOLOv5***

A picture containing text, screenshot, diagram, colorfulness

Description automatically generated

Для обучения модели были использованы следующие параметры: epoch=50, batch\_size=16. Также, в качестве оптимизатора модели был выбран Stochastic Gradient Descent (SGD) - итерационный метод оптимизации. В отличие от стандартного градиентного спуска, SGD аппроксимирует истинный градиент функции потерь, беря градиент функции потерь, рассчитанный по одной случайно выбранной точке данных. В результате SGD часто сходится намного быстрее на практике, а также может обучаться на наборах данных, которые больше, чем помещается в памяти, поскольку ему нужно загружать только одну точку данных за раз.

***Рисунок 20. Процесс обучения модели***

A screen shot of a computer

Description automatically generated with low confidence

Результаты обучения модели приведены ниже. Точность определения составляет более 80%, что удовлетворяет потребностям алгоритма.

***Рисунок 21. Результаты модели по определению объекта яблока***

A picture containing text, diagram, line, plot

Description automatically generated

***Рисунок 22. Визуализация модели по определению объекта яблока***

A collage of images of apples

Description automatically generated with low confidence

Результатом тренировки модели стало получение весов обученной модели, для определения объекта яблока по фото. Определенные яблоки использовались в классификационной модели для определения болезней.

### 4.3.2 Классификация болезней

Классификация болезней яблок была реализована с использованием предобученной модели Tensorflow.Keras MobileNetV2, использующей веса imagenet.

MobileNetV2 - это архитектура сверточной нейронной сети, разработанная исследователями Google для мобильных и встроенных приложений компьютероного зрения. Она является усовершенствованием оригинальной модели MobileNet.

MobileNetV2 оптимизирована для мобильных устройств, так как использует эффективную архитектуру, которая снижает вычислительные требования без существенной потери точности, что делает ее подходящей для устройств с ограниченными вычислительными ресурсами и соответствует необходимости быстрой обработки.

Для обучения модели использовался датасет описанный в пункте 4.3.2. Загруженный в модель при помощи класса ImageDataGenerator библиотеки Keras. Данный класс был также использован для обогащения набор данных при помощи техники data augmentation.

***Рисунок 23. Параметры ImageDataGenerator***

A picture containing text, screenshot, font

Description automatically generated

Для обучения модели было использованны следующие параметры: epoch=25, batch\_size=32. Для валидации был использован параметр epoch=10 (приложение 1):

***Рисунок 24. Обучение модели по классификации болезней***

Text

Description automatically generated

Полученная модель обладает 72% точностью. В качестве, результата тренировки модели стало получение весов для применения классификации.

Для объединения результатов моделей был написан скрипт, которые принимает на вход путь к папке с фото и в качестве результата выводит данные об определенных объектах, а также их качестве (приложение 2):

***Рисунок 25. Пример выделения объекта яблока моделью***

A picture containing fruit, tree, fruit tree, apple

Description automatically generated

***Рисунок 26. Пример определения болезней моделью***

A picture containing fruit, tree, fruit tree, outdoor

Description automatically generated

# 5. Заключение

В заключение следует отметить, что данная работа успешно продемонстрировал применение моделей глубокого обучения для обнаружения и классификации болезней яблок. Используя современную модель обнаружения объектов YOLOv5, было точно определено местоположение яблок на изображениях. Это стало важным начальным шагом построения модели, позволившим выделить интересующие нас области для дальнейшего анализа.

Затем для классификации состояния здоровья обнаруженных яблок была использована предварительно обученная модель MobileNetV2. С помощью дообучения модели и использования техники data augmentation модель MobileNetV2 была настроена под классификацию различных заболеваний яблок.

Интеграция этих двух моделей позволила создать эффективный алгоритм для обнаружения болезней яблок. Данная работа не только демонстрирует потенциал искусственного интеллекта в сельскохозяйственной отрасли, но и закладывает основу для будущих исследований и приложений. Разработанная модель может существенно помочь фермерам и сельскохозяйственным специалистам в мониторинге состояния здоровья их культур, позволяя обнаружить болезни на ранней стадии и тем самым потенциально предотвратить значительные потери урожая.

Однако необходимо отметить, что производительность этой системы в значительной степени зависит от качества и разнообразия обучающих данных. Поэтому постоянный сбор и аннотирование данных, отражающих различные виды яблок, стадии заболевания и условия окружающей среды, позволит еще больше повысить эффективность модели и ее способность к обобщению на непредвиденные сценарии.

Дальнейшие шаги могут быть применены к модели для повышения ее точности:

* Сбор большего количества данных - большее количество примеров позволит модели обучиться точнее. Собранные данные должны охватывать как можно больше различных сценариев, ракурсов, условий освещения, типов яблок и болезней.
* Использование нескольких моделей - часто можно получить лучшие результаты, используя несколько моделей. Это может включать в себя обучение нескольких моделей и усреднение их прогнозов.
* Настройка гиперпараметров – проведение экспериментов с различными скоростями обучения, размерами партий, оптимизаторами и так далее.
* Балансировка классов: - если некоторые классы имеют гораздо больше примеров, чем другие, это может привести к тому, что модель будет плохо работать с недостаточно представленными классами. Необходимо рассмотреть возможность использования таких методов, как перевыборка класса меньшинств, недовыборка класса большинства или использование комбинации обоих методов.
* Анализ ошибок: необходимо проанализировать примеры, в которых ваша модель ошибается и определить наличие закономерностей. Это может дать представление о том, что происходит не так и как это можно исправить.
* Использование моделей сегментации - вместо обнаружения яблоке и их классификации, можно использовать модель сегментации для определения точных пикселей яблока, которые поражены болезнью.

Данная модель имеет 2 основных направления применения:

* Использование в садах: Модель может помочь выявить болезни на ранних стадиях, что потенциально может спасти значительную часть урожая, которая могла бы быть потеряна, если бы болезнь развивалась незамеченной. Раннее лечение может предотвратить распространение болезней, что приведет к оздоровлению посевов и повышению урожайности.

Традиционные методы обнаружения болезней часто предполагают ручной осмотр посевов, что может отнимать много времени и сил. Aвтоматизированная система может быстро и точно сканировать большое количество изображений яблок, экономя фермерам значительное время и усилия.

* Использование ритейлерами: Ритейлеры могут использовать эту модель для регулярного контроля качества яблок, которые они получают от поставщиков. Выявляя больные или некачественные яблоки до того, как они попадут на полки магазинов, розничные торговцы могут гарантировать, что покупателям будут продаваться только яблоки самого высокого качества.

Выявляя некачественные или больные яблоки на ранней стадии, ритейлеры могут сократить количество отходов, вызванных яблоками, которые не подлежат продаже или портят другие яблоки во время хранения.

Розничные торговцы могут использовать информацию, полученную с помощью модели, для более эффективного управления запасами яблок. Например, если выявлено, что партия яблок подвержена риску определенного заболевания, они могут решить продать эти яблоки со скидкой, чтобы быстрее их реализовать.

# Литература

1. Аббасов Ифтихар Балакишиевич, Дешмух Ратнадип Р РАСПОЗНАВАНИЕ ИЗОБРАЖЕНИЙ СЕЛЬСКОХОЗЯЙСТВЕННЫХ КУЛЬТУР, РАСТЕНИЙ И ЛЕСНЫХ МАССИВОВ // Известия ЮФУ. Технические науки. 2020. № 3.
2. О.Н. Ивашова, Н.В. Гавриловская, Е.В. Щедрина ВНЕДРЕНИЕ ЦИФРОВЫХ ТЕХНОЛОГИЙ ДЛЯ ОБЕСПЕЧЕНИЯ РАЗВИТИЯ СЕЛЬСКОХОЗЯЙСТВЕННОЙ ОТРАСЛИ // Международный журнал гуманитарных и естественных наук. 2022. №3-2.
3. Рыбаков Алексей Владимирович, Рыбаков Илья Александрович, Арыкбаев Равиль Каримович ОЦЕНКА ЭКОНОМИЧЕСКОЙ ЭФФЕКТИВНОСТИ ИНВЕСТИЦИОННОГО ПРОЕКТА «АГРОРОБОТ» // Каспийский регион: политика, экономика, культура. 2022. №4 (73).
4. Рыбаков Алексей Владимирович, Выборнов Николай Анатольевич, Рыбаков Илья Александрович АНАЛИЗ МЕТОДОВ КОМПЬЮТЕРНОГО ЗРЕНИЯ, ПЕРСПЕКТИВНЫХ ДЛЯ ПРИМЕНЕНИЯ В АГРОПРОМЫШЛЕННОМ КОМПЛЕКСЕ // Прикаспийский журнал: управление и высокие технологии. 2022. №1 (57).
5. Терещенко Сергей Николаевич, Перов Артём Андреевич, Осипов Александр Леонидович КЛАССИФИКАЦИЯ ВНЕКОРНЕВЫХ ЗАБОЛЕВАНИЙ ЯБЛОНЕВЫХ КУЛЬТУР МЕТОДАМИ КОМПЬЮТЕРНОГО ЗРЕНИЯ // Siberian Journal of Life Sciences and Agriculture. 2021. №3.
6. Gao, Ruilong & Zhou, Qiaojun & Cao, Songxiao & Jiang, Qing. (2022). An Algorithm for Calculating Apple Picking Direction Based on 3D Vision. Agriculture. 12. 1170. 10.3390/agriculture12081170.
7. Rehman, Samra & Khan, Muhammad & Alhaisoni, Majed & Armghan, Ammar & Alenezi, Fayadh & Alqahtani, Abdullah & Vesal, Khean & Nam, Yunyoung. (2023). Fruit Leaf Diseases Classification: A Hierarchical Deep Learning Framework. Computers, Materials and Continua. 75. 1179-1194. 10.32604/cmc.2023.035324.
8. Alagu S. (2020). Apple Fruit disease detection using Multiclass SVM classifier and IP Webcam APP.
9. Zhang, Liangji & Zhou, Guoxiong & Chen, Aibin & Yu, Wentao & Peng, Ning & Chen, Xiao. (2023). Rapid computer vision detection of apple diseases based on AMCFNet. Multimedia Tools and Applications. 1-21. 10.1007/s11042-023-15548-x.
10. Sean, Y. & Smith, D.D. & Bitra, Venkata & Beera, Vimala & Umar, Sk. (2021). Development of Computer Vision System for Fruits. Current Journal of Applied Science and Technology. 1-11. 10.9734/cjast/2021/v40i3631576.
11. Pothula, Anand & Zhang, Zhao & Lu, Renfu. (2023). Evaluation of a new apple in-field sorting system for fruit singulation, rotation and imaging. 208. 10.1016/j.compag.2023.107789.
12. Li, GuoYan & Yang, Yi. (2023). Smart vision for quality apple classification using SURF–Harris optimizing techniques. Wireless Networks. 1-13. 10.1007/s11276-023-03323-7.
13. Bhargava, Anuja & Bansal, Atul. (2021). Classification and Grading of Multiple Varieties of Apple Fruit. Multimedia Tools and Applications. 14. 10.1007/s12161-021-01970-0.
14. Buyukarikan, Birkan & Ulker, Erkan. (2023). Classification of physiological disorders in apples using deep convolutional neural network under different lighting conditions. Multimedia Tools and Applications. 1-21. 10.1007/s11042-023-14766-7.
15. Gené-Mola, Jordi & Ferrer-Ferrer, Mar & Gregorio Lopez, Eduard & Blok, Pieter & Hemming, Jochen & Morros Rubió, Josep Ramon & Rosell-Polo, Joan & Vilaplana, Verónica & Ruiz-Hidalgo, Javier. (2023). Looking behind occlusions: A study on amodal segmentation for robust on-tree apple fruit size estimation. Computers and Electronics in Agriculture. 209. 107854. 10.1016/j.compag.2023.107854.
16. Gongal, A. & Karkee, Manoj & Amatya, Suraj. (2018). Apple fruit size estimation using a 3D machine vision system. Information Processing in Agriculture. 5. 498-503. 10.1016/j.inpa.2018.06.002.
17. Baneh, Nesar & Navid, Hossein & Kafashan, Jalal & Fouladi, Hatef & Gonzales-Barrón, Ursula. (2023). Development and Evaluation of a Small-Scale Apple Sorting Machine Equipped with a Smart Vision System. AgriEngineering. 5. 473-487. 10.3390/agriengineering5010031.

# Приложения

## Приложение 1

import tensorflow as tf

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.applications import MobileNetV2

from tensorflow.keras.layers import Dense, GlobalAveragePooling2D, Input

from tensorflow.keras.models import Model

from tensorflow.keras.optimizers import Adam

from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint

from tensorflow.keras.layers import Dropout

tf.config.threading.set\_inter\_op\_parallelism\_threads(1)

tf.config.threading.set\_intra\_op\_parallelism\_threads(1)

# Load and preprocess data

print("Loading and preprocessing data...")

train\_datagen = ImageDataGenerator(

rescale=1./255,

rotation\_range=20,

width\_shift\_range=0.2,

height\_shift\_range=0.2,

shear\_range=0.2,

zoom\_range=0.2,

horizontal\_flip=True,

fill\_mode='nearest'

)

train\_generator = train\_datagen.flow\_from\_directory('apple\_dataset\\train', target\_size=(224, 224), batch\_size=32, class\_mode='categorical', subset='training')

validation\_generator = train\_datagen.flow\_from\_directory('apple\_dataset\\validation', target\_size=(224, 224), batch\_size=32, class\_mode='categorical', subset='validation')

# Load pre-trained MobileNetV2 and customize the classifier

print("Loading pre-trained MobileNetV2 and customizing the classifier...")

base\_model = MobileNetV2(input\_shape=(224, 224, 3), include\_top=False, weights='imagenet')

x = base\_model.output

x = GlobalAveragePooling2D()(x)

x = Dense(1024, activation='relu')(x)

predictions = Dense(4, activation='softmax')(x) # Assuming 3 disease types + 1 healthy class

dropout\_layer = Dropout(0.5)

model = Model(inputs=base\_model.input, outputs=predictions)

# Freeze base model layers

for layer in base\_model.layers:

layer.trainable = False

# Add early stopping

early\_stopping = EarlyStopping(monitor='val\_loss', patience=5)

model\_checkpoint = ModelCheckpoint('best\_model.h5', monitor='val\_loss', save\_best\_only=True)

callbacks = [early\_stopping, model\_checkpoint]

# Compile and train the model

print("Compiling and training the model...")

model.compile(optimizer=Adam(learning\_rate=0.0001), loss='categorical\_crossentropy', metrics=['accuracy'])

model.fit(train\_generator, epochs=25, validation\_data=validation\_generator)

# Unfreeze some layers of the base model for fine-tuning

print("Unfreezing some layers of the base model for fine-tuning...")

for layer in model.layers[:100]:

layer.trainable = False

for layer in model.layers[100:]:

layer.trainable = True

# Compile the model with a lower learning rate for fine-tuning

print("Compiling the model with a lower learning rate for fine-tuning...")

model.compile(optimizer=Adam(learning\_rate=0.00001), loss='categorical\_crossentropy', metrics=['accuracy'])

# Train the model with fine-tuning

print("Training the model with fine-tuning...")

model.fit(train\_generator, epochs=10, validation\_data=validation\_generator)

# Save the trained model

print("Saving the trained model...")

model.save('apple\_disease\_classifier.h5')

# Evaluate the model on test data (assuming you have a separate test set)

print("Evaluating the model on test data...")

test\_datagen = ImageDataGenerator(rescale=1./255)

test\_generator = test\_datagen.flow\_from\_directory('apple\_dataset\\test', target\_size=(224, 224), batch\_size=32, class\_mode='categorical')

test\_loss, test\_acc = model.evaluate(test\_generator)

print('Test accuracy:', test\_acc)

## Приложение 2

import cv2

import numpy as np

from tensorflow.keras.models import load\_model

# Load the trained model

model = load\_model('apple\_disease\_classifier.h5')

class\_names = ['Blotch', 'Normal', 'Rot', 'Scab']

# Load the image

img\_path = r"C:\Users\GDS\_Manager\Documents\yolov5-master\test\_images\post\_5d5fb10e60a47.jpg"

labels = r"C:\Users\GDS\_Manager\Documents\yolov5-master\runs\detect\exp3\labels\post\_5d5fb10e60a47.txt"

img = cv2.imread(img\_path)

height, width = img.shape[:2] # get the height and width of the image

# Load bounding boxes from the file

with open(labels, 'r') as f:

lines = f.readlines()

bounding\_boxes = []

yolo\_classes = []

for line in lines:

values = line.split()

class\_id = int(values[0])

center\_x = float(values[1]) \* width

center\_y = float(values[2]) \* height

box\_width = float(values[3]) \* width

box\_height = float(values[4]) \* height

x\_min = int(center\_x - box\_width / 2)

y\_min = int(center\_y - box\_height / 2)

x\_max = int(center\_x + box\_width / 2)

y\_max = int(center\_y + box\_height / 2)

bounding\_boxes.append((x\_min, y\_min, x\_max, y\_max))

yolo\_classes.append(class\_id)

# Continue with the rest of the script as before...

for box, yolo\_class in zip(bounding\_boxes, yolo\_classes):

x\_min, y\_min, x\_max, y\_max = box

apple = img[y\_min:y\_max, x\_min:x\_max]

# Resize the apple image to the input size of your classifier model

apple = cv2.resize(apple, (224, 224))

# Add an extra dimension to the image tensor for the batch size

apple\_batch = np.expand\_dims(apple, axis=0)

if yolo\_class != 0:

# Predict the class of the apple

predictions = model.predict(apple\_batch)

# Find the class label with the highest probability

predicted\_class = np.argmax(predictions)

# Print the predicted class

print(f'The predicted class for apple is: {class\_names[predicted\_class]}')

else:

predicted\_class = 1

# Draw the bounding box and the predicted class on the image

img = cv2.rectangle(img, (x\_min, y\_min), (x\_max, y\_max), (0, 255, 0), 2)

cv2.putText(img, class\_names[predicted\_class], (x\_min, y\_max+20), cv2.FONT\_HERSHEY\_SIMPLEX, 0.9, (0, 255, 0), 2)

# Display the image with bounding boxes and class names

cv2.imshow('Apples', img)

cv2.waitKey(0)

cv2.destroyAllWindows()