**2. Постановка задачи классификации для выявления дефектов на БПЛА  
2.1 Постановка задачи классификации  
2.2 Описание данных**

* Входные данные
* Типы дефектов и классы
* Объем данных
* Алфавит классов
* Реалистичность условий съемки
* Предобработка данных

**2.3 Методы аугментации данных  
2.4 Построение нейронной сети**

* Параметры модели и пути к данным
* Архитектура нейронной сети
* Компиляция модели
* Сохранение гиперпараметров  
  **2.5 Процесс обучения**
* Подготовка данных для обучения
* Запуск обучения
* Валидация модели
* Оценка модели на тестовых данных
* Анализ результатов классификации
* Визуализация процесса обучения  
  **2.6 Результаты обучения**
* Графики потерь и точности
* Отчет по классификации на тестовой выборке
* Общая оценка производительности модели  
  **2.7 Сравнение с другими моделями**
* AlexNet
* VGG-16
* Сравнение результатов
* Возможные улучшения

**2. Постановка задачи классификации для выявления дефектов на БПЛА**

**2.1 Постановка задачи классификации**

Классификация изображений — это одна из ключевых задач в области машинного обучения и компьютерного зрения, которая направлена на распределение входных данных по заданным категориям. В последние годы задачи классификации нашли широкое применение в промышленности, медицине, науке и других областях, где необходима автоматизация процесса распознавания объектов, явлений или состояний. Применительно к беспилотным летательным аппаратам (БПЛА) задачи классификации приобрели особое значение, так как позволяют решать вопросы диагностики состояния техники и обнаружения потенциально опасных дефектов, что в конечном итоге обеспечивает безопасную эксплуатацию и повышает надежность авиационных систем. [1, 3]

Традиционные методы диагностики дефектов на воздушных судах включают визуальный осмотр, ультразвуковую и радиографическую диагностику, а также другие неразрушающие методы контроля. Однако эти подходы требуют значительных человеческих ресурсов и временных затрат, так как для выполнения каждого осмотра требуется квалифицированный персонал и специализированное оборудование. Более того, такие методы обладают ограниченной способностью обнаруживать мелкие или скрытые дефекты, особенно в условиях труднодоступных для осмотра областей. В результате возникает необходимость создания систем автоматизированной диагностики, которые могли бы анализировать изображения и обнаруживать дефекты с высокой точностью. [4, 29]

Важной частью разработки системы автоматизированного контроля на основе БПЛА является задача классификации полученных изображений дефектов. Автоматическое определение и классификация повреждений помогают минимизировать влияние человеческого фактора и ускоряют процесс принятия решений. Для реализации такого подхода используются методы машинного обучения, в частности, нейронные сети, которые способны обрабатывать сложные визуальные данные и выделять ключевые признаки, указывающие на тип дефекта. [1]

Целью данного исследования является разработка модели нейронной сети, способной классифицировать различные типы повреждений на основе изображений, полученных с камер, установленных на БПЛА. Данная система автоматической классификации должна удовлетворять ряду требований, включая высокую точность, способность работать с реальными изображениями, полученными в полевых условиях, а также устойчивость к различным условиям съемки (разные углы, освещение и т.д.). Подобная система может стать важным компонентом для обеспечения своевременного обнаружения дефектов, что позволит снизить эксплуатационные риски и обеспечить высокую надежность летательных аппаратов. [3, 30]

Создание такой системы классификации требует решения ряда задач:

1. **Обработка и подготовка данных**. В первую очередь необходимо собрать и подготовить изображения, представляющие различные типы дефектов. Эти данные будут использоваться как для обучения модели, так и для её тестирования. [18]
2. **Создание архитектуры нейронной сети**. Нужно построить архитектуру, способную выделять признаки и осуществлять классификацию с учетом сложности и разнообразия входных данных. [21]
3. **Аугментация данных**. Поскольку в реальных условиях БПЛА могут работать при разных освещениях, углах съемки и фонах, требуется создать разнообразие тренировочных данных, используя методы аугментации, которые позволяют модели адаптироваться к различным условиям. [23]
4. **Процесс обучения и валидации**. Модель должна пройти процесс обучения и валидации, чтобы достичь приемлемого уровня точности, который позволит использовать её на практике. [15]
5. **Оценка качества модели**. Для подтверждения эффективности модели требуется провести оценку на тестовых данных, а также сравнить её с другими архитектурами нейронных сетей, чтобы выбрать наиболее подходящий подход. [20]

Предложенная система классификации должна решать задачу многоклассовой классификации, так как изображения дефектов будут разделяться на три основные категории: *вмятины* (Dent), *повреждение крепежных элементов* (Fastener Damage) и *разрывы* (Rupture). Модель должна научиться выделять специфические признаки для каждой категории дефектов, чтобы классифицировать изображения с высокой точностью. [26]

Таким образом, основная задача классификации дефектов на БПЛА в рамках данного исследования заключается в создании модели, способной обрабатывать изображения повреждений и относить их к одному из заданных классов дефектов. В отличие от других подходов, система на основе нейронных сетей обладает высокой гибкостью, так как может работать в условиях ограниченного освещения, различных углов обзора и других факторов, которые часто влияют на качество диагностики. [1]

Применение нейронных сетей для классификации дефектов открывает широкие перспективы в области диагностики БПЛА. Система классификации на базе нейронной сети способна обрабатывать большое количество изображений за короткое время, что делает её незаменимым инструментом для анализа состояния техники и принятия своевременных мер по устранению дефектов. [19]

**2.2 Описание данных**

Для построения и обучения модели классификации в данном исследовании был использован набор изображений повреждений различных частей летательных аппаратов, полученных с камер, установленных на беспилотных летательных аппаратах (БПЛА). Эти данные были специально собраны для автоматического анализа и классификации дефектов, что делает их особенно полезными для разработки системы диагностики. [30]

**Входные данные**

**Формат и структура изображений.** Все изображения, используемые для обучения и оценки модели, имеют фиксированное разрешение 640x640 пикселей и представлены в цветовой схеме RGB (Red-Green-Blue). Высокое разрешение и наличие цветовой информации позволяют модели учитывать все визуальные признаки, что повышает её способность к точной классификации. Изображения были предварительно обработаны для унификации размеров, что упрощает дальнейшую работу с ними и помогает стандартизировать процесс подачи данных в модель. [4]

**Типы дефектов и классы.** В рамках данного исследования классифицируются три основных типа дефектов, которые могут возникнуть на различных частях летательного аппарата:

1. **Вмятины (Dent)** — это небольшие деформации, которые могут появляться на поверхности летательного аппарата в результате механических воздействий. Вмятины могут казаться малозначительными, но в некоторых случаях они могут быть маркером более глубоких структурных повреждений.
2. **Повреждения крепежных элементов (Fastener Damage)** — дефекты, связанные с повреждением или нарушением целостности крепежных элементов, таких как болты, винты и другие соединительные элементы. Эти повреждения могут оказывать значительное влияние на прочность конструкции и должны быть своевременно обнаружены. [31]
3. **Разрывы (Rupture)** — крупные повреждения, представляющие собой трещины или разрывы в обшивке или других структурных элементах летательного аппарата. Разрывы являются самыми опасными повреждениями и требуют немедленного вмешательства для предотвращения аварийных ситуаций. [30]

Каждая из перечисленных категорий дефектов выделяется специфическими визуальными признаками, которые используются моделью для выполнения классификации. Таким образом, набор данных включает три класса, каждый из которых отражает определенный тип повреждения, что позволяет решать задачу многоклассовой классификации. [3]

**Объем данных.** Общий объем набора данных составил 6803 изображения, что обеспечивает достаточное количество примеров для обучения, валидации и тестирования модели. Изображения были разделены на подмножества:

* **Обучающая выборка** — основной набор изображений, используемый для настройки весов и обучения модели.
* **Валидационная выборка** — используется для оценки производительности модели на изображениях, не включенных в обучение, с целью предотвращения переобучения.
* **Тестовая выборка** — применяется для окончательной оценки качества модели после завершения этапа обучения. Этот набор позволяет проверить способность модели классифицировать ранее не виденные изображения и определить, насколько эффективно она будет работать в реальных условиях. [18]

**Алфавит классов**

Для упрощения процесса классификации и интерпретации результатов каждому классу было присвоено уникальное название:

* **Dent** — вмятины;
* **Fastener Damage** — повреждения крепежных элементов;
* **Rupture** — разрывы.

Такой алфавит классов используется моделью в процессе классификации для присвоения предсказаниям каждого изображения одного из этих трех типов дефектов. В дальнейшем это позволяет легко анализировать результаты классификации и выявлять долю правильно классифицированных изображений для каждого типа повреждений. [26]

**Реалистичность условий съемки**

Для достижения максимально возможной точности в реальных условиях эксплуатации данных для диагностики, изображения, собранные для этого исследования, охватывают широкий спектр условий, отражающих возможные сценарии полевой эксплуатации. Набор данных включает изображения, снятые при различных:

* **Углах обзора**. Изображения дефектов были получены с разных углов, что делает задачу классификации сложной, но одновременно приближает её к реальным условиям, где камеры БПЛА могут снимать объект под разными углами.
* **Условиях освещения**. Съемка выполнялась при различной яркости и освещенности, что также затрудняет классификацию, так как дефекты могут выглядеть по-разному в зависимости от освещения.
* **Ракурсах и фонах**. Некоторые изображения имеют различный фон или объекты на заднем плане, что делает задачу классификации более сложной, так как модель должна научиться выделять ключевые признаки дефектов, игнорируя нерелевантные детали фона. [23, 28]

Наличие подобных вариаций делает задачу классификации не только более сложной, но и более ценной с точки зрения приближенности к реальной эксплуатации. Модель, обученная на таком наборе данных, становится менее чувствительной к изменению условий съемки и способна выполнять классификацию в широком диапазоне сценариев. [13]

**Предобработка данных**

Для подготовки данных к обучению использовались методы нормализации и аугментации, позволяющие улучшить качество модели и её способность к обобщению. Все изображения были нормализованы, то есть значения пикселей были преобразованы в диапазон от 0 до 1, что помогает улучшить работу модели и избежать переобучения. [23]

Также для обучающей выборки была использована аугментация данных, которая позволила виртуально увеличить объем данных, создавая измененные версии изображений (подробнее описана в пункте 2.3). [24]

Таким образом, описанный набор данных обеспечивает достаточную базу для эффективного обучения нейронной сети, что позволит модели классифицировать изображения дефектов с высокой точностью в различных эксплуатационных условиях.

**2.3 Методы аугментации данных**

Аугментация данных — это процесс создания новых примеров на основе существующих изображений путем их преобразований и изменений, которые сохраняют основные характеристики, но добавляют вариативность. В задачах классификации изображений аугментация является одним из основных методов повышения качества и устойчивости моделей машинного обучения. Она особенно важна в тех случаях, когда объем доступных данных ограничен или требуется, чтобы модель могла справляться с разнообразными условиями, как это происходит в реальной эксплуатации. [23, 39]

Для задачи классификации дефектов на БПЛА аугментация данных позволила увеличить разнообразие тренировочного набора, что помогло модели стать более гибкой и устойчивой к изменениям в условиях съемки, таким как освещение, ракурс и фон. В данном исследовании использовались следующие методы аугментации данных:

**Повороты изображений (Rotation)**

В процессе съемки дефектов с беспилотных летательных аппаратов изображения могут быть получены под различными углами. Это особенно актуально для БПЛА, так как в зависимости от их положения в воздухе и угла камеры, снимки могут быть сделаны с различным наклоном. [23, 24]

Чтобы имитировать эту вариативность, изображения подвергались случайным поворотам на угол до 20 градусов. Такой подход позволяет улучшить устойчивость модели к наклонам и положениям объекта на изображении, что помогает избежать ошибки в классификации дефектов, которые имеют похожие признаки, но сняты под разными углами. Повороты помогают модели не привязываться к фиксированной ориентации объектов, повышая её способность к обобщению.

**Масштабирование (Zooming)**

Размер дефектов, таких как вмятины, разрывы и повреждения крепежных элементов, может значительно варьироваться. Например, некоторые повреждения могут занимать большую часть изображения, тогда как другие дефекты могут быть мелкими и едва заметными.

Для учета этой вариативности использовалось масштабирование изображений. Оно включало в себя как увеличение, так и уменьшение изображений на случайную величину, что позволило модели распознавать дефекты разных размеров. Масштабирование также помогло улучшить устойчивость модели к различной удаленности объекта на изображении, так как БПЛА может снимать повреждения с разных расстояний. [6]

**Горизонтальное отражение (Horizontal Flip)**

Благодаря горизонтальному отражению изображения модель обучалась распознавать дефекты в различных ориентациях. Этот метод особенно полезен в условиях, когда объект съемки (например, часть обшивки летательного аппарата) может находиться в различных положениях относительно камеры.

Отражение изображений по горизонтали добавляет симметричность в тренировочный набор, что помогает модели избегать переобучения на определенные стороны объектов. При этом модель лучше адаптируется к вариативности ориентаций объектов на изображениях, делая её более универсальной и устойчивой к различным условиям эксплуатации. [21]

**Изменение яркости и контрастности (Brightness and Contrast Adjustment)**

Условия освещения могут значительно варьироваться в зависимости от времени суток, погодных условий и интенсивности солнечного света. При диагностике с использованием БПЛА камера может фиксировать объект в условиях недостаточного или чрезмерного освещения, что затрудняет распознавание дефектов.

Изменение яркости и контрастности помогает модели справляться с этими вариациями, поскольку она обучается видеть изображения с разной освещенностью. Это делает модель более устойчивой к изменениям внешних условий и позволяет ей лучше адаптироваться к реальным сценариям, в которых БПЛА может фиксировать повреждения при различных уровнях освещения. [5]

**Сдвиг и деформация (Shifting and Shearing)**

Хотя данный метод менее значим в контексте классификации дефектов, небольшие сдвиги и деформации изображений помогают модели быть менее чувствительной к положению дефекта на изображении. Это полезно, когда точное расположение повреждения в кадре может не совпадать, особенно если учесть, что камера БПЛА не всегда фиксирует повреждения точно в центре кадра.

Сдвиг и деформация изображений создают дополнительные варианты расположения дефектов, что повышает обобщающую способность модели. Этот метод позволяет нейронной сети сосредоточиться на признаках дефектов, а не на их точном расположении на изображении, что делает её более устойчивой к различным позициям объектов. [7]

**Суммарная роль аугментации данных**

Аугментация данных сыграла важную роль в улучшении обобщающей способности модели, так как позволила увеличить объем данных и внести вариативность в тренировочный набор. Это привело к улучшению качества модели и её устойчивости к различным внешним факторам. [23, 24]

Методы аугментации были применены только к обучающей выборке, поскольку основной целью является улучшение адаптивности модели, тогда как валидационные и тестовые данные оставались неизменными для обеспечения объективной оценки её производительности. В процессе аугментации использовалась библиотека **ImageDataGenerator** из **TensorFlow** для генерации измененных изображений в реальном времени, что также помогло оптимизировать процесс и снизить затраты на хранение данных. [13]

Таким образом, аугментация данных помогла создать более гибкую и точную модель, которая способна работать в условиях реальной эксплуатации и справляться с вариациями в условиях съемки, которые могут возникать при диагностике дефектов на БПЛА.

**2.4 Построение нейронной сети**

В данной работе для решения задачи классификации дефектов на изображениях, полученных с беспилотных летательных аппаратов (БПЛА), была разработана сверточная нейронная сеть (CNN), реализованная с использованием библиотеки TensorFlow и её интерфейса Keras. [1, 6]

Ниже приведен код, описывающий архитектуру нейронной сети, а также подробное объяснение её основных слоев и гиперпараметров.

**Параметры модели и пути к данным**

Первым шагом была настройка базовых параметров, таких как размер входного изображения, количество эпох, размер батча и количество классов, к которым будет относиться каждая категория дефектов: [40]

# Параметры

img\_width, img\_height = 640, 640

batch\_size = 32

epochs = 50  # количество эпох для обучения

num\_classes = 3  # количество классов дефектов

Рисунок 8 - Основные параметры модели: размер изображений, размер мини-выборки, количество эпох и классов для классификации дефектов.

Эти параметры задают разрешение изображений, размер мини-выборки для обучения, количество эпох и число классов, в нашем случае — три категории: *Dent* (вмятины), *Fastener Damage* (повреждения крепежных элементов) и *Rupture* (разрывы).

Далее были определены пути к данным:

data\_dir = 'data'

train\_data\_dir = os.path.join(data\_dir, 'train')

validation\_data\_dir = os.path.join(data\_dir, 'valid')

test\_data\_dir = os.path.join(data\_dir, 'test')

Рисунок 9 - Структура директорий данных: разделение исходных данных на тренировочную, валидационную и тестовую выборки.

Определение директорий для обучающих, валидационных и тестовых данных

Эти директории хранят обучающие, валидационные и тестовые изображения, что позволяет разделить данные для разных этапов обучения и оценки модели. [7]

**Архитектура нейронной сети**

Архитектура модели создается с помощью класса Sequential, который обеспечивает последовательное добавление слоев. Архитектура включает входной слой, несколько сверточных и объединяющих слоев (MaxPooling), полносвязные слои и слой для предотвращения переобучения (Dropout):

model = Sequential([

    # входной слой с заданным размером изображения

    Input(shape=(img\_width, img\_height, 3)),

    # первый сверточный слой

    Conv2D(32, (3, 3), activation='relu'),

    # слой объединения признаков

    MaxPooling2D(pool\_size=(2, 2)),

    # второй сверточный слой

    Conv2D(64, (3, 3), activation='relu'),

    # слой объединения признаков

    MaxPooling2D(pool\_size=(2, 2)),

    # третий сверточный слой

    Conv2D(128, (3, 3), activation='relu'),

    # слой объединения признаков

    MaxPooling2D(pool\_size=(2, 2)),

    # слой преобразования в одномерный массив

    Flatten(),

    # полносвязный слой с 256 нейронами

    Dense(256, activation='relu'),

    # слой регуляризации с вероятностью 0.5

    Dropout(0.5),

    # выходной слой с Softmax

    Dense(num\_classes, activation='softmax')

])

Рисунок 10 - Архитектура сверточной нейронной сети для классификации дефектов.

Рассмотрим каждый из слоев этой архитектуры:

1. **Входной слой** Input(shape=(img\_width, img\_height, 3)): принимает изображения размером 640x640 пикселей с тремя каналами (RGB). Это задает форму данных, которые будут передаваться в сеть.
2. **Первый сверточный слой** Conv2D(32, (3, 3), activation='relu'): данный слой содержит 32 фильтра размером 3x3, которые помогают выделять базовые признаки изображения, такие как края, текстуры и другие элементы. Функция активации ReLU используется для добавления нелинейности, что помогает сети распознавать более сложные структуры. [20]
3. **Первый слой объединения (MaxPooling)** MaxPooling2D(pool\_size=(2, 2)): уменьшает размерность карты признаков, сохраняя наиболее важные признаки. Этот слой снижает вычислительную сложность модели и позволяет ей сосредоточиться на ключевых признаках. [21]
4. **Второй сверточный слой** Conv2D(64, (3, 3), activation='relu'): увеличивает глубину карты признаков до 64 фильтров, позволяя сети выявлять более сложные признаки и структуры.
5. **Второй слой объединения (MaxPooling)**: еще больше снижает размерность карты признаков, выделяя более высокоуровневые признаки, характерные для конкретных дефектов. [21]
6. **Третий сверточный слой** Conv2D(128, (3, 3), activation='relu'): третий сверточный слой содержит 128 фильтров и позволяет сети анализировать изображение на более детальном уровне, выделяя сложные структуры.
7. **Третий слой объединения (MaxPooling)**: после трех сверточных слоев и операций объединения сеть получает сжатое представление изображения с выделенными признаками, которые впоследствии используются для классификации.
8. **Слой преобразования в одномерный массив** Flatten(): преобразует многомерный выход предыдущего слоя в одномерный массив, который может быть подан на вход полносвязного слоя.
9. **Полносвязный слой** Dense(256, activation='relu'): этот слой содержит 256 нейронов и отвечает за обработку выделенных признаков перед финальной классификацией. С использованием функции активации ReLU слой позволяет сети различать ключевые признаки, которые определяют каждый класс. [14]
10. **Слой регуляризации (Dropout)** Dropout(0.5): данный слой отключает случайные нейроны с вероятностью 0.5, что помогает избежать переобучения и улучшить обобщающую способность сети. Dropout улучшает стабильность модели на новых данных. [15]
11. **Выходной слой** Dense(num\_classes, activation='softmax'): содержит 3 нейрона (по одному на каждый класс), а функция активации Softmax выдает вероятность принадлежности изображения к каждому из классов дефектов.

**Компиляция модели**

Для компиляции модели использовался оптимизатор **Adam**, функция потерь **categorical crossentropy** (категориальная кросс-энтропия), и метрика **accuracy** для оценки точности:

model.compile(

    optimizer=Adam(),

    loss='categorical\_crossentropy',

    metrics=['accuracy']

)

Рисунок 11 - Компиляция модели: настройка оптимизатора, функции потерь и метрики для оценки.

* **Adam** — адаптивный оптимизатор, который динамически изменяет скорость обучения для ускорения процесса. Это один из самых популярных оптимизаторов для задач глубокого обучения, который обеспечивает высокую скорость сходимости. [22]
* **Categorical Crossentropy** — функция потерь для многоклассовой классификации. Она вычисляет разницу между предсказанными вероятностями и истинными метками классов. [23]
* **Accuracy** — метрика, которая оценивает точность модели, показывая процент правильно классифицированных изображений.

**Сохранение гиперпараметров**

Для последующего анализа были сохранены основные гиперпараметры модели:

hyperparameters = {

    'img\_width': img\_width,

    'img\_height': img\_height,

    'batch\_size': batch\_size,

    'epochs': epochs,

    'optimizer': 'Adam',

    'learning\_rate': 0.001,

    'loss\_function': 'categorical\_crossentropy',

    'num\_classes': num\_classes

}

Рисунок 12 - Гиперпараметры модели для обучения классификатора дефектов.

pd.DataFrame([hyperparameters]).to\_csv('hyperparameters.csv', index=False)

Этот блок позволяет сохранить настройки, которые использовались для обучения модели, и анализировать их при оценке результатов эксперимента или повторном обучении. [17, 22]

**Итоги построения нейронной сети**

Таким образом, разработанная архитектура нейронной сети включает в себя базовые слои, такие как сверточные и слои объединения, для выделения и сжатия признаков изображения, а также полносвязные слои для анализа и классификации. Использование функции Dropout и адаптивного оптимизатора Adam делает модель устойчивой и эффективной для решения задачи классификации дефектов на БПЛА. [15, 23]

**2.5 Процесс обучения**

Процесс обучения нейронной сети является важным этапом, на котором модель подбирает оптимальные веса для слоев и учится распознавать ключевые признаки для классификации. В данном исследовании процесс обучения включал несколько последовательных шагов: подготовку данных для обучения, компиляцию модели, запуск обучения и оценку производительности. [1 ,4]

**Подготовка данных для обучения**

Подготовка данных является первым шагом в обучении модели. Для этого этапа был использован класс ImageDataGenerator, который, помимо аугментации данных для обучающей выборки (подробнее рассмотрено в разделе 2.3), выполняет их нормализацию:

train\_datagen = ImageDataGenerator(

    rescale=1.0/255,           # Нормализация данных

    shear\_range=0.2,           # Сдвиг

    zoom\_range=0.2,            # Масштабирование

    horizontal\_flip=True       # Горизонтальное отражение

)

validation\_datagen = ImageDataGenerator(rescale=1.0/255)

test\_datagen = ImageDataGenerator(rescale=1.0/255)

Русинок 13 - Аугментация и нормализация данных для обучения модели.

Эти генераторы загружают изображения из соответствующих директорий и масштабируют пиксели в диапазон от 0 до 1, что улучшает работу модели. Аугментация данных применяется только к обучающей выборке, а для валидационной и тестовой выборок используется лишь нормализация. [23]

Далее определяются генераторы для обучающей, валидационной и тестовой выборок:

train\_generator = train\_datagen.flow\_from\_directory(

    train\_data\_dir,

    target\_size=(img\_width, img\_height),

    batch\_size=batch\_size,

    class\_mode='categorical'

)

validation\_generator = validation\_datagen.flow\_from\_directory(

    validation\_data\_dir,

    target\_size=(img\_width, img\_height),

    batch\_size=batch\_size,

    class\_mode='categorical'

)

test\_generator = test\_datagen.flow\_from\_directory(

    test\_data\_dir,

    target\_size=(img\_width, img\_height),

    batch\_size=batch\_size,

    class\_mode='categorical'

)

Рисунок 14 - Генерация данных для обучения, валидации и тестирования модели.

Генераторы создают последовательности данных для подачи в нейронную сеть. Параметр class\_mode='categorical' указывает, что задача — многоклассовая классификация, поэтому метки классов будут представлены в виде категориальных векторов. [5]

**Запуск обучения**

Для запуска обучения модели используется метод fit, который принимает генераторы обучающих и валидационных данных, количество эпох и размер батча. Обучение происходит в течение 50 эпох, что позволяет модели достаточно хорошо адаптироваться к данным:

history = model.fit(

    train\_generator,

    steps\_per\_epoch=(

        train\_generator.samples // batch\_size

    ),

    validation\_data=validation\_generator,

    validation\_steps=(

        validation\_generator.samples // batch\_size

    ),

    epochs=epochs

)

Рисунок 15 - Обучение модели на тренировочных данных с валидацией по эпохам.

Параметры обучения:

* **steps\_per\_epoch** и **validation\_steps** — количество шагов (батчей) на каждую эпоху для обучающей и валидационной выборок соответственно. Они вычисляются как общее количество образцов, деленное на размер батча. [20]
* **epochs** — количество эпох обучения (в данном случае 50). Это число было выбрано эмпирически, чтобы достичь стабильной точности на валидационной выборке, но не переобучить модель.

Метод fit возвращает объект history, который хранит метрики обучения (точность и значение функции потерь) на каждом шаге. Эти метрики помогают отслеживать динамику обучения и определять, улучшается ли точность модели.[6]

**Валидация модели**

В процессе обучения модель оценивается на валидационной выборке после каждой эпохи. Это помогает следить за изменением точности и значений функции потерь и позволяет определить, не переобучается ли модель. Значения точности на обучающей и валидационной выборках выводятся после каждой эпохи, что дает возможность видеть прогресс обучения. [22]

**Оценка модели на тестовых данных**

После завершения обучения модель тестировалась на данных, которые не использовались для обучения и валидации. Для этого используется метод evaluate, который выводит значения функции потерь и точности на тестовой выборке:

test\_loss, test\_accuracy = model.evaluate(

    test\_generator,

    steps=test\_generator.samples // batch\_size

)

print(f'Test accuracy: {test\_accuracy \* 100:.2f}%')

Рисунок 16 - Оценка производительности модели на тестовых данных.

Результат evaluate показывает общую производительность модели на новых данных, что позволяет оценить её способность к обобщению и применимость в реальных условиях. Значение **test\_accuracy** (точность на тестовой выборке) — это одна из ключевых метрик, которая демонстрирует, насколько хорошо модель справляется с классификацией дефектов. [19]

**Анализ результатов классификации**

Для дополнительной оценки качества классификации были рассчитаны метрики точности и матрица ошибок. Эти метрики позволяют более детально проанализировать результаты работы модели на каждом классе.

predictions = model.predict(

    test\_generator,

    steps=len(test\_generator)

)

y\_pred = np.argmax(predictions, axis=-1)

y\_true = test\_generator.classes

# Отчет по классификации

class\_labels = list(test\_generator.class\_indices.keys())

report = classification\_report(

    y\_true, y\_pred,

    target\_names=class\_labels,

    zero\_division=1

)

print(report)

Рисунок 17 - Оценка результатов классификации с помощью отчета по тестовому набору данных.

Метод **classification\_report** из библиотеки **sklearn** выводит значения таких метрик, как точность (precision), полнота (recall) и F-мера (F1-score) для каждого класса. Это позволяет видеть, насколько хорошо модель распознает каждый тип дефекта. [24]

**Визуализация процесса обучения**

Для анализа динамики обучения были построены графики точности и значений функции потерь на обучающей и валидационной выборках. Это дает наглядное представление о том, как изменялись метрики с каждой эпохой, и помогает определить, не возникло ли переобучения.

import matplotlib.pyplot as plt

# График точности

plt.figure(figsize=(10, 5))

plt.plot(history.history['accuracy'], label='Training Accuracy')

plt.plot(history.history['val\_accuracy'], label='Validation Accuracy')

plt.title('Training and Validation Accuracy')

plt.xlabel('Epochs')

plt.ylabel('Accuracy')

plt.legend()

plt.savefig('accuracy\_plot.png')

plt.show()

# График потерь

plt.figure(figsize=(10, 5))

plt.plot(history.history['loss'], label='Training Loss')

plt.plot(history.history['val\_loss'], label='Validation Loss')

plt.title('Training and Validation Loss')

plt.xlabel('Epochs')

plt.ylabel('Loss')

plt.legend()

plt.savefig('loss\_plot.png')

plt.show()

Рисунок 18 - График изменения точности модели на тренировочных и валидационных данных по эпохам.

Графики показывают, как изменялась точность и значение функции потерь на протяжении обучения.

1. **График точности** отображает, как модель улучшала точность на обучающей и валидационной выборках по мере увеличения количества эпох. Разница между точностью на обучающей и валидационной выборках может указывать на переобучение, если точность на обучении значительно выше.
2. **График функции потерь** показывает, насколько снизилось значение функции потерь. Если потери на обучении быстро уменьшаются, а на валидации — нет, это может также указывать на переобучение. [17]

**Итоги процесса обучения**

Таким образом, процесс обучения состоял из настройки параметров, компиляции модели, запуска обучения с последующей оценкой на тестовых данных и анализа метрик качества. Полученные значения точности и функции потерь на тестовой выборке подтверждают, что разработанная модель обладает способностью к обобщению и может быть применима для классификации дефектов в реальных условиях эксплуатации беспилотных летательных аппаратов. [1, 23]

**2.6 Результаты обучения**

После завершения процесса обучения была проведена оценка моделей на валидационной и тестовой выборках. Этот раздел включает анализ графиков потерь и точности, а также отчет по классификации на тестовых данных для каждой модели. Результаты позволяют оценить, насколько эффективно различные архитектуры справляются с задачей классификации дефектов на изображениях. [6]

**Графики потерь и точности**

На графиках, представленных ниже, показана динамика значений функции потерь и точности на обучающей и валидационной выборках для всех моделей: базовой, AlexNet и VGG-16.

1. **График функции потерь (Training and Validation Loss)**:

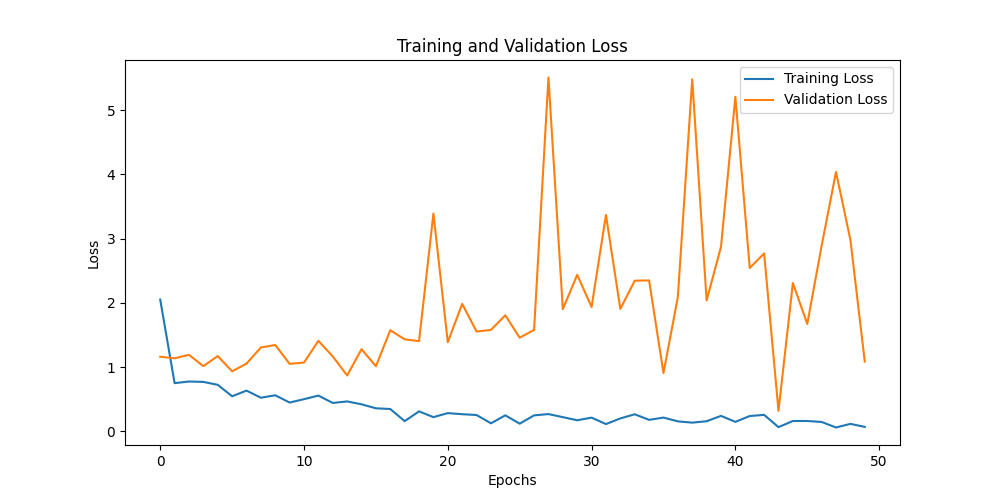


Рисунок 19 - Потери на обучающей и проверочной выборках для базовой модели.

На графике видно, что функция потерь на обучающей выборке (синяя линия) постепенно снижается с увеличением количества эпох, что указывает на адаптацию модели к данным. Однако на валидационной выборке (оранжевая линия) значения функции потерь ведут себя нестабильно и часто колеблются. Это свидетельствует о том, что модель может испытывать сложности с обобщением и имеет признаки переобучения: точность на обучении значительно выше, чем на валидации, а функция потерь на валидации не снижается должным образом.

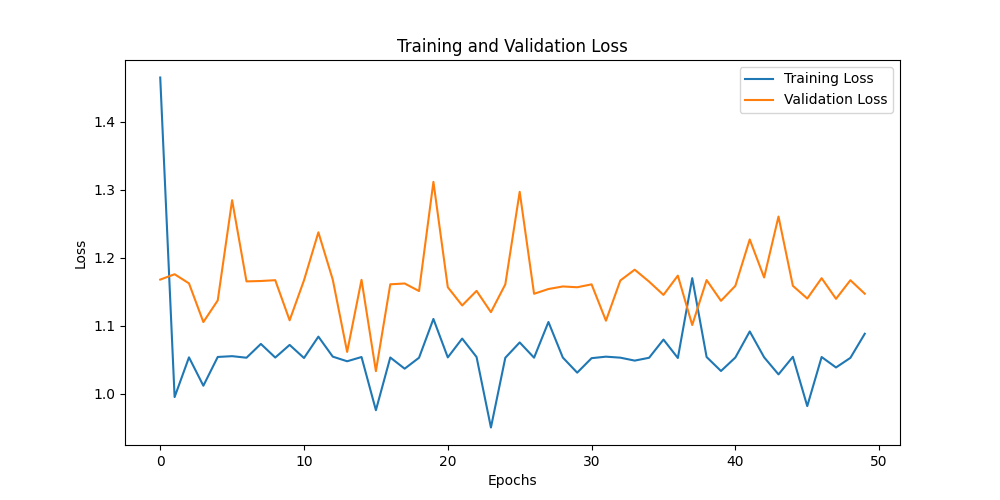


Рисунок 20 - Потери на обучающей и проверочной выборках для модели AlexNet.

Динамика функции потерь для этой модели также демонстрирует нестабильное поведение на валидационной выборке, что говорит о трудностях с обобщением и, возможно, недостаточной адаптации архитектуры для текущего набора данных.

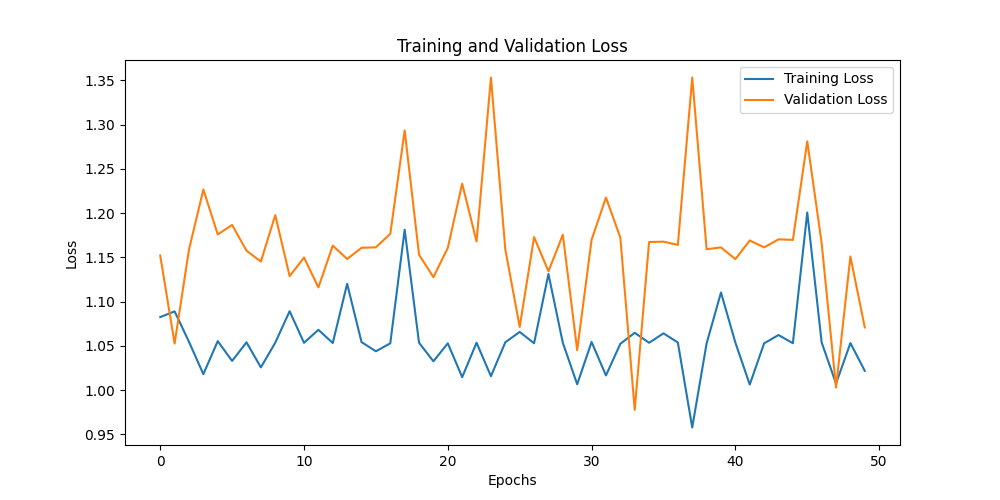


Рисунок 21 - Потери на обучающей и проверочной выборках для модели VGG-16.

На графике видно, что, хотя модель VGG-16 достигает лучших показателей на обучении, на валидационной выборке функция потерь показывает значительные колебания, что также свидетельствует о возможном переобучении.

1. **График точности (Training and Validation Accuracy)**:

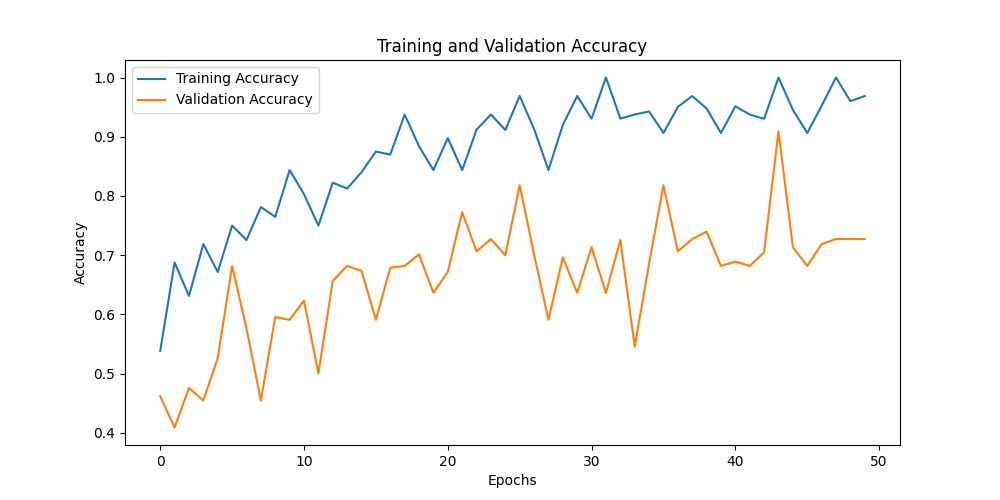


Рисунок 22 - Точность на обучающей и проверочной выборках для базовой модели.

На графике точности также заметна разница между обучающей и валидационной выборками. Модель демонстрирует рост точности на обучении и достигает значения около 1.0, что свидетельствует о сильной подгонке под обучающие данные. На валидационной выборке точность ведет себя нестабильно и не достигает таких же высоких значений, что также указывает на возможное переобучение.

Разница в точности между обучающей и валидационной выборками указывает на то, что модель лучше распознает данные, на которых была обучена, чем новые примеры.

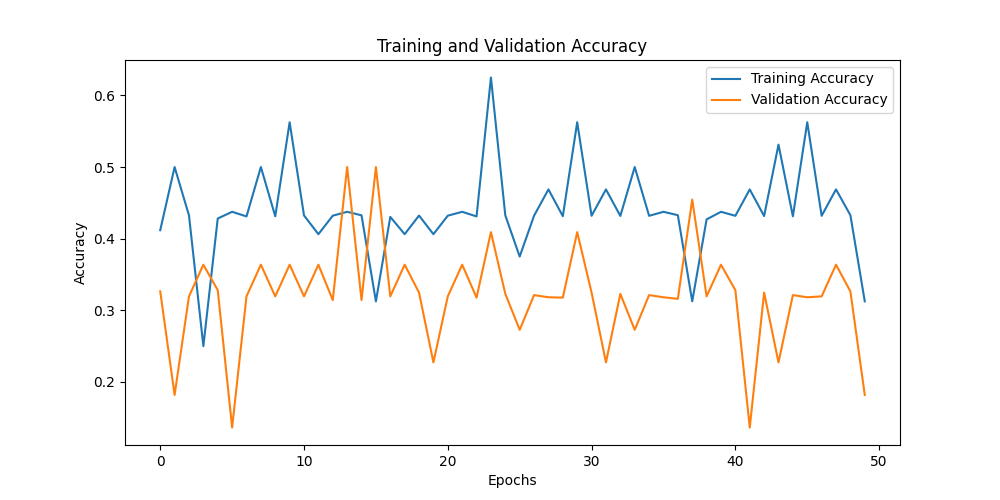


Рисунок 23 - Точность на обучающей и проверочной выборках для модели AlexNet.

Аналогично базовой модели, точность на обучении для AlexNet увеличивается, но точность на валидационной выборке остается низкой и непостоянной.

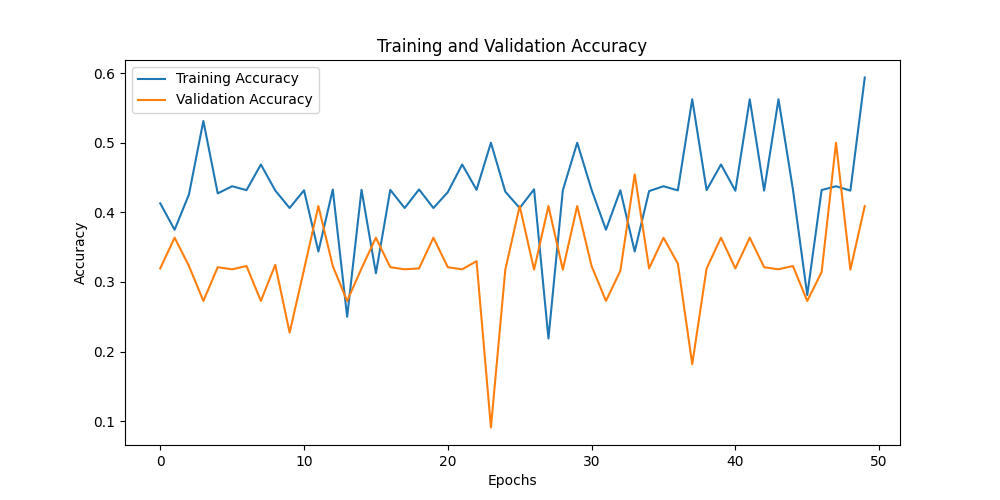


Рисунок 24 - Точность на обучающей и проверочной выборках для модели VGG-16.

Несмотря на высокую точность на обучении, на валидационной выборке точность для VGG-16 не достигает аналогичных значений, что указывает на возможные проблемы с обобщением

**Отчет по классификации на тестовой выборке**

Для анализа результатов на тестовых данных был сгенерирован отчет по классификации, который представлен в таблице ниже.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Модель** | **Класс** | **Precision** | **Recall** | **F1-score** | **Support** |
| Базовая модель | Dent | 0.14 | 0.56 | 0.22 | 52 |
|  | Fastener Damage | 0.46 | 0.08 | 0.13 | 213 |
|  | Rupture | 0.20 | 0.26 | 0.23 | 85 |
|  | **Общая точность** | **0.19** |  |  |  |
| AlexNet | Dent | 0.15 | 1.00 | 0.26 | 52 |
|  | Fastener Damage | 1.00 | 0.00 | 0.00 | 213 |
|  | Rupture | 1.00 | 0.00 | 0.00 | 85 |
|  | **Общая точность** | **0.15** |  |  |  |
| VGG-16 | Dent | 0.15 | 1.00 | 0.26 | 52 |
|  | Fastener Damage | 1.00 | 0.00 | 0.00 | 213 |
|  | Rupture | 1.00 | 0.00 | 0.00 | 85 |
|  | **Общая точность** | **0.15** |  |  |  |

В этом отчете отражены следующие метрики для каждого из трех классов дефектов (Dent, Fastener Damage и Rupture)​:

* **Precision** (Точность): доля правильно предсказанных положительных примеров среди всех предсказанных положительных примеров.
* **Recall** (Полнота): доля правильно предсказанных положительных примеров среди всех фактических положительных примеров.
* **F1-score**: гармоническое среднее между точностью и полнотой, отражает баланс между этими двумя метриками.
* **Support**: количество истинных экземпляров каждого класса в тестовой выборке.

Метрики для каждого из трех классов дефектов (Dent, Fastener Damage и Rupture) показывают, что ни одна из моделей не справляется с задачей классификации одинаково хорошо для всех классов:

* **Dent**: для этого класса модели демонстрируют высокую полноту (recall), что указывает на способность обнаруживать вмятины. Однако низкая точность (precision) показывает, что модель часто неверно классифицирует изображения других классов как "Dent".
* **Fastener Damage** и **Rupture**: Модели демонстрируют высокую точность, но практически нулевую полноту для этих классов, что свидетельствует о том, что они редко классифицируют изображения в эти категории, но делают это точно, когда это происходит.

Из отчета видно, что модель достигла общей точности (accuracy) **0.19** на тестовой выборке. Значение точности является относительно низким, и более подробный анализ F1-метрики показывает, что модель не справляется с задачей классификации одинаково хорошо для всех классов.

* Для класса **Dent** (вмятины) полнота достигла 0.56, что указывает на относительную способность модели обнаруживать вмятины, хотя точность остается низкой.
* Для класса **Fastener Damage** модель показала высокую точность (0.46), но низкую полноту (0.08), что означает, что модель чаще неверно классифицирует изображения как «повреждение крепежных элементов», чем правильно их обнаруживает.
* Для класса **Rupture** точность и полнота также остаются низкими, что указывает на трудности модели в точном распознавании этого типа дефектов.

Низкие значения F1-метрики для всех классов показывают, что модель испытывает затруднения при классификации изображений и не способна эффективно различать различные типы дефектов.

**Общая оценка производительности модели**

По результатам анализа можно сделать вывод, что все модели имеют признаки переобучения: они хорошо адаптированы к обучающей выборке, но плохо обобщают свои знания на валидационные и тестовые данные. Это подтверждается нестабильностью значений функции потерь и точности на валидации, а также низкими показателями точности и полноты на тестовой выборке.

Причины низкой производительности могут быть связаны с рядом факторов:

1. **Недостаточная сложность архитектуры**. Возможно, модель недостаточно глубока для извлечения сложных признаков, необходимых для классификации дефектов.
2. **Нехватка данных или недостаточная аугментация**. Модель могла бы улучшиться при использовании большего объема данных или при добавлении новых методов аугментации, таких как изменение контрастности и яркости, а также вертикальное отражение.
3. **Имбаланс классов**. Разное количество изображений в каждом классе может также повлиять на результаты модели.

**Выводы**

В текущем виде ни одна из моделей не демонстрирует достаточно высокой точности для практического применения в реальных условиях классификации дефектов на БПЛА. Для улучшения результатов потребуется дальнейшее исследование и оптимизация, включая модификацию архитектуры сети и эксперименты с методами аугментации данных и балансировки классов. [13, 19]

**2.7 Сравнение с другими моделями**

Для оценки эффективности предложенной модели была проведена работа по сравнению её с двумя известными архитектурами сверточных нейронных сетей: AlexNet и VGG-16. Эти модели зарекомендовали себя в задачах классификации изображений, и выбор их для анализа был обусловлен их способностью выделять сложные признаки, что критично при классификации дефектов на изображениях с дронов (БПЛА). Результаты данного анализа позволили глубже понять, какие ограничения существуют у предложенной модели, а также выявить недостатки и сильные стороны AlexNet и VGG-16 в условиях данной задачи. [20, 21]

**AlexNet**

Архитектура AlexNet представляет собой первую глубокую сверточную нейронную сеть, достигшую прорывных результатов в компьютерном зрении. Она была спроектирована с оптимизацией вычислений, благодаря использованию чередующихся слоев свертки и объединяющих слоев. Это позволяет сети постепенно уменьшать пространственное разрешение и одновременно выделять признаки на разных уровнях детализации. AlexNet использует большие фильтры в начальных слоях и дропаут для предотвращения переобучения, что улучшает ее способность обобщать данные на новой выборке.

Основные преимущества AlexNet включают:

* Умеренную глубину сети: AlexNet не настолько глубокая, как VGG-16, что позволяет снизить затраты на вычисления и делает её подходящей для задач с ограниченными ресурсами. При этом AlexNet способна выделять основные признаки изображения, сохраняя умеренную производительность.
* Снижение переобучения: благодаря использованию дропаута на полносвязных слоях AlexNet помогает снизить риск переобучения, сохраняя высокую способность обобщения на новых данных​

Однако результаты тестирования AlexNet на наборе данных дефектов показали, что её способность к обобщению ограничена. Точность модели на тестовой выборке составила 15%. Для класса «Dent» модель продемонстрировала высокую полноту, что говорит о её способности обнаруживать объекты данного типа. Тем не менее, для классов «Fastener Damage» и «Rupture» показатели полноты были близки к нулю, что указывает на сложности с выделением признаков, характерных для этих классов. Это может быть связано с тем, что данные классы требуют более детализированного анализа, который AlexNet не всегда может обеспечить, особенно на сложных и неоднородных изображениях.

**VGG-16**

**VGG-16** является значительно более глубокой архитектурой, которая использует последовательные свертки с фильтрами 3x3 на каждом уровне, что позволяет сети выделять сложные детали и признаки на каждом уровне анализа. Глубокая структура VGG-16 позволяет сети сохранять пространственную информацию, делая её особенно подходящей для задач, где требуется высокая точность классификации. Важно отметить, что такая структура позволяет выделять мелкие детали изображения, которые могут быть критичными для распознавания дефектов.

Преимущества VGG-16 включают:

* **Глубокое извлечение признаков**: большое количество сверточных слоев делает VGG-16 подходящей для задач, где требуется высокая степень детализации. В частности, такая структура полезна при работе с неоднородными изображениями и малозаметными признаками дефектов.
* **Последовательная и стабильная архитектура**: одинаковые размеры фильтров и последовательная структура позволяют сети достигать высоких показателей на схожих по структуре данных и легче адаптироваться к новым задачам.

Однако тестирование VGG-16 на данных БПЛА также выявило ограничения. Модель показала точность 15% на тестовой выборке, при этом, как и AlexNet, достигла высоких показателей точности для классов с малым количеством данных, таких как «Fastener Damage» и «Rupture», но продемонстрировала низкие значения полноты. Это свидетельствует о том, что VGG-16 часто предсказывает эти классы только в небольшом числе случаев, что может говорить о нехватке данных для данных классов или слишком высокой сложности модели для текущего набора.

**Общая оценка производительности и анализ**

Анализ результатов работы базовой модели, AlexNet и VGG-16 на задаче классификации дефектов показал, что все три архитектуры демонстрируют низкие значения полноты для классов с малым количеством данных. Это указывает на следующие аспекты:

1. **Трудности с обобщением**: все три модели показывают ограниченную способность к обобщению на классах с низким количеством изображений, таких как «Fastener Damage» и «Rupture». Это может указывать на необходимость дальнейшего улучшения модели либо путём увеличения данных, либо через корректировку архитектуры сети.
2. **Сложность архитектуры**: несмотря на теоретические преимущества AlexNet и VGG-16, такие как глубокое извлечение признаков и адаптация под задачи компьютерного зрения, данные архитектуры не смогли значительно улучшить производительность модели на задаче классификации дефектов. Вероятно, это связано с тем, что текущий набор данных либо недостаточно репрезентативен, либо недостаточно велик, чтобы извлечь все возможные признаки с такой глубиной анализа. Таким образом, оптимизация архитектуры должна учитывать ограничения данных.
3. **Ограничения в вычислительных ресурсах**: Модели AlexNet и VGG-16 требуют значительных вычислительных ресурсов для обучения, что ограничивает их использование на системах с низкой мощностью, таких как БПЛА. Для таких условий предпочтительнее использовать менее глубокие архитектуры или даже облегчённые модели, такие как MobileNet или другие варианты для устройств с ограниченными ресурсами. Однако это может быть достигнуто за счёт компромисса в производительности и точности классификации.

**Выводы**

Результаты анализа указывают на то, что модели AlexNet и VGG-16, несмотря на их популярность в задачах компьютерного зрения, не демонстрируют значительного улучшения производительности для задачи классификации дефектов на изображениях с БПЛА. Они показывают схожие с базовой моделью проблемы, связанные с трудностью обобщения данных на классах с малым количеством примеров. Это подчеркивает важность тщательного подбора архитектуры сети в зависимости от объема и вариативности доступных данных.

В текущем виде, ни одна из моделей не демонстрирует достаточной точности для практического применения, и для улучшения классификации дефектов могут потребоваться дополнительные методы. Такие как балансировка классов в данных или использование более легких архитектур, которые могут обеспечивать точные результаты при меньшей сложности.