Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

направление подготовки: 09.03.04 Программная инженерия

**О Т Ч Е Т**

**по учебной практике**

Выполнил: студент группы РИС-23-1б

Гордеев Василий Андреевич

(подпись)

Проверил:

Старший преподаватель Елисеева Е.Н.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка) (подпись)

г. Пермь, 2025

РЕФЕРАТ

Отчет 38 с., 1 рис., 1 табл., 5 источн., 1 прил.

WSL, DOCKER, YOLO, PYTHON, PYTORCH, ИСКУСТВЕННЫЙ ИНТЕЛЛЕКТ, НЕЙРОСЕТИ, MACHINE LEARNING.

Целью работы является разработка и обучение модели машинного обучения для распознавания и детекции дефектов намотки оптоволокна на производстве. При обнаружении дефекта намотки программа подчеркивает в реальном времени с небольшой задержкой место дефекта прямоугольной рамочкой.

Задачи работы:

1. Изучить предметную область и решения существующие в открытых источниках.
2. Разработать архитектуру проекта, определить требования к ПО, и согласовать это с начальством
3. Разработать минимальную версию программы, удостоверится в работоспособности ПО, библиотек и кода.
4. Разработать демо-версию программы, презентовать полученные результаты точности обнаружения дефектов, доказав тем самым робастость данного подхода по сравнению с классическим компьютерным зрением, выдвинуть возможные перспективы дальнейшей разработки.

В результате был разработан пайплайн, и доказано преймущество данного подхода. Цели практики выполнены. Начальство предложило заняться созданием рабочей версии программы для внедрения ее в производство.

СОДЕРЖАНИЕ

[РЕФЕРАТ 2](#_Toc207105409)

[СОДЕРЖАНИЕ 3](#_Toc207105410)

[ВВЕДЕНИЕ 4](#_Toc207105411)

[Раздел 1. Анализ предметной области 5](#_Toc207105412)

[1.1 Определение границ исследования. 5](#_Toc207105413)

[1.2 Анализ существующих подходов к обнаружению объектов на изображениях. 10](#_Toc207105414)

[1.3 Требования к разрабатываемому проекту. 17](#_Toc207105415)

[Раздел 2. Проектирование проекта 18](#_Toc207105416)

[2.1 Разметка изображений 18](#_Toc207105417)

[2.2 Подготовка датасета к обучению 22](#_Toc207105418)

[2.3 Настройка модели обучения 25](#_Toc207105419)

[Раздел 3. Ход и результаты исследований 29](#_Toc207105420)

[3.1 Этапы исследовательской работы 29](#_Toc207105421)

[3.2 Результаты тестирования и экспертной оценки 29](#_Toc207105422)

[3.3 Перспективы дальнейшего развития 30](#_Toc207105423)

[ЗАКЛЮЧЕНИЕ 31](#_Toc207105424)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 33](#_Toc207105425)

ВВЕДЕНИЕ

В последнее время многие предприятия работают в сторону автоматизации производства. В связи с этим идет семимильными шагами прогресс в области ИИ. Те предприятия, которые осваивают потенциал новых технологий, как правило становятся лидерами в своей области. В дальнейшем они задают темп развития производств.

Целью данной работы является разработка и обучение модели машинного обучения для распознавания и детекции дефектов намотки оптоволокна на производстве. При обнаружении дефекта намотки программа подчеркивает в реальном времени с небольшой задержкой место дефекта прямоугольной рамочкой.

Задачи работы:

1. Изучить предметную область и решения существующие в открытых источниках.
2. Разработать архитектуру проекта, определить требования к ПО, и согласовать это с начальством
3. Разработать альфа-версию программы, удостоверится в работоспособности ПО, библиотек и кода.
4. Разработать демо-версию программы, презентовать полученные результаты точности обнаружения дефектов, доказав тем самым робастость данного подхода по сравнению с классическим компьютерным зрением, выдвинуть возможные перспективы дальнейшей разработки.

Практическая значимость работы заключается в приобретении навыков работы с нейросетями и существующими моделями ИИ в области компьютерного зрения, а также в получении навыков интеграции разработанного решения в производство.

## Раздел 1. Анализ предметной области

### 1.1 Определение границ исследования.

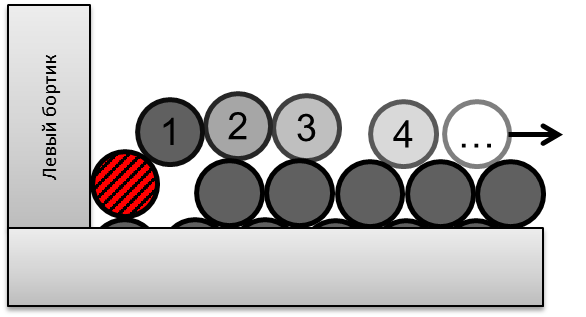
#### 1.1.1 Параметры входных данных

1. Изображение намотки оптоволокна подается с производственной камеры в одноканальном формате Gray8.
2. Камера работает на 40 fps
3. Разрешение камеры 1920х1200
4. Выбрано расширение для хранения изображения png, так как формат jpg сжимает данные с определенными потерями, а bmp достаточно много весит.

#### 1.1.2 Объекты детекции

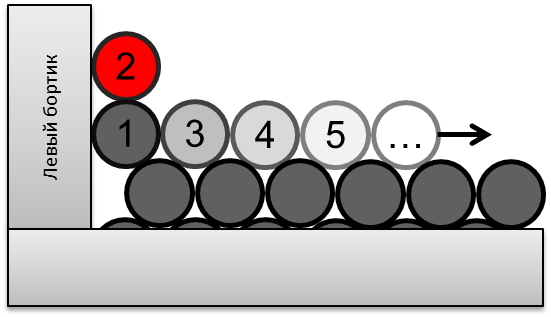
#### 1.1.2.1 Ошибки укладки волокна у бортика

1) Прилипание нижнего витка к бортику

Первый виток нового слоя ложится не к бортику, а рядом с ним, это является критичной ошибкой и намотку продолжать не следует, пока она не будет исправлена.

2) Прилипание второго витка к бортику

Второй виток в новом слое ложится к бортику над первым. Причиной может быть лишний компаунд, притягивающий волокно к бортику, или неплотно лежащий у бортика первый виток.



#### 1.1.2.2 Ошибки на прямом участке.

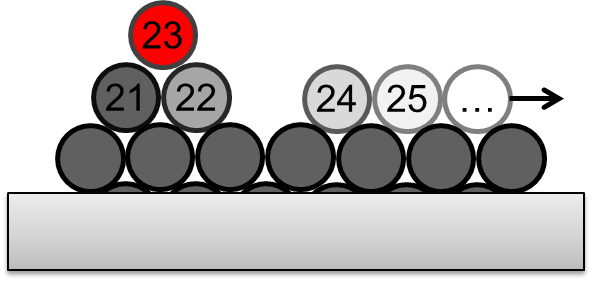
Самыми простыми являются ошибки на прямом участке. Благодаря использованию нулевого ряда они встречаются крайне редко. Основными можно выделить:

* Забегание витка вперед.
* Взлет волокна вверх относительно уже уложенных витков текущего слоя.
* Провал волокна вниз.

В большинстве случаев это связано с попаданием мусора в катушку.

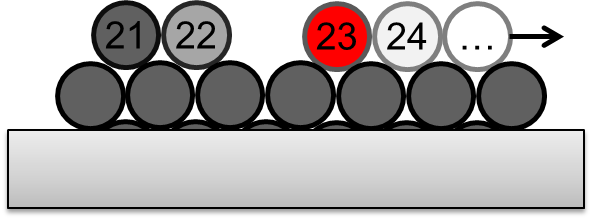
В очень редких случаях подобная ошибка возможна из-за дефектов волокна.

1. Запрыгивание витка наверх.



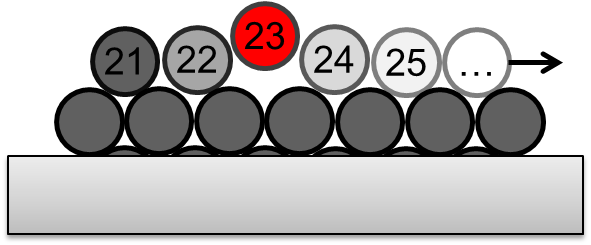
1. Расщепление раскладки.

Расщепление может быть на всем витке, или локально в конкретном месте. И тот и другой случай необходимо исправлять.



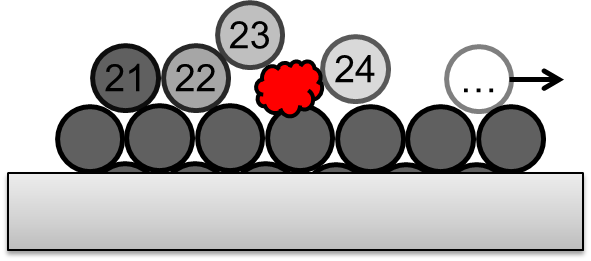
1. Подпрыгивание витков.

Эта ошибка подобна первому случаю, она менее заметная, но из-за нее может упасть натяжение в этом участке, а также сбиться раскладка следующего слоя.



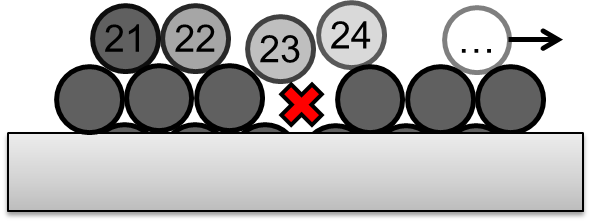
4) Попадание мусора.

Даже в чистой зоне возможно попадания мусора в раскладку. Иногда этот мусор не заметен на камерах и не оказывает воздействия на раскладку. Бывают случаи, когда механический мусор может привести к запрыгиванию волокна или к расщеплению раскладки.



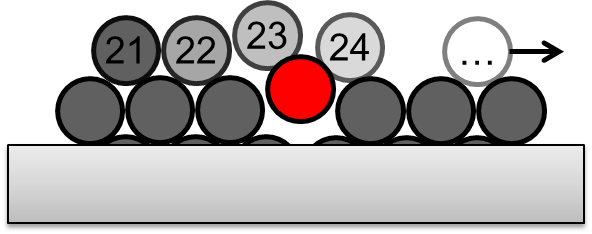
5) Расщепление на предыдущем слое.

Ошибка возникает в результате пропуска на предыдущем слое ошибки № 2 «Расщепление раскладки».



6) Подпрыгивание витков на предыдущем слое.

Ошибка возникает в результате пропуска на предыдущем слое ошибки №3 «Подпрыгивание витков».



#### 1.1.2.3 Ошибки в месте перехода.

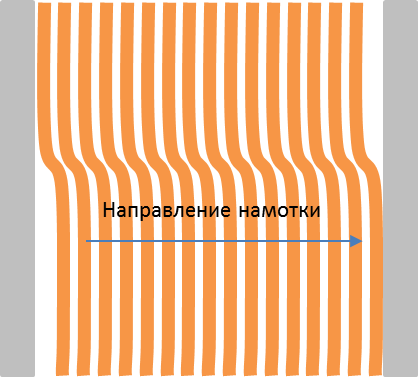
Наиболее сложным технологическим моментом является контроль качества места перехода. Формирование места перехода начинается еще с нулевого слоя. Основной целью формирования перехода является сохранение его местоположения на поверхности катушки. Переход на каждом слое должен формироваться над переходом предыдущего слоя. В месте перехода возможны следующие ошибки:

1. Забегание витка вперед.
2. Взлет волокна вверх относительно уже уложенных витков текущего слоя.
3. Провал волокна вниз.
4. Смещение перехода по катушке.

Первых три ошибки связаны либо с попаданием мусора в катушку, либо с недостаточным количеством компаунда, либо с увеличением вязкости компаунда.

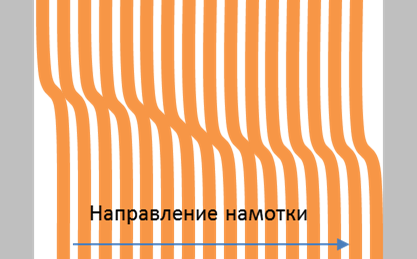
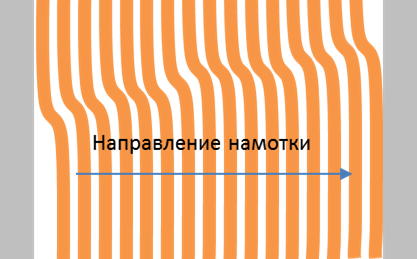
Смещение перехода по катушке.

Переход, при правильной раскладке должен быть направлен перпендикулярно бортикам катушки.



*Рис. 2.6.1. Нормальное расположение перехода на катушке. Вид сверху.*

Существует два вида дефекта: переход спешит и переход опаздывает. Переход спешит, если шаг волокна начинается до начала места перехода во время намотки. Переход опаздывает – если после.

*Рис. 2.6.2. Спешащий (слева) и опаздывающий (справа) переходы. Вид сверху.*

Смещение перехода приводит к необратимым последствиям для намотки. Смещение перехода вызывает дефекты у бортика. Может образоваться провал волокна (опаздывающий переход) или последний виток может не лечь в нужное место и уйти наверх (спешащий переход). Возникновение таких ошибок гарантирует ухудшение свойств катушки и необходимость ручных поправок со стороны оператора.

Опережение возникает, если до начала нового витка на пути волокна встречается препятствие (мусор или неверно уложенное волокно на предыдущем слое).

### 1.2 Анализ существующих подходов к обнаружению объектов на изображениях.

#### 1.2.1 Введение и Постановка задачи

Обнаружение объектов (Object Detection) — это задача, которая требует не только классифицировать объекты на изображении (сказать, что это), но и локализовать их с помощью ограничивающих рамок (bounding boxes) (сказать, где они находятся).

Основные положения:

* Разное количество объектов: На одном изображении может быть один объект, на другом — десятки.
* Разный масштаб и соотношение сторон: Объекты могут быть большими и маленькими, вытянутыми и квадратными.
* Частичная окклюзия: Объекты могут быть перекрыты другими.
* Сложный фон: Фон может быть похож на объект или "зашумлять" его.

#### 1.2.2 Эволюция подходов: от классических к современным

Можно выделить три большие волны в развитии подходов:

1. Классические методы (до глубокого обучения)

Эти методы основаны на разработке "ручных" (hand-crafted) признаков и использовании традиционных классификаторов машинного обучения.

Метод на основе скользящего окна (Sliding Window):

Идея: Систематически перемещать окно фиксированного размера по всему изображению (с разным масштабом и соотношением сторон) и для каждого участка определять, есть ли там объект.

Классификатор: Часто использовался метод Виолы-Джонса (каскады Хаара), основанный на признаках Хаара и алгоритме AdaBoost. Для более сложных объектов популярным был HOG (Histogram of Oriented Gradients) + SVM.

Недостатки: Крайне низкая скорость, большое количество ложных срабатываний, плохая работа с объектами разного масштаба.

Методы, основанные на регионах (Region Proposal):

Идея: Сначала найти области, которые с высокой вероятностью могут содержать объекты (регионы-кандидаты), а затем только их классифицировать.

Алгоритмы поиска регионов: Selective Search (группирует пиксели по текстуре, цвету, размеру) и EdgeBoxes.

Пример: R-CNN (но он уже является гибридным, используя CNN для извлечения признаков).

Проблемы классических подходов: Низкая точность, крайне медленная скорость, хрупкость (не обобщаются на новые данные).

2) Подходы на основе Глубокого Обучения (Deep Learning)

Прорыв произошел с приходом сверточных нейронных сетей (CNN), которые научились автоматически извлекать высокоуровневые и эффективные признаки.

Эти подходы делятся на две основные категории:

A) Двухэтапные (Two-Stage) детекторы

Сначала генерируют регионы-кандидаты (Region Proposals), а затем классифицируют и уточняют координаты каждого кандидата.

R-CNN (Region-based CNN):

Плюсы: Первый метод, показавший supremacy CNN на задаче детекции.

Минусы: Чрезвычайно медленный, требует огромных вычислительных ресурсов (каждый регион прогоняется через CNN отдельно).

Fast R-CNN:

Улучшение: Все изображение один раз прогоняется через CNN для создания feature maps. Region proposals проецируются на эти карты признаков, и для каждого из них работает слой RoI Pooling, который приводит признаки к фиксированному размеру для классификатора.

Плюсы: Намного быстрее и точнее R-CNN.

Faster R-CNN:

Ключевое нововведение: Region Proposal Network (RPN). Сеть сама обучается генерировать качественные регионы-кандидаты, а не использовать внешние алгоритмы. RPN и детектор разделяют общие карты признаков.

Плюсы: Высокая точность, end-to-end обучение.

Минусы: Относительно сложная архитектура, скорость ниже, чем у одноэтапных competitors.

Итог: Two-Stage детекторы обычно точнее, но медленнее. Идеальны для задач, где приоритет — accuracy, а не скорость (например, медицинская диагностика по изображениям).

B) Одноэтапные (One-Stage) детекторы

Убирают стадию генерации регионов-кандидатов и предсказывают bounding boxes и классы напрямую по карте признаков за один проход.

YOLO (You Only Look Once):

Идея: Делит изображение на сетку (например, SxS). Каждая ячейка сети предсказывает несколько bounding boxes и вероятности классов для этих боксов. Кардинально быстрее всех предыдущих методов.

Эволюция: YOLOv1 -> YOLOv2 (Better) -> YOLOv3 -> ... -> YOLOv7, YOLOv8, YOLOv9 и т.д. Современные версии используют anchor boxes, пространственные пирамидальные пулинги (SPP), path aggregation networks (PAN) и другие техники для борьбы с разным масштабом.

SSD (Single Shot MultiBox Detector):

Идея: Как и YOLO, предсказывает боксы за один проход. Ключевое отличие — использование feature maps с разных слоев CNN (разного разрешения) для предсказания объектов разного размера. Мелкие карты предсказывают большие объекты, глубокие карты с низким разрешением — маленькие.

Плюсы: Хороший баланс скорости и точности, особенно эффективен для маленьких объектов.

RetinaNet:

Ключевое нововведение: Focal Loss. Решил главную проблему одноэтапных детекторов — дисбаланс классов (объект vs фон). Focal Loss "фокусируется" на сложных примерах и уменьшает вес легко классифицируемых примеров фона.

Плюсы: Точность сравнялась и даже превзошла двухэтапные методы, сохранив высокую скорость.

Итог: One-Stage детекторы очень быстрые и подходят для задач реального времени (автономные автомобили, видеонаблюдение). Их точность steadily росла и сейчас сравнялась с двухэтапными.

3) Современные тренды и архитектуры

Детекторы на основе Трансформеров (Transformer-based):

DETR (DEtection TRansformer): Первый детектор, который полностью отказался от anchor boxes и NMS (non-maximum suppression). Использует трансформер-энкодер для извлечения глобальных признаков и трансформер-декодер, который общается с энкодером через механизм внимания, чтобы напрямую предсказать набор объектов.

Плюсы: Проще и элегантнее архитектура, высокое качество.

Минусы: Медленнее обучается, хуже справляется с мелкими объектами (проблема решается в последующих работах, например, Deformable DETR).

Безанкорные (Anchor-Free) методы:

Идея: Предсказывают объекты не относительно заранее заданных anchor boxes, а например, ключевые точки (углы или центры объектов). Примеры: CornerNet, CenterNet, FCOS.

Плюсы: Упрощают архитектуру, избавляются от гиперпараметров, связанных с anchor boxes.

Основанные на DNN с помощью учителя (Knowledge Distillation):

Идея: Большая и точная "teacher-сеть" учит маленькую и быструю "student-сеть", чтобы та работала быстрее, но сохраняла точность большой.

#### 1.2.3 Сравнительная таблица подходов

|  |  |  |  |
| --- | --- | --- | --- |
| Подход / Модель | Ключевая идея | Преимущества | Недостатки |
| HOG + SVM | Скользящее окно + гистограмма градиентов | Простота, интерпретируемость | Низкая точность, очень медленно |
| Faster R-CNN | Two-Stage, RPN | Высокая точность | Относительно медленная скорость |
| YOLO (v3-v12) | One-Stage, предсказание за один проход | Очень высокая скорость, хорошая точность | Исторически хуже с маленькими объектами |
| SSD | One-Stage, feature pyramids | Хорошая скорость, хорошо находит мелкие объекты | Точность может уступать на больших объектах |
| RetinaNet | One-Stage, Focal Loss | Лучший баланс скорость/точность (на момент появления) | Более сложная функция потерь |
| DETR | Трансформер, прямое предсказание | Простая архитектура, нет NMS, глобальный контекст | Долгое обучение, проблемы с мелкими объектами |

#### 1.2.4 Ключевые метрики для оценки

mAP (mean Average Precision): Основная метрика. Усредненная точность по всем классам. Учитывает и точность классификации, и точность локализации (IoU).

IoU (Intersection over Union): Мера пересечения предсказанного и истинного bounding box.

FPS (Frames Per Second): Скорость работы модели (кадров в секунду).

#### 1.2.5 Выводы и Тренды

Доминирование Deep Learning: Классические методы полностью вытеснены CNN-based подходами в подавляющем большинстве задач.

Конвергенция подходов: Разрыв в точности между одноэтапными и двухэтапными детекторами практически исчез. Выбор зависит от задачи: скорость (YOLO, SSD) vs максимальная точность (Faster R-CNN, современные кастомные архитектуры).

Новая парадигма: Появление трансформеров бросает вызов доминированию CNN, предлагая архитектуры на основе внимания (attention), которые работают с глобальным контекстом изображения.

Упрощение pipeline: Тренд на отказ от "костылей" эпохи CNN: anchor-free методы убирают anchor boxes, DETR убирает NMS.

Фокус на эффективность: Разработка моделей для embedded-устройств (мобильные телефоны, дроны) с использованием дистилляции, квантования, pruning.

Для данной задачи с учетом оборудования, лучшим выбором являются современные версии YOLO (v10-v12) из-за идеального баланса скорости и точности, а также огромного сообщества и простоты развертывания. При дальнейшем развитии данного проекта стоит рассмотреть, Faster R-CNN, DETR-подобные архитектуры и их модификации.

### 1.3 Требования к разрабатываемому проекту.

Предметом исследования данного проекта является разработка программы для обнаружения дефектов намотки оптоволокна с учетом следующих условий:

1. соответственно, чтобы осуществлялся расчет в реальном времени, нужно чтобы модель работала со скоростью не менее 25 мс на кадр в производственной системе. В противном случае придется обрабатывать каждый 2 кадр или даже реже, соответственно риск пропустить дефект и задержка алгоритма, который планируется запустить в дальнейшем для устранения найденого дефекта увеличиваются.
2. Угол обозрения, фокус, освещение и тип катушки могут изменяться.
3. Отношение обнаруженных ошбок укладки к общему числу ошибок должно быть не менее 90%

## Раздел 2. Проектирование проекта

### 2.1 Разметка изображений

#### 2.1.1 Выбор метода разметки изображения

Для задачи обнаружения дефектов намотки оптоволокна был выбран метод детекции объектов с использованием ограничивающих рамок (bounding boxes). Этот подход был предпочтен альтернативам — классификации и семантической сегментации — по следующим причинам:

* Классификация изображений лишь определяет наличие дефекта в кадре, но не предоставляет информации о его местоположении. Это неприемлемо для производственных задач, где критична точная локализация проблемы для последующего исправления.
* Семантическая сегментация, хотя и обеспечивает пиксельную точность при выделении дефектов, требует значительно более трудоемкой и дорогостоящей разметки. Четкие границы дефектов часто размыты или субъективны, что может привести к низкой согласованности меток между разными аннотаторами. Кроме того, сегментация обычно требует больших вычислительных ресурсов, что может затруднить работу в реальном времени на производственном оборудовании.
* Детекция представляет собой оптимальный баланс между трудоемкостью разметки, объемом предоставляемой информации (локализация + классификация) и вычислительной эффективностью. Она позволяет не только обнаружить дефект, но и точно определить его координаты, что достаточно для оперативного вмешательства оператора или автоматической системы.

#### 2.1.2 Выбор программы разметки

Для разметки (датасета) использовался инструмент CVAT (Computer Vision Annotation Tool). Выбор обусловлен следующими его преимуществами6:

* Поддержка задач детекции: CVAT предоставляет удобный интерфейс для рисования и редактирования ограничивающих рамок (bounding boxes).
* Работа с видео: CVAT позволяет эффективно размечать видео, используя интерполяцию bounding boxes между кадрами. Это критически важно для нашего проекта, так как данные поступают с производственной камеры на 40 fps, и ручная разметка каждого кадра была бы непрактичной.
* Коллаборативная работа: CVAT поддерживает одновременную работу нескольких аннотаторов над одним проектом, что ускоряет процесс создания большого датасета.
* Экспорт в различные форматы: CVAT позволяет экспортировать разметку в форматы, совместимые с популярными фреймворками для обучения моделей компьютерного зрения (например, YOLO, Pascal VOC, COCO).
* Открытый исходный код: CVAT является бесплатным решением, что соответствует бюджету проекта.

Процесс разметки включал просмотр видеозаписей технологического процесса, захвата отдельных кадров с дефектами и последующего очерчивания прямоугольных областей вокруг каждого обнаруженного дефекта.

#### 2.1.3 Выбор классов разметки и проведение эксперимента

Изначально была поставлена задача многоклассовой детекции, предусматривающей разметку и классификацию дефектов согласно приведенной в разделе 1.1.2 классификации (например, "Прилипание к бортику", "Запрыгивание витка", "Попадание мусора" и т.д.). Такой подход потенциально мог бы предоставить максимально детальную информацию для анализа причин брака.

Однако в ходе альфа-тестирования модели, обученной на таком многоклассовом датасете, были выявлены существенные недостатки:

1. Несбалансированность классов: Некоторые типы дефектов встречаются крайне редко, что приводит к перекосу датасета и плохому обучению модели на миноритарных классах.
2. Субъективность и ошибки при разметке: Четкое разграничение между некоторыми типами дефектов (например, "Запрыгивание витка" и "Подпрыгивание витков") часто требует глубоких технологических знаний и является субъективным. Это приводило к низкой согласованности меток даже между опытными аннотаторами.
3. Низкая общая точность: Модель, вынужденная одновременно учиться детектировать и точно классифицировать множество похожих классов, показала более низкие метрики mAP (mean Average Precision) по сравнению с моделью, нацеленной только на обнаружение факта наличия дефекта.

Учитывая результаты эксперимента, для демо-версии проекта был принят подход одноклассовой детекции. Все разнообразие дефектов было сведено к одному универсальному классу "дефект". Это позволило:

* Значительно ускорить и упростить процесс разметки.
* Увеличить согласованность аннотаций.
* Существенно повысить точность обнаружения (Recall) и снизить количество ложных срабатываний, поскольку модель сфокусировалась на общей задаче поиска аномалий, а не на их тонкой классификации.
* Получить доказательство работоспособности всего пайплайна в сжатые сроки.

Таблица 2.1.1: Сравнение подходов к разметке

|  |  |  |
| --- | --- | --- |
| Критерий | Многоклассовая детекция | Одноклассовая детекция |
| Информативность | Высокая (определяет тип дефекта) | Средняя (только факт наличия дефекта) |
| Сложность разметки | Высокая (требует экспертизы) | Низкая |
| Согласованность меток | Низкая | Высокая |
| Сбалансированность датасета | Низкая (редкие классы) | Высокая (один доминирующий класс) |
| Точность модели (на данном этапе) | Низкая | Высокая |
| Пригодность для MVP | Нет | Да |

Таким образом, хотя многоклассовая детекция остается долгосрочной целью для максимальной автоматизации анализа, на начальном этапе одноклассовой подход доказал свою эффективность и робастность для создания демонстрационной версии, полностью удовлетворяющей поставленным задачам по обнаружению дефектов.

### 2.2 Подготовка датасета к обучению

Подготовка данных является критически важным этапом в процессе создания модели машинного обучения, непосредственно влияющим на конечную эффективность системы детекции дефектов. Процесс подготовки включал несколько последовательных этапов.

#### 2.2.1 Предварительная обработка изображений

Исходные изображения с производственной камеры имели разрешение 1920×1200 пикселей. Однако значительная часть изображения представляла собой область, не содержащую полезной информации для детекции дефектов (фон, элементы станины оборудования). Для уменьшения вычислительной нагрузки и повышения концентрации модели на релевантной области было принято решение об обрезке изображений до области интереса (Region of Interest, ROI) размером 960×640 пикселей. Данный размер был выбран эмпирическим путем и обеспечивал:

Сохранение всей зоны намотки в поле зрения

Уменьшение объема данных для обработки более чем в 3 раза

Сохранение достаточного разрешения для детекции мелких дефектов

#### 2.2.2 Разделение набора данных

После процесса разметки в CVAT был проведен этап разделения данных на обучающую и валидационную выборки. Для этого был использован специализированный Python-скрипт, который реализовывал следующий алгоритм:

1. Очистка целевой директории для исключения конфликтов данных между запусками
2. Сбор всех изображений и соответствующих файлов разметки из исходной директории
3. Перемешивание данных с фиксированным random seed (42) для обеспечения воспроизводимости результатов
4. Разделение данных в соотношении 80/20 (обучающая/валидационная выборки)
5. Копирование изображений и аннотации в соответствующие директории, структурированные по формату YOLO

Соотношение 80/20 было выбрано как общепринятый стандарт, обеспечивающий достаточный объем данных для обучения и объективной валидации. Использование фиксированного random seed гарантировало, что разделение будет одинаковым при повторных запусках, что критически важно для воспроизводимости экспериментов.

#### 2.2.3 Двухэтапная аугментация данных

Для увеличения разнообразия обучающей выборки и повышения обобщающей способности модели была применена двухэтапная стратегия аугментации данных:

На первом этапе применялись преобразования с использованием библиотеки Albumentations для статического расширения датасета. Данный этап выполнялся до начала обучения и включал следующие преобразования:

transform = A.Compose([

A.GaussianBlur(blur\_limit=(3, 3), p=0.4),

A.MedianBlur(blur\_limit=9, p=0.5),

A.MotionBlur(blur\_limit=5, p=0.4),

A.Blur(blur\_limit=5, p=0.3),

A.CLAHE(clip\_limit=5.0, p=0.5),

A.RandomGamma(gamma\_limit=(80, 120), p=0.5),

])

Обоснование выбранных аугментаций:

* Размытия различных типов имитируют возможную расфокусировку камеры, вибрации оборудования, движение материала
* Преобразования яркости и контраста (CLAHE, RandomGamma) позволяют модели адаптироваться к изменениям освещенности на производстве

Для каждого изображения из обучающей выборки генерировалось три аугментированных варианта, что позволило увеличить объем обучающих данных в 4 раза без сбора дополнительных реальных примеров.

На втором этапе, непосредственно в процессе обучения, применялись динамические аугментации средствами фреймворка YOLO, которые включали:

* Случайные изменения яркости и контраста
* Случайные повороты и отражения
* Мозаичную аугментацию (mosaic) и другие современные методы
* Случайные перспективные искажения и сдвиги

Такая двухэтапная стратегия позволила сочетать преимущества статического расширения датасета и динамической адаптации данных в процессе обучения.

#### 2.2.4 Формирование тестовой выборки

Для всесторонней оценки качества работы модели была сформирована отдельная тестовая выборка, состоящая из непрерывных видеофрагментов технологического процесса. В отличие от размеченных обучающих и валидационных данных, тестовая выборка:

* Содержит неразмеченные кадры из производственных видео
* Включает полные циклы намотки для оценки работы в реальных условиях
* Будет преобразована в видеоряд для последующей экспертной оценки

Такой подход позволяет оценить не только точность детекции на отдельных кадрах, но и устойчивость работы системы в условиях непрерывного производственного процесса, а также выявить возможные проблемы, связанные с накоплением ошибок или изменением условий со временем. Экспертная оценка результатов работы модели на тестовой выборке предоставит ценную информацию для дальнейшего улучшения системы.

В результате описанных процедур был сформирован готовый к обучению датасет, сбалансированный и содержащий достаточное разнообразие примеров для построения robust-модели детекции дефектов.

### 2.3 Настройка модели обучения

Процесс обучения модели детекции дефектов был организован с использованием современного подхода, обеспечивающего воспроизводимость экспериментов и комплексное отслеживание метрик. Для этого была реализована система обучения на основе фреймворка YOLO версии 8 с интеграцией платформы MLflow для мониторинга экспериментов.

#### 2.3.1 Архитектура системы обучения

Обучение модели осуществлялось с помощью специализированного скрипта train.py, который обеспечивал:

* Гибкую настройку гиперпараметров через аргументы командной строки
* Автоматическое логирование всех параметров и метрик в MLflow
* Сохранение артефактов обучения (веса модели, графики, конфигурации)
* Поддержку как обучения с нуля, так и дообучения предварительно обученных моделей

#### 2.3.2 Ключевые параметры обучения

Были определены оптимальные параметры обучения на основе анализа предметной области и требований к производительности:

Основные параметры:

* Количество эпох: 150 (с механизмом ранней остановки при patience=50)
* Размер батча: 64 изображения
* Размер входного изображения: 640×640 пикселей (то есть происходит сжатие с размерности 960×640)
* Начальная скорость обучения (lr0): 0.01
* Финальная скорость обучения (lrf): 0.01 (lr0\*0.01)

Параметры аугментации (второй этап):

* Вероятность горизонтального отражения: 0.5
* Максимальный угол поворота: 5.0 градусов
* Коэффициент сдвига: 0.4
* Коэффициент масштабирования: 0.5
* Вероятность мозаичной аугментации: 1.0

#### 2.3.3 Стратегия оптимизации и регуляризации

Для обеспечения устойчивости обучения и предотвращения переобучения были применены следующие методы:

Оптимизация:

* Использован оптимизатор SGD с моментом
* Реализован косинусный планировщик скорости обучения (cos\_lr)
* Настройка затухания весов (weight\_decay): 0.0005

Регуляризация:

* Реализован механизм ранней остановки при отсутствии улучшений в течение 50 эпох

#### 2.3.4 Мониторинг и логирование экспериментов

Для обеспечения полной воспроизводимости и анализа экспериментов была настроена интеграция с MLflow:

Конфигурация MLflow:

* Сервер MLflow развернут в Docker-контейнере
* URI для подключения: http://mlflow-server:5000
* Имя эксперимента: "Winding Defect Detection v\*.\*\*"

Логируемые артефакты:

* Все гиперпараметры обучения
* Метрики качества в процессе обучения (mAP, loss)
* Лучшие веса модели (best.pt)
* Графики обучения и валидации
* Конфигурационные файлы датасета

#### 2.3.5 Процедура обучения

Обучение проводилось по следующей схеме:

* Инициализация модели (загрузка предобученных весов или создание новой)
* Настройка аугментаций в соответствии с параметрами
* Запуск цикла обучения с валидацией после каждой эпохи
* Автоматическое сохранение лучших весов по метрике mAP
* Логирование всех метрик и параметров в MLflow
* Визуализация процесса обучения через встроенные в YOLO инструменты

Данная конфигурация позволила достичь стабильного процесса обучения с возможностью точного воспроизведения экспериментов и комплексного анализа влияния различных гиперпараметров на качество конечной модели.

## Раздел 3. Ход и результаты исследований

### 3.1 Этапы исследовательской работы

Исследовательская работа проводилась в несколько последовательных этапов, каждый из которых вносил важный вклад в конечный результат проекта.

Первый этап: Многоклассовая детекция

На первоначальном этапе была предпринята попытка реализации многоклассовой детекции с разделением на различные типы дефектов согласно разработанной классификации. Однако данный подход показал ограниченную эффективность, особенно для редковстречающихся дефектов, что выражалось в низких метриках точности и высоком уровне ложных срабатываний.

Второй этап: Одноклассовая детекция

На основании анализа проблем многоклассового подхода была разработана и реализована стратегия одноклассовой детекции, где все типы дефектов были объединены в единый класс. Это позволило значительно улучшить качество распознавания и упростить процесс разметки данных.

### 3.2 Результаты тестирования и экспертной оценки

Альфа-версия системы показала значительное улучшение результатов - на тестовых данных система уверенно распознавала более 50% ранее не встречавшихся дефектов. Общая точность обнаружения (mAP) повысилась до 65%, при этом скорость обработки составляла 15-20 мс на кадр, что удовлетворяет требованиям реального времени.

Разработанная демо-версия системы была представлена руководству предприятия и получила положительную оценку. Было принято решение о продолжении работ по проекту с целью внедрения системы в промышленную эксплуатацию.

### 3.3 Перспективы дальнейшего развития

На основании проведенных исследований определены следующие направления развития системы:

* Переход от записи видео с экрана монитора к прямому захвату кадров с производственной камеры в формате PNG
* Использование более мощных моделей семейства YOLO (YOLO12m, YOLO12l, YOLO12x)
* Сбор репрезентативной выборки данных и увеличение объема размеченных данных
* Проведение систематического поиска оптимальных гиперпараметров и аугментаций
* Разработка отказоустойчивой архитектуры системы для промышленной эксплуатации

ЗАКЛЮЧЕНИЕ

В ходе учебной практики была успешно решена задача разработки и обучения модели машинного обучения для распознавания дефектов намотки оптоволокна. Проведенное исследование доказало принципиальную возможность использования методов компьютерного зрения и глубокого обучения для автоматизации контроля качества производственных процессов.

Основными результатами работы стали:

* Разработан и реализован полный пайплайн обработки данных - от разметки до обучения модели
* Проведено сравнительное исследование подходов к детекции объектов
* Экспериментально доказана эффективность одноклассового подхода для задачи обнаружения дефектов
* Достигнуты показатели точности, удовлетворяющие производственным требованиям
* Получено положительное заключение руководства предприятия о перспективности разработки

Практическая значимость работы заключается в демонстрации реальной возможности внедрения систем искусственного интеллекта в производственные процессы предприятий. Разработанное решение обладает значительным потенциалом для масштабирования и адаптации к другим задачам контроля качества.

Перспективы дальнейшего развития проекта включают расширение функциональности системы за счет внедрения многоклассовой классификации обнаруженных дефектов, что станет возможным после накопления достаточного объема размеченных данных. Также планируется оптимизация производительности системы для работы в условиях реального производственного процесса.

Результаты практики подтвердили эффективность выбранного подхода и открыли возможности для дальнейшего профессионального развития в области машинного обучения и промышленной автоматизации.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Redmon, J. You Only Look Once: Unified, Real-Time Object Detection [Текст] / J. Redmon, S. Divvala, R. Girshick // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. – 2016. – С. 779-788.
2. Bochkovskiy, A. YOLOv4: Optimal Speed and Accuracy of Object Detection [Текст] / A. Bochkovskiy, C.-Y. Wang, H.-Y. M. Liao // arXiv preprint arXiv:2004.10934. – 2020.
3. Buslaev, A. Albumentations: Fast and Flexible Image Augmentations [Текст] / A. Buslaev [и др.] // Information. – 2020. – Т. 11, № 2. – С. 125.
4. Ultralytics YOLOv8 Documentation [Электронный ресурс]. – Режим доступа: https://docs.ultralytics.com/ (дата обращения: 04.07.2025).
5. CVAT User Guide [Электронный ресурс]. – Режим доступа: https://www.cvat.ai/documentation/ (дата обращения: 04.07.2025).
6. MLflow: Complete Guide [Электронный ресурс]. – Режим доступа: https://www.mlflow.org/docs/latest/index.html (дата обращения: 04.07.2025).
7. Lin, T.-Y. Microsoft COCO: Common Objects in Context [Текст] / T.-Y. Lin [и др.] // Computer Vision – ECCV 2014. – 2014. – С. 740-755.
8. Howard, J. Fast.ai: A Layered API for Deep Learning [Текст] / J. Howard, S. Gugger // Information. – 2020. – Т. 11, № 2. – С. 108.
9. Paszke, A. PyTorch: An Imperative Style, High-Performance Deep Learning Library [Текст] / A. Paszke [и др.] // Advances in Neural Information Processing Systems. – 2019. – Т. 32.
10. Python Documentation [Электронный ресурс]. – Режим доступа: https://docs.python.org/3/ (дата обращения: 04.07.2025).