МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ

Кафедра информационных систем управления

СВИСТУНОВА Ксения Игоревна

АНАЛИЗ ПРИЗНАКОВ ЗАБОЛЕВАНИЯ НА ИЗОБРАЖЕНИЯХ ГЛАЗНОГО ДНА МЕТОДАМИ МАШИННОГО ОБУЧЕНИЯ

Дипломная работа

Научный руководитель:
Доктор технических наук,
профессор С.В.Абламейко

Допущена к защите
«___»_____2025 г.
Зав. кафедрой информационных систем управления
доктор технических наук, доцент А. М. Недзьведь

Минск 2025

ОГЛАВЛЕНИЕ

РЕФЕРАТ

Дипломная работа: 46 с., 13 рисунков, 4 таблицы, 1 приложение, 13 источников.

Ключевые слова: СВЕРТОЧНЫЕ НЕЙРОННЫЕ СЕТИ, КЛАССИФИКАЦИЯ, ИЗОБРАЖЕНИЯ ГЛАЗНОГО ДНА, БАЛАНСИРОВКА КЛАССОВ, АУГМЕНТАЦИЯ, СЕГМЕНТАЦИЯ, ВЕБ-ПРИЛОЖЕНИЯ, CONFUSION MATRIX, EFFICIENTNET, YOLO, U-NET.

Объект исследования: принципы распознавания образов, свёрточные нейронные сети, модели классификации и сегментации.

Цель работы: исследование и применение нейронных сетей для анализа признаков заболевания на изображениях глазного дна методами машинного обучения, проведение сегментации изображений для выделения наиболее информативных областей глазного дна, создание веб-приложения, применимого для анализа заболеваний.

Методы исследования: теории глубокого обучения, технологии и методы машинного обучения, распознавания образов, искусственного интеллекта, проектирование.

Результаты: выбраны нейронные сети на основе анализа наилучших для классификации и сегментации, произведен анализ эффективности их применения, рассмотрены результаты, реализован практический модуль.

Область применения: исследование медицинских изображений, офтальмология.

РЭФЕРАТ

Дыпломная праца: 46 с., 13 малюнкаў, 4 табліцы, 1 дадатак, 13 крыніц.

Ключавыя словы: СВЕРТКАВЫЯ НЕЙРОННЫЯ СЕТКІ, КЛАСІФІКАЦЫЯ, ВЫЯВЫ ВОЧНАГА ДНА, БАЛАНСАВАННЕ КЛАСАЎ, АЎГМЕНТАЦЫЯ, СЕГМЕНТАЦЫЯ, EFFICIENTNET, YOLO, CONFUSION MATRIX, U-NET.

Аб'ект даследавання: прынцыпы распазнання выяў, сверткавыя нейронныя сеткі, мадэлі класіфікацыі і сегментацыі.

Мэта працы: даследаванне і прымяненне нейронных сетак для аналізу прыкмет захворвання на выявах дна вока метадамі машыннага навучання, правядзенне сегментацыі выяў для вылучэння найбольш інфарматыўных абласцей дна вока, стварэнне вэб-дадатку, які можа выкарыстовывацца для аналізу захворванняў.

Метады даследавання: тэорыі глыбокага навучання, тэхналогіі і метады машыннага навучання, распазнання вобразаў, штучнага інтэлекту, праектаванне.

Вынікі: выбраны нейронныя сеткі на аснове аналізу найлепшых для класіфікацыі і сегментацыі, праведзены аналіз эфектыўнасці іх выкарыстання, прадстаўлены вынікі, рэалізаваны практычны модуль.

Вобласць ужывання: даследаванне медыцынскіх выяў, афтальмалогія.

SUMMARY

Diploma work: 46 pp., 13 figures, 4 tables, 1 appendix, 13 sources.

Keywords: CONVOLUTIONAL NEURAL NETWORKS, CLASSIFICATION, EYE FUNDUS IMAGES, CLASS BALANCING, AUGMENTATION, SEGMENTATION, WEB APPLICATIONS, EFFICIENTNET, YOLO, CONFUSION MATRIX, U-NET.

Object of study: principles of pattern recognition, convolutional neural networks, classification and segmentation models.

Objective: research and application of neural networks for analyzing disease signs on eye fundus images by machine learning methods, performing image segmentation to highlight the most informative areas of the eye fundus, creating a web application usable for disease analysis.

Research methods: theories of deep learning, machine learning technologies and methods, pattern recognition, artificial intelligence, design.

Results: based on the analysis of the best for classification and segmentation were selected neural networks, was analyzed the efficiency of their application, provided the results, was implemented practical module.

Scope: medical imaging research, ophthalmology.

ВВЕДЕНИЕ

С проблемами болезней глаз сталкивается практически каждый человек. Под болезнью глаз подразумевается любое состояние или расстройство, при котором нарушается способность глаза правильно работать, и оказывается вредное воздействие на остроту зрения. На сегодняшний день они являются основной причиной слепоты у людей. Своевременная диагностика позволяет предотвратить прогрессирование заболевания за счет оперативно начатого лечения.

В анализе биомедицинских изображений все более распространенными становятся алгоритмы глубокого обучения. Модели, основанные на глубоком обучении, хорошо справляются с различными задачами, такими как обнаружение объектов, классификация медицинских изображений и обнаружение заболеваний. В связи с развитием технологий машинного обучения появилась возможность для разработки автоматизированной системы для обнаружения заболеваний глаза по снимкам глазного дна. Благодаря такой системе появится возможность выявления признаков заболеваний по изображениям, что позволит осуществлять анализ и выявлять различные нарушения.

В связи с этим будем проводить исследования для анализа различных аспектов визуализации глазного дна для корректной диагностики заболеваний глаз и выявления признаков.

Целью данной работы является исследование и применение нейронных сетей для анализа признаков заболевания на изображениях глазного дна методами машинного обучения, а также проведение сегментации изображений для выделения наиболее значимых и информативных областей глазного дна и создание веб-приложения, применимого для анализа заболеваний.

ГЛАВА 1
СИМПТОМЫ ЗАБОЛЕВАНИЙ ГЛАЗА НА ИЗОБРАЖЕНИЯХ ГЛАЗНОГО ДНА

1.1 Методы исследования глаза

Глаз является одним из самых сложных органов человеческого тела, поэтому необходимо применение современных технологий и методов для его лучшего изучения. Для выявления заболеваний используются различные способы визуализации. Они позволяют выявлять заболевания на ранних стадиях, и, в то же время, отслеживать их прогресс и оценивать эффективность лечения.

На сегодняшний день существует широкий спектр инструментов для исследования глаза. Многие из них предоставляют детализированные изображения, необходимые офтальмологам для точной диагностики и мониторинга. Ниже представлены методы исследования глаза [1]:

1. Исследование глаза с помощью фундус-камеры (снимки глазного дна);
2. Оптическая когерентная томография (ОКТ);
3. Ультразвуковое исследование глаза (УЗИ);
4. Кератотопография;
5. Пахиметрия (толщина роговицы);
6. МРТ и КТ орбиты глаза.

Рассмотрим некоторые из них более подробно. Начнем с рассмотрения снимков глазного дна. Цифровые фотографии глазного дна получают с помощью фундус-камеры, у которой существует несколько преимуществ в сравнении с другими методами исследования. Например, при использовании фундус-камеры нет необходимости в предварительном расширении зрачка. Благодаря этому упрощается и ускоряется обследование состояния глаза. Также не требуется введение контрастного вещества. Использование фундус-камеры дает возможность собрать архив изображений глазного дна, что в дальнейшем позволит оценить динамику состояний сетчатки и глазного нерва, оценить эффективность лечения, а также скорректировать курс лечения.

Фотография глазного дна играет важную роль в диагностике и мониторинге широкого спектра заболеваний глаз, таких как диабетическая ретинопатия, дегенерация желтого пятна, глаукома, заболевания сосудов сетчатки, отслойка сетчатки.

Еще одним наиболее распространенным методом исследования является оптическая когерентная томография (ОКТ) – современный неинвазивный

метод визуализации, который позволяет получать детализированные изображения внутренних структур глаза. ОКТ основана на использовании оптического излучения ближнего инфракрасного диапазона для исследования тканей, на интерференции света, что позволяет визуализировать тонкие слои тканей [1]. ОКТ позволяет изучать слои сетчатки и выявлять патологические изменения, такие как возрастная макулярная дегенерация, диабетическая ретинопатия и глаукома. Метод помогает оценить состояние зрительного нерва, что важно при глаукоме и других нейродегенеративных заболеваниях.

В данной работе будем рассматривать снимки глазного дна полученные с помощью фундус-камеры.

1.2 Признаки заболеваний

Снимки глазного дна позволяют неинвазивно визуализировать внутренние структуры глаза, включая сетчатку, диск зрительного нерва, макулу и задний полюс. Это имеет большое значение в выявлении и мониторинге многочисленных заболеваний глаз. Для лучшего понимания заболеваний и их признаков рассмотрим основную информацию о каждом заболевании и опишем основные патологические признаки распространенных заболеваний глаз, которые проявляются на изображениях глазного дна.

Первое заболевание, которое будем рассматривать – миопия. Патологическая миопия обычно связана с осевым удлинением глазного яблока, что приводит к механическому растяжению и истончению сетчатки. На изображениях глазного дна у пациентов с миопией часто наблюдается перипапиллярная атрофия, мозаичный вид глазного дна, наклонные диски зрительных нервов и стафиломы. Данные особенности отражают структурную деформацию и прогрессирующую дегенерацию заднего сегмента, что может предрасполагать глаз к хориоретинальной атрофии и отслоению сетчатки.

Гипертоническая ретинопатия возникает из-за хронического повышенного артериального давления. Она проявляется различными микрососудистыми изменениями. Признаками заболевания на снимках глазного дна являются: генерализованное и очаговое сужение артериол, артериовенозные надрезы, помутнение стенок артериол, кровоизлияния в форме пламени, ватные пятна и твердые экссудаты. В тяжелых случаях может возникнуть отек диска зрительного нерва и образование звездочек в макулярной области, что указывает на злокачественную гипертонию, которая требует немедленного вмешательства.

Диабетическая ретинопатия (ДР) является одной из наиболее распространенных причин слепоты во всем мире. Ее отличительными чертами

на снимках глазного дна являются микроаневризмы, интраретинальные кровоизлияния (типа точек и пятен), твердые экссудаты, ватные пятна и отек сетчатки. Пролиферативная диабетическая ретинопатия (ПДР) может проявляться неоваскуляризацией на диске зрительного нерва или в других местах сетчатки, кровоизлиянием в стекловидное тело и тракционной отслойкой сетчатки. Диабетический макулярный отек, характеризующийся утолщением сетчатки в макулярной области, является основной причиной потери зрения при ДР [5].

Глаукома характеризуется прогрессирующей оптической нейропатией и потерей ганглиозных клеток сетчатки с соответствующими изменениями, видимыми на фотографиях глазного дна. К ним относятся повышенное соотношение чаши к диску (C/D), истончение или выемка нейроретинального обода, перипапиллярная атрофия и кровоизлияния в диск зрительного нерва (особенно при глаукоме с нормальным давлением). На поздних стадиях может наблюдаться экскавация головки зрительного нерва. Важное значение для обнаружения глаукомы имеет оценка соотношения чаши к диску, а также оценка асимметрии между глазами [9].

Катаракта в первую очередь поражает хрусталик. Она лучше всего визуализируется с помощью биомикроскопии с щелевой лампой, однако может иметь косвенные эффекты на снимках глазного дна. Например, помутнение хрусталика приводит к снижению контрастности изображения, размытости и снижению видимости структур сетчатки. На снимках глазного дна это выглядит как генерализованная дымка, особенно в красном канале, что может усложнить оценку сетчатки и повлиять на точность автоматизированного анализа [1].

Возрастная макулярная дегенерация (ВМД) является дегенеративным заболеванием центральной сетчатки. Данное заболевание подразделяется на сухую (неэкссудативную) и влажную (экссудативную) формы. Ранние признаки возрастной макулярной дегенерации на снимках глазного дна включают друзы (желтоватые внеклеточные отложения под сетчаткой), пигментные изменения и географическую атрофию. При неоваскулярной ВМД могут наблюдаться субретинальное кровоизлияние, накопление жидкости и хориоидальные неоваскулярные мембраны. Данные проявления часто приводят к потере центрального зрения. Они идентифицируются с помощью визуализации с высоким разрешением глазного дна.

Специфический вид и признаки этих патологий на фотографиях глазного дна составляют основу автоматизированных диагностических алгоритмов. Осуществление сегментации и локализация наиболее важных, для

каждого конкретного заболевания, анатомических структур впоследствии позволят моделям сосредоточиться на наиболее информативных областях [9].

1.3 Постановка задачи

В данной работе были поставлены следующие задачи для выполнения исследования.

Для проведения качественного исследования, необходимо на каждом этапе подобрать оптимальные средства, с помощью которых будет осуществляться достижение цели работы. На первом этапе необходимо подобрать набор данных, удовлетворяющий запросам нашей задачи. Для этого ознакомимся с датасетами, находящимися в открытом доступе, и выберем наиболее подходящий. После чего необходимо подробно ознакомиться со строением глазного дна и основными особенностями заболеваний, что даст понимание о наиболее важных аспектах в анализе признаков заболеваний.

Второй этап включает анализ существующих нейронных сетей и подбор оптимальных для решения задачи мультиклассовой классификации. Затем необходимо осуществить процесс обучения выбранных сетей, предварительно подготовив данные из датасета.

На третьем этапе требуется проанализировать полученные результаты и на основе их сделать вывод о возможностях модели. Данный этап предусматривает рассмотрение вариантов для улучшения модели.

На четвертом этапе требуется рассмотреть подходы к сегментации объектов на изображениях и создать модель для сегментации значимых областей глазного дна.

Для демонстрации работы моделей необходимо создать веб-приложение, которое позволит пользователю, имеющему снимки глазного дна, проверить их на наличие заболевания, а также при необходимости осуществить выделение наиболее информативных зон для дальнейшего исследования. Создание веб-приложения дает возможность практического применения данной модели.

1.4 Выводы

В данной главе были рассмотрены виды изображений, которые используют для определения болезней глаза человека, было принято решение работать с изображениями, полученными фундус-камерой, в связи с рядом преимуществ в получении данных изображений, а также их информативностью.

Были рассмотрены болезни и их особенности, на которые стоит ориентироваться при нахождении признаков заболеваний на снимках глазного дна. Знание этих особенностей позволит сделать выводы о подходах к изучению снимков.

Также были сформулированы задачи для последовательного осуществления исследования.

ГЛАВА 2
ВЫБОР НЕЙРОННОЙ СЕТИ И МЕТОДЫ ПРЕДОБРАБОТКИ ДАННЫХ

2.1 Нейронная сеть EfficientNet и ее архитектура

Первоначально ознакомимся с EfficientNet – семейством сверточных нейронных сетей (CNN), которое будем применять в исследовании. Цель данного семейства сетей заключается в достижении высокой производительности с меньшими вычислительными ресурсами по сравнению с более ранними архитектурами.

Данная сеть была представлена в статье 2019 года «EfficientNet: переосмысление масштабирования модели для сверточных нейронных сетей». В ней используется новый метод масштабирования, который получил название составной коэффициент. Он масштабирует модели простым, но эффективным способом: используется составное масштабирование, которое равномерно масштабирует ширину, глубину и разрешение с определенным фиксированным набором коэффициентов. Из-за этого модель может легко адаптироваться к различным вычислительным ограничениям, при этом сохраняет свою производительность в задачах.

Используя данный метод масштабирования, Mingxing Tan и Quoc V. Ле из Google Research разработали семь моделей различных размеров, которые превосходили самую современную точность большинства сверточных нейронных сетей [10].

Архитектура EfficientNet представлена набором готовых к использованию моделей. Выбор зависит от требуемой точности, доступных ресурсов для обучения и разрешения входных изображений. Модели маркируются от B0 (самая простая) до B7 (самая сильная). В таблице 2.1 представлены входные разрешения изображений для различных версий EfficientNet.

Таблица 2.1 – Принимаемые входные разрешения изображений

Имя модели	Входное разрешение изображений
EfficientNetB0	224x224
EfficientNetB1	240x240
EfficientNetB2	260x260

Продолжение таблицы 2.1

EfficientNetB3	300x300
EfficientNetB4	380x380
EfficientNetB5	456x456
EfficientNetB6	528x528
EfficientNetB7	600x600

Было исследовано влияние каждой стратегии масштабирования на эффективность и производительность модели перед созданием сложного метода масштабирования и сделан вывод, что, хотя масштабирование одного измерения может помочь улучшить производительность модели, лучший способ повысить производительность модели в целом – это сбалансировать масштаб во всех трех измерениях (ширина, глубина и разрешение изображения) с учетом доступных изменных ресурсов.

На приведенных ниже изображениях показаны различные методы масштабирования: базовый уровень (исходная сеть без масштабирования), масштабирование ширины (увеличение количества каналов в каждом слое), масштабирование глубины (увеличение количества слоев), масштабирование разрешения (увеличение разрешения входного изображения).

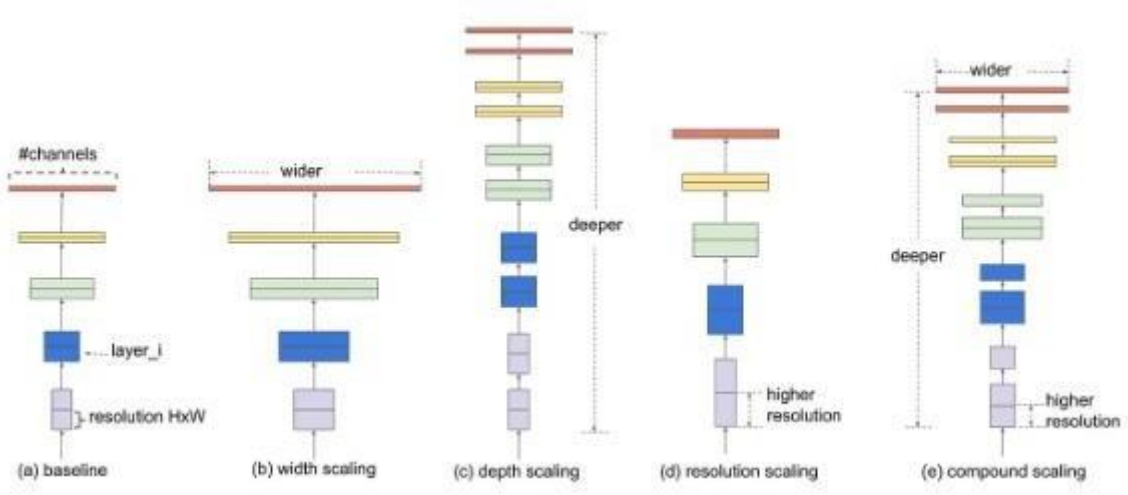

Рисунок 2.1 – Различные методы масштабирования против сложного масштабирования

Составное масштабирование – это одновременное увеличение ширины, глубины и разрешения в соответствии с формулой сложного масштабирования.

EfficientNet использует разделяемые свертки по глубине для снижения вычислительной сложности без ущерба для представительных возможностей. Это достигается путем разделения нормальной свертки на две части: свертка по глубине – применяет один фильтр к каждому входному каналу и точковая свертка, когда агрегированные функции из разных каналов. Это делает сеть более эффективной, требуя меньше вычислений и параметров.

EfficientNet использует перевернутые остаточные блоки для оптимизации использования ресурсов. Они начинаются с легкой свертки по глубине, за которой следует точечное расширение и еще одна глубинная свертка. Кроме того, включены операции сжатия и возбуждения. Они улучшают представление признаков путем перекалибровки канальных ответов.

EfficientNet включает в себя эффективные механизмы внимания, такие как блоки сжатия и возбуждения (SE), для улучшения представления особенностей. Данные блоки выборочно усиливают информативные функции, изучая веса внимания по каналу. Это повышает дискриминационные возможности сети при минимизации вычислительных расходов [10].

2.2 Нейронная сеть YOLO

YOLO (You Only Look Once) – это модель для обнаружения объектов и сегментации изображений. Она была разработана авторами Джозефом Редмоном и Али Фархади в Университете Вашингтона. Выпущенная в 2015 году, YOLO завоевала популярность благодаря своей высокой скорости и точности. В данной работе будем рассматривать версию YOLOv12.

На рисунке 2.2 сравнивается производительность YOLOv12 с различными моделями серии YOLO (от YOLOv9 до YOLOv12). В ней показана оценка производительности на разных GPU в различных масштабах моделей, от Tiny/Nano до Extra Large.

YOLOv12 – архитектура, ориентированная на внимание, с подходами отличными от предыдущих моделей YOLO. При этом сохраняет необходимую для многих приложений скорость обработки в режиме реального времени.

Основные характеристики YOLOv12 включают механизм зонального внимания, улучшенный модуль агрегирования признаков на основе ELAN, оптимизированную архитектуру внимания, всестороннюю поддержку задач и повышенную эффективность.

Model	Scale	FLOPs (G)	RTX 3080	A5000	A6000
YOLOv9 [58]	T	8.2	2.4/1.5	2.4/1.6	2.3/1.7
	S	26.4	3.7/1.9	3.4/2.0	3.5/1.9
	M	76.3	6.5/2.8	5.5/2.6	5.2/2.6
	C	102.1	8.0/2.9	6.4/2.7	6.0/2.7
	E	189.0	17.2/6.7	14.2/6.3	13.1/5.9
YOLOv10 [53]	N	6.7	1.6/1.0	1.6/1.0	1.6/1.0
	S	21.6	2.8/1.4	2.4/1.4	2.4/1.3
	M	59.1	5.7/2.5	4.5/2.4	4.2/2.2
	B	92.0	6.8/2.9	5.5/2.6	5.2/2.8
YOLOv11 [28]	N	6.5	1.6/1.0	1.6/1.0	1.5/0.9
	S	21.5	2.8/1.3	2.4/1.4	2.4/1.3
	M	68.0	5.6/2.3	4.5/2.2	4.4/2.1
	L	86.9	7.4/3.0	5.9/2.7	5.8/2.7
	X	194.9	15.2/5.3	10.7/4.7	9.1/4.0
YOLOv12	N	6.5	1.7/1.1	1.7/1.0	1.7/1.1
	S	21.4	2.9/1.5	2.5/1.5	2.5/1.4
	M	67.5	5.8/1.5	4.6/2.4	4.4/2.2
	L	88.9	7.9/3.3	6.2/3.1	6.0/3.0
	X	199.0	15.6/5.6	11.0/5.2	9.5/4.4

Рисунок 2.2 – Сравнение производительности YOLOv12 с различными моделями серии YOLO

Рассмотрим эти характеристики подробнее. В механизме зонального внимания организован новый подход к самовниманию, который используется специально для обработки больших рецептивных полей. Для этого карты признаков делятся на N равных по размеру областей по вертикали или горизонтали. По умолчанию N установлено равным 4. При этом избегаются сложные операции, и осуществляется сохранение большого рецептивного поля, что приводит к снижению вычислительных затрат в сравнении со стандартным самовниманием [13].

В [13] описывается, что «R-ELAN представляет:

1. остаточные связи на уровне блоков с масштабированием (аналогично масштабированию слоев);

2. переработанный метод агрегирования признаков, создающий структуру, подобную узкому месту».

Также в YOLO12 для достижения большей эффективности оптимизируется механизм внимания. Обеспечивается совместимость YOLO12 с фреймворком YOLO. Например, минимизация накладных расходов на доступ к памяти происходит за счет использования FlashAttention, и быстрота и чистота модели обеспечивается через удаление позиционного кодирования. В дополнение в YOLO12 осуществляют регулировку соотношения MLP,

благодаря чему, обеспечивается лучший баланс вычислений между слоями внимания и feed-forward.

В YOLO12 стали использовать там, где есть необходимость, добавление к механизму внимания сепарабельной свертки, которая позволяет осуществлять неявное кодирование позиционной информации.

Преимуществом YOLO12 является достижение высокой точности с использованием меньшего количества параметров в сравнении с предыдущими моделями. Поэтому создателями был сделан вывод о том, что она демонстрирует улучшенный баланс между скоростью и точностью.

Данная сеть поддерживает решение задач компьютерного зрения, таких как ориентированное обнаружение объектов, сегментация, классификация и другие.

2.3 Подготовка данных для обучения нейронных сетей

2.3.1 Проблема несбалансированности данных

При решении задачи классификации для каких-то классов может быть собрано больше данных. Такой неравномерный баланс между классами приводит к тому, что модель обучения основывается в большей степени на данных мажоритарных классов, чем на данных миноритарных классов. Для того чтобы улучшить модель, можно сбалансировать классы [7].

Если не решать проблему несбалансированных данных, то модель будет перегружена большими классами и из-за этого будет плохо обобщаться на другие классы. Будут теряться свойства редких классов и наблюдаться нехватка примеров таких классов, что приведет к неправильной их классификации. Это особенно критично для задач, где точности классификации необходимо достичь для всех классов, а не только для наиболее представленных.

Следующие проблемы возникают при работе с несбалансированными наборами данных: игнорирование редких классов, сосредоточенность на мажоритарных классах, некорректные значения оценок метрик, которые могут скрывать проблемы с определением меньших классов.

2.3.2 Влияние балансировки данных

Выполняя балансировку классов для данных, можем получить модель, в большей степени ориентированную на признаки и основанную на данных более редкого класса. Балансировка также может влиять на модель следующим образом:

1. может получиться более высокое значение F1-меры – это гармоническая средняя между точностью и полнотой. Она позволяет учесть и долю правильно классифицированных положительных примеров, и способность модели обнаруживать положительные примеры. Более высокое значение F1-меры получается за счет увеличения веса миноритарного класса;

2. немного снизить общий показатель точности;

3. получается более информативная модель за счет того, что она ориентируется на признаки и на то, как выделить отдельные классы [7].

Однако для небольших наборов данных балансировка может привести к потере признаков. При изменении пропорций в наборах данных может быть потеряна часть информации, что может искажать прогнозы модели.

2.3.3 Методы решения проблемы несбалансированных данных

Для того чтобы сбалансировать данные в классах, сначала необходимо определить, какой баланс является идеальным для конкретного бизнес-сценария. Он может быть где-то от 80/20 до 50/50. Балансировка должна быть достаточной, чтобы получить то, что нужно, поскольку чрезмерная балансировка классов может привести к переобучению модели. Затем необходимо проверить модель с использованием вручную отложенных данных.

Наиболее распространенными методами балансировки классов являются аугментация данных и сокращение количества объектов мажоритарного класса (undersampling), увеличение выборки миноритарных классов(oversampling), алгоритмические решения, методы ансамблей.

Сокращение количества объектов больших классов осуществляется путем случайного исключения данных этих классов.

На рисунке 2.3 продемонстрирован процесс выбора элементов мажоритарного класса из исходного набора данных.

Однако методы, основывающиеся на уменьшении объектов большего класса, могут привести к ухудшению качества модели, а не к ее улучшению. Основной причиной этого является вероятность недообучения модели из-за уменьшения количества данных для ее обучения. Однако этот метод полезен в использовании в комбинации с другими методами решения проблемы несбалансированных данных.

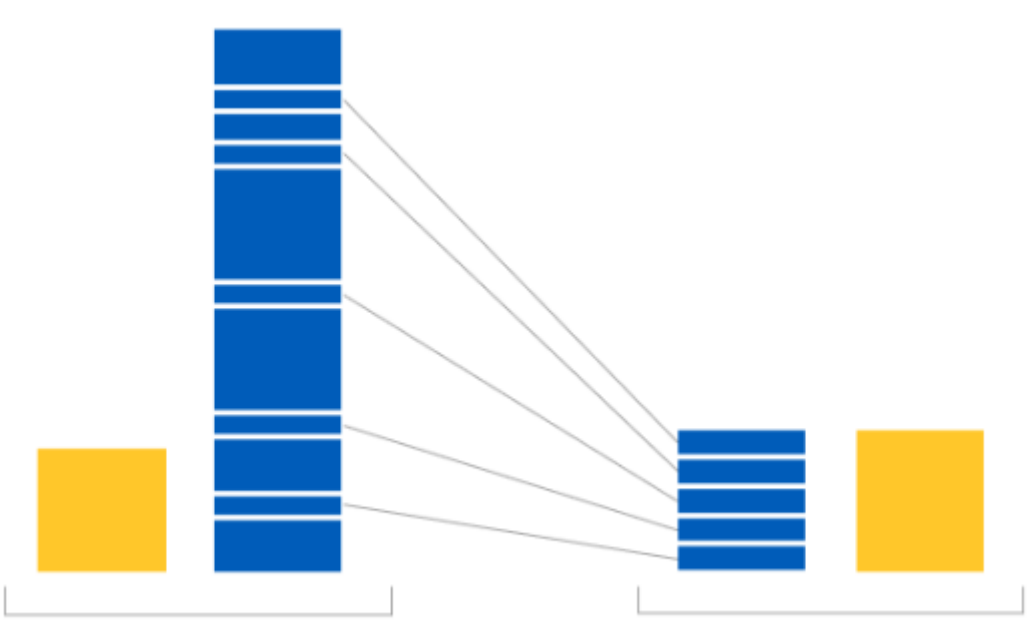

Рисунок 2.3 – Выборка данных из мажоритарного класса (синий) для выравнивания баланса с миноритарным классом (желтый)

Увеличение выборки меньшего класса актуально для небольших наборов данных. В таких наборах использование уменьшения количества данных большего класса может привести к потере информации, и в следствии этого к ухудшению точности классификации. Данный метод может осуществляться через копирование случайных объектов миноритарного класса или создание синтетических данных, используя алгоритмы как SMOTE, которые создают новые синтетические примеры данных, используя и немного изменяя уже существующие примеры миноритарного класса. Использование SMOTE делает данные более разнообразными и уменьшает риск переобучения.

Используются также и алгоритмические решения. Одним из них, может являться фокусирование на более сложных примерах. В таком случае применяют специальную функцию потерь – focal loss. При ее использовании делается упор на более трудные примеры, в то время как значение легко классифицируемых снижается.

Также часто применяют метод взвешивания классов. Он заключается в том, что большее внимание уделяется ошибкам менее представленных классов. В таком случае, в качестве функции потерь используют взвешенную кросс-энтропию, в которой коэффициент для каждого класса обратно пропорционален его распространённости в данных.

Если говорить об ансаблевых методах, то наиболее распространенными являются bagging, boosting. Можно комбинировать модели, обученные на

различных подвыборках, что помогает компенсировать дисбаланс данных и улучшает обобщающую способность модели.

Стоит помнить, что для каждой задачи необходимо подбирать оптимальную именно для нее стратегию и метод балансировки. Некоторые задачи не требуют равенства выборок классов для достижения высоких значений точности. Правильный подбор способа балансировки положительно сказывается на качестве обучения моделей.

2.3.4 Аугментация данных для балансировки классов

Аугментация данных – это процесс искусственного генерирования новых данных на основе существующих, который используется преимущественно для обучения моделей машинного обучения, так как для получения высоких результатов при обучении необходимы большие и разнообразные наборы данных. Получить такие наборы – трудоемкий процесс из-за различных ограничений, таких как несогласованность, нормативные требования и другие. Аугментация данных позволяет искусственно увеличить исходный набор данных, внося в исходные данные изменения с помощью различных преобразований, таких как повороты, сдвиги, изменение размеров, изменение яркости и контраста снимка, добавление шума на изображение, отражение изображений по горизонтали или вертикали, и другие [3].

Аугментация данных широко используется при работе с биомедицинскими изображениями, что обусловлено несколькими причинами. Она играет важную роль в улучшении диагностических моделей классификации и выявления заболеваний. Создавая дополнительные изображения, мы получаем большее количество обучающих данных, что имеет существенное значение для редких заболеваний, так как вариативность для них может быть сильно ограничена. Также наиболее проблематичными, с точки зрения исследований, в медицине являются строгие требования о конфиденциальности данных. В связи с этим, аугментация предоставляет нам возможность создавать синтетические данные, которые могут быть использованы для исследований, не нарушая конфиденциальности данных пациентов.

2.3.5 Аугментация на лету

Изначально, при аугментации данных использовался следующий механизм: данные искажались один раз перед обучением, добавлялись к имеющимся обучающим данным и сохранялись на жестком диске, что имело свои недостатки. Из-за этого требовалось гораздо больше места на диске для хранения подготовленных данных. Также вариативность данных, несмотря на проведенную аугментацию, продолжала быть ограничена. Поэтому возникла

необходимость в создании механизма, разрешающего эти проблемы. Им стала аугментация на лету, которая подразумевает, что аугментация происходит во время обучения модели, то есть она встроена в процесс обучения. Рассмотрим, как она осуществляется.

Искажение части данных происходит во время обучения между итерациями, из-за чего нейронная сеть на каждом этапе видит новые данные. При этом аугментированные данные не увеличивают исходный размер выборки для обучения, находящийся на жестком диске.

Для оптимизации процесса обучения необходимо было предусмотреть способы избежания простоев. При аугментации данных на лету используют распараллеливание, то есть обучение сети осуществляется на GPU, в то время как на CPU происходит аугментация данных [3].

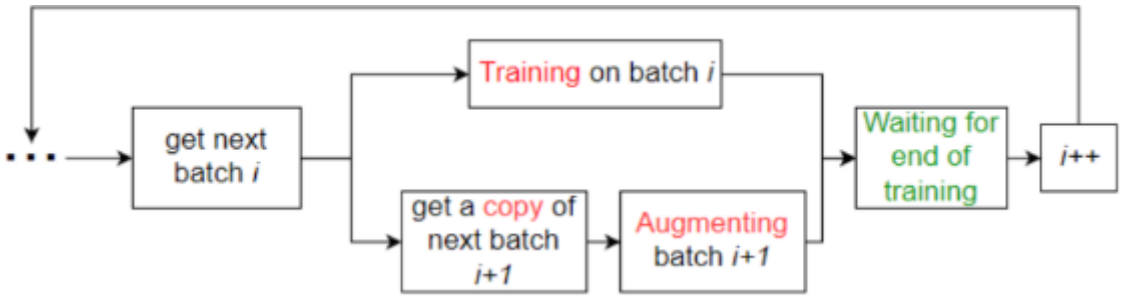

Рисунок 2.4 – Блок-схема работы аугментации на лету

Основным преимуществом аугментации на лету является то, что за время обучения модель может видеть все возможные комбинации заданных преобразований. В связи с этим, полученная в ходе обучения модель будет подготовлена ко всем заданным при аугментации искажениям в данных, что улучшит ее способность к решению задачи. Однако аугментация на лету очень чувствительна к параметрам заданных преобразований, из-за чего, неверный подбор и настройка искажений могут ухудшать данные на протяжении всего процесса обучения, что скажется на точности модели и ее способностях к решению задачи. В связи с этим в аугментации данных возникает проблема отслеживания влияния параметров аугментации на процесс обучения. К способам решения этой проблемы относится контроль за результатами во время обучения. Это подразумевает вывод ошибок с дальнейшим добавлением и настройкой искажения. Это позволит получить хорошо обученную модель, без переобучения и дополнительного дообучения [3].

Также особое внимание необходимо уделять выбору преобразований. Они должны выбираться в зависимости от задачи. Каждая из них может иметь свой набор специфических преобразований, подходящих для решения только

данной задачи. Верный подбор преобразований позволяет продвинуть качество модели без необходимости в усложнении архитектуры сети.

Если рассматривать медицинские изображения, то полезно учитывать искажения вида вращения, изменения масштаба, поскольку они совпадают с реальными искажениями, которые могут произойти при получении снимков.

2.4 Сегментация

2.4.1 Подходы к сегментации изображений

Рассмотрим наиболее информативные области на изображениях глазного дна, содержащие наибольшее количество признаков, указывающих на заболевание. Проведя анализ болезней глаза по данным, собранным в первой главе, был сделан вывод, что к наиболее значимым областям относятся диск зрительного нерва, чаша и сосуды глаза.

Было решено рассмотреть сегментацию диска зрительного нерва на изображениях. Основным преимуществом такого решения является отсутствие необходимости в сжатии изображения для его анализа. Из-за этого уменьшается вероятность потери признаков заболеваний.

Рассмотрим существующие подходы к сегментации данных. Для начала перечислим некоторые из них:

1. кластеризация;
2. методы выделения границ и контуров;
3. пороговая сегментация;
4. использование нейронных сетей.

Рассмотрим более подробно каждый из этих подходов.

Одним из методов является использование кластеризации. При таком подходе пиксели группируются на основе их сходства. Преимуществом такого подхода является то, что выделяются области с близкими по похожести характеристиками. Из недостатков выделяют то, что алгоритм теряет эффективность, если на изображениях присутствует шум и перекрывание объектов.

Также к методам сегментации относится метод выделения границ и контуров. Данный подход осуществляет выделение очертаний объектов. Он полезен в задачах анализа формы, выделения объектов, их детекции. Однако метод выделения границ и контуров не гарантирует полное выделение объектов.

Еще одним из методов сегментации является пороговая сегментация. В данном подходе разделение на классы происходит на основании значений

яркости пикселей. Устанавливается порог яркости, если пиксель имеет яркость выше, то он относится к одному классу, если ниже, то к другому. Однако на изображениях с неоднородным фоном или сложными условиями освещения качество метода пороговой сегментации может снижаться.

Также для сегментации изображений используются нейронные сети. Они обеспечивают высокую точность и позволяют сегментировать изображения с неоднородным фоном или в условиях сложного освещения. Это важно для нашей задачи, так как диск зрительного нерва может иметь нечеткие границы, а также иметь неоднородную освещенность вокруг.

В данной работе будем рассматривать традиционные сегментационные модели. Для решения задачи используем модель на основе U-NET. Могли также рассматривать модели на основе генеративно-состязательных сетей. Однако они решают более сложные задачи. Выбор был сделан в пользу традиционных сегментационных моделей по причине того, что для них не требуется такой большой объем тренировочных данных как для генеративно-состязательных сетей. Для понимания принципа работы моделей на основе U-NET, рассмотрим ее архитектуру.

2.4.2 Архитектура U-NET

Рассмотрим архитектуру сети U-NET. Обучение сети U-NET происходит на небольшом количестве изображений и осуществляется сквозным способом. Благодаря современным графическим процессорам сегментация изображений занимает менее секунды.

Рассмотрим некоторые основные характеристики U-NET. Данная сеть показывает высокие результаты в решении различных реальных задач. Она часто используется для решения биомедицинских задач. U-NET требует использования небольшого количества данных, сохраняя при этом высокие результаты.

Архитектура сети приведена на рисунке 2.5.

Сеть содержит 23 сверточных слоя. Архитектура состоит из расширяющего пути (находится на рисунке 2.5 справа) и сужающего пути (находится на рисунке 2.5 слева). Сужающий путь представляет собой повторное применение двух сверток 3×3. За ними следует слой ReLU. Для понижения разрешения используется операция максимального объединения.

Также происходит удваивание каналов свойств на каждом из этапов понижающей дискретизации.

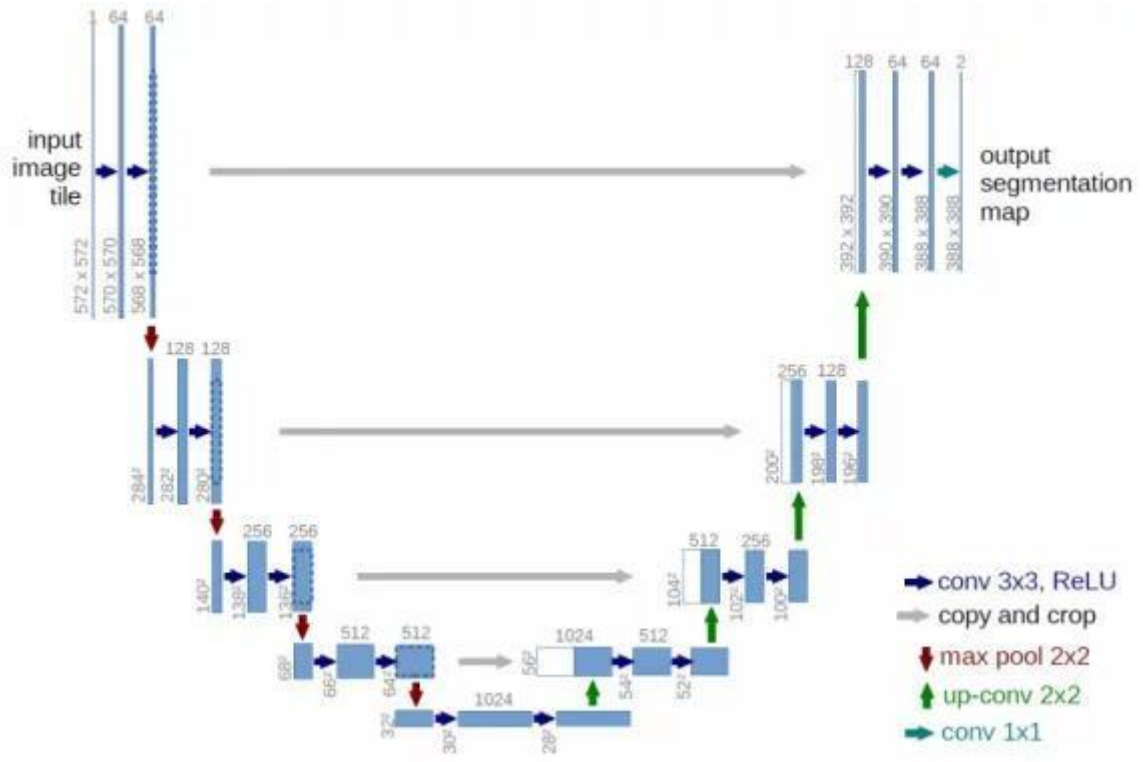

Рисунок 2.5 – Архитектура U-Net

На каждом этапе расширяющего пути осуществляется операция повышающей дискретизации карты свойств. Расширяющий путь представляет собой использование свертки 2×2. Она уменьшает количество каналов свойств. Затем следует объединение определенным образом с обрезанной картой свойств, полученной во время выполнения сужающего пути. Далее следуют две свертки 3×3, после которых идет слой ReLU.

Необходима обрезка, так как происходит потеря граничных символов на каждом этапе свертки [6].

На последнем слое используется свертка 1×1, которая позволяет сопоставить вектор свойств с желаемым количеством классов.

2.5 Фреймворки для создания веб-приложений

Для демонстрации работы моделей необходимо создать приложение, которое позволит пользователю классифицировать заболевание по снимку глазного дна и сегментировать изображение. Было принято решение, что будем рассматривать развертывание моделей машинного обучения как веб-приложений. Традиционно API оборачивается с помощью фреймворков, например, таких как Django или Flask. Однако существует более простой подход - использовать решение с небольшим количеством кода для создания веб-приложения. Примером такого подхода является использование Streamlit.

Streamlit – это фреймворк для создания интерактивных веб-приложений на Python.

Перечислим основные возможности Streamlit:

1. Streamlit предлагает быстрое развертывание модели: ML-модель можно быстро превратить в одностраничное веб-приложение и управлять им. При этом, нет необходимости в долгой верстке и использовании традиционных инструментов для создания веб-интерфейсов.

2. Приложения в Streamlit можно сделать интерактивными, что увеличивает простоту их изменения и удобство использования. Когда пользователь взаимодействует с веб-интерфейсом или происходят изменения в коде, Streamlit автоматически обновляет и перерисовывает нужные части страницы, то есть интерфейс откликается на действия разработчика или пользователя в реальном времени. В связи с этим, удобно делать интерактивные визуализации или простые пользовательские сервисы.

3. Также в Streamlit уже есть готовые встроенные стандартные виджеты для частых действий, такие как ползунки или поля для ввода текста. Благодаря этому можно довольно легко взять готовые виджеты и собрать из них работающий интерфейс. При необходимости можно отрисовать графики или картинки, вывести схемы или таблицы. Существуют функции для специализированных задач, например, отрисовка карт, на которых можно указать координаты, маршруты и линии.

4. Еще одним преимуществом является легкая интеграция с библиотеками, такими как pandas, matplotlib, plotly, tensorflow и другие.

Данный фреймворк был использован в работе из-за быстроты создания веб-интерфейсов для использования моделей машинного обучения.

2.6 Выводы

Во второй главе были рассмотрены сети EfficientNetB6 и YOLOv12, которые будут использоваться при реализации исследования. Изучены их архитектуры и преимущества в решении задач классификации изображений.

Также были изучены подходы к балансировке данных, которая является одним из наиболее важных этапов в подготовке данных для обучения модели, так как несбалансированный набор данных может сильно влиять на качество обучения. В качестве метода балансировки данных был выбран метод аугментации на лету, который имеет ряд преимуществ.

По данным первой главы видно, что некоторые области глазного дна имеют более важное значение, в связи с этим было рассмотрены подходы, которые могут использоваться для выделения данных областей. Была

рассмотрена и выбрана нейронная сеть U-Net для осуществления сегментации диска зрительного нерва.

Также был рассмотрен фреймворк для реализации веб-приложения. В качестве него был выбран фреймворк Streamlit.

Таким образом была изучена информация, необходимая для дальнейшей практической реализации исследования и выбраны инструменты и технологии для ее написания.

ГЛАВА 3
ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ МОДУЛЯ

3.1 Обзор исходного набора данных

Выбранный датасет представляет собой набор данных, доступный на Kaggle (система организации конкурсов по исследованию данных) для выявления глазных заболеваний. Он содержит снимки глаз более 5000 пациентов.

Набор данных состоит из цветных фотографий глазного дна левого и правого глаза и ключевых диагнозов, поставленных врачами. Они классифицируют пациентов по восьми категориям:

- нормальный(здоровый) (Н);
- диабет (Д);
- глаукома (Г);
- катаракта (С);
- возрастная макулярная дегенерация (А);
- гипертония (Н);
- патологическая близорукость (М);
- другие заболевания/аномалии (О).

Для лучшего понимания и анализа в первую очередь стоит разобраться в особенностях каждой болезни и в том, как она влияет на структуру глаза и, следовательно, понять, как это отражается на снимках глазного дна.

3.2 Подготовка данных для обучения моделей

Для начала было рассмотрено количество снимков, соответствующие каждому классу, для анализа того, какую подготовку данных следуют совершить для решения задачи. В таблице 3.1 представлено количество снимков, содержащих определенное заболевание.

Таблица 3.1 – Количество снимков согласно заболеваниям

Заболевание	Количество изображений
Здоровый	5501
Катаракта	594
Диабетическая ретинопатия	165
Миопия	457
Гипертоническая ретинопатия	382
Возрастная макулярная дегенерация	551
Глаукома	616

В таблице 3.1 показано, что при рассмотрении любого класса заболевания со здоровым, у нас будет несбалансированный набор данных. Для решения этой проблемы было решено применять аугментацию на лету.

Для каждого класса, кроме класса “Здоровый” выбираем максимальное возможное количество изображений, для класса “Здоровый” будем использовать по 600 изображений левого и правого глаза.

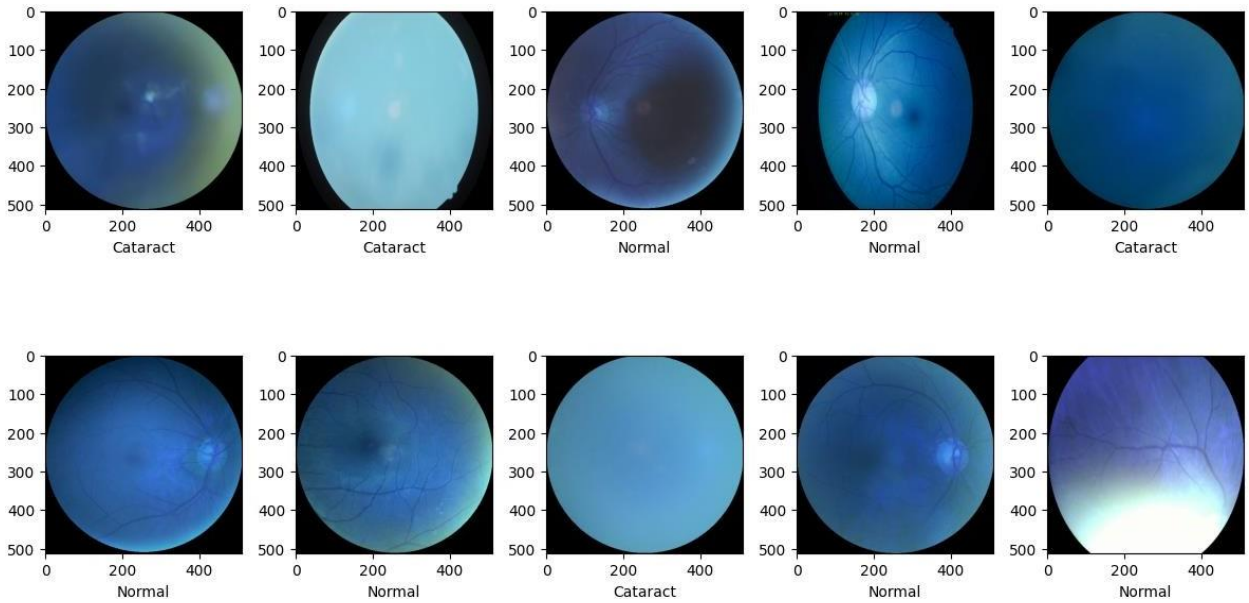

Рисунок 3.1 – Готовые для анализа изображения глазного дна

Осуществляем подготовку данных: загружаем данные из CSV-файла с диагнозами. Классификация заболеваний осуществляется на основе ключевых слов (например, "glaucoma", "diabetic retinopathy"). Загружаем и делаем препроцессинг изображений: для EfficientNetB6 и YOLOv12 рассмотрим изображения размера 512x512(исходный размер изображений). На Рисунке 3.1 можем увидеть готовые для анализа нейронной сетью изображения здоровых и больных миопией глаз.

Изображения делим на обучающую и тестовую выборки. 80 % используем для тренировки нашей нейронной сети, 20 % используем для проверки качества обучения нашей нейронной сети.

3.3 Обучение моделей

3.3.1 Обучение EfficientNetB6

Первоначально создаем экземпляры моделей с предварительно обученными весами на наборе данных ImageNet. Параметр include_top=False указывает, что верхние полностью связанные слои не будут включены, а с

помощью input_shape=(image_size, image_size, 3) задаем форму входных изображений.

К модели EfficientNetB6 добавляем слой GlobalAveragePooling2D, который выполняет глобальное усреднение по каждому из пространственных размеров входных данных. Он уменьшает пространство признаков, создавая вектор признаков фиксированной длины из выходных данных EfficientNetB6, и устраняет избыточность, что может помочь предотвратить переобучение. Также добавляем слой Dense - полносвязный слой с 512 нейронами. Активация ReLU (Rectified Linear Unit) применяется для введения нелинейности в модель, что позволяет ей обучаться более сложным функциям. Используем Dropout(0.3) для предотвращения переобучения. В качестве алгоритма оптимизации используется алгоритм Адама, в качестве функции потерь – Sparse Categorical Crossentropy для многоклассовой классификации, в качестве метрики - Accuracy. Для того, чтобы остановить обучение в случае, если потери на валидационных данных перестают уменьшаться в течение 5 эпох, используется EarlyStopping с параметром patience=5.

Для обучения модели используем аугментацию на лету. Так наша модель будет обучаться на datagen.flow(X_train, y_train, batch_size= 4), которая используется для создания генератора данных, который в свою очередь будет подавать данные в модель по партиям. Это полезно для обработки больших наборов данных, так как позволяет загружать данные по частям в память. Использование аугментации на лету заключается в том, что каждый раз на обучение поступают изображения, которые модель еще не видела. Это улучшает обобщающую способность модели, снижает вероятность переобучения и делает её более устойчивой к изменениям в данных. Используются следующие виды аугментации: поворот изображений на случайный угол, смещение изображения по ширине, смещение изображения по высоте. Для заполнения новых пикселей, которые могут возникнуть в результате трансформаций (например, при повороте или изменении размера) используется "nearest" (ближайший сосед). Это означает, что новые пиксели будут заполнены значениями ближайших пикселей.

В конце обучения модель тестируется на тестовой выборке и выводится классификационный отчет, содержащий метрики precision, recall, F1-score для каждого класса. А также выводится матрица ошибок.

3.3.2 Обучение YOLOv12

Первоначально конвертируем данные в формат YOLO и создаем YAML-файл для указания путей к данным и названий классов. Используем YOLOv12 с предобученной моделью для дообучения на наших данных. Загружается

модель из предоставленного файла (yolo12n.pt), которая уже обучена на базовых данных, таких как изображения общего назначения.

Настраиваем гиперпараметры: задаем количество эпох, размер изображений, размер батча и имя модели "yolov12_classifier", которое используется для идентификации результатов обучения.

Модель обучается с использованием предоставленных данных. YOLOv12 анализирует входные изображения, выполняя многослойное преобразование данных для выявления признаков заболеваний. Вычисляется функция потерь, оценивается, насколько сильно её текущие предсказания отличаются от истинных меток. На каждом шаге производится обновление весов нейронов сети с целью уменьшения функции потерь.

После каждой эпохи сохраняются контрольные точки модели (веса нейронной сети), чтобы можно было вернуться к промежуточным этапам обучения или использовать лучшую версию модели, а также производится проверка метрик на тренировочной и валидационной выборках, таких как точность, полнота и F1-score, чтобы отслеживать прогресс и предотвращать переобучение.

После окончания обучения проводится валидация модели с помощью следующих показателей: Precision (P, точность), Recall (R, полнота), mAP50 (Mean Average Precision при пороге IoU (Intersection over Union) = 0.5), mAP50-95(средняя точность по нескольким уровням IoU (от 0.5 до 0.95 с шагом 0.05), отражающая качество работы модели в целом).

3.4 Результаты

На тестовых данных модель EfficientNetB6 заключает диагноз с диапазоном точности в среднем 79-82 %.

Также для оценки работы для каждого класса были рассчитаны следующие метрики:

Precision (точность) – доля объектов, названных классификатором положительными, действительно является положительными. Она рассчитывалась по формуле (3.1):

$$\mathrm{Pr}\,e\,sision = \frac{TP}{TP+FP}, \tag{3.1}$$

где TP – истинно положительные

FP – ложно положительные

Recall (полнота) – то, какую долю объектов положительного класса из всех объектов положительного класса нашёл алгоритм. Рассчитывается по формуле вида (3.2):

$$Recall = \frac{TP}{TP+FN}, \tag{3.2}$$

где TP – истинно положительные

FN – ложно отрицательные

F1-score – это важная метрика для оценки качества мультиклассовых классификаторов, учитывающая как точность (Precision), так и полноту (Recall). В мультиклассовых задачах эта метрика может быть рассчитана для каждого отдельного класса или агрегирована по всем классам. Также баланс между точностью и полнотой помогает оценить, насколько модель хорошо идентифицирует каждый класс. Обозначим каждый класс через переменную i. Тогда вычисление для каждого класса i происходит по формуле (3.3):

$$F1 - score = \frac{2 \times \mathrm{Pr}\, ecision_i \times Recall_i}{\mathrm{Pr}\, ecision_i + Recall_i} \tag{3.3}$$

Также можно вычислять агрегированную по всем классам оценку. Существуют следующие виды таких оценок: Macro-averaging, Micro-averaging и Weighted-averaging(взвешенная оценка). При вычислении Macro-averaging первоначально просиходит вычисление Precision, Recall и F1-score отдельно для каждого класса, а затем усреднение полученных значений. Micro-averaging вычисляется по суммарным TP, FP, FN для всех классов перед расчетом Precision, Recall и F1-score. Weighted-averaging является аналог macro-averaging. Однако в нем учитывается частота встречаемости каждого класса.

В таблице 3.2 представлены показатель точности, полноты и F1-score для модели EfficientNetB6:

Таблица 3.2 – Оценки работы EfficientNetB6

Заболевание	precision	recall	f1-score	support
Миопия	0.90	0.81	0.85	123
Гипертоническая ретинопатия	0.85	0.85	0.85	84
Диабетическая ретинопатия	0.74	0.76	0.75	41
Глаукома	0.75	0.77	0.76	149
Катаракта	0.79	0.87	0.82	149

Продолжение таблицы 3.2

Возрастная макулярная дегенерация	0.76	0.76	0.76	93
Здоровый	0.78	0.75	0.76	260

В таблице 3.2 видно, что лучшая точность достигается для класса Миопии и Гипертоническая ретинопатия и составляет 85 %. Худший показатель у класса Диабетическая ретинопатия – 75 %, что связано с малым количеством изображений данного класса.

Для оценки работы модели YOLOv12 для каждого класса были рассчитаны:

mAP50 – средняя точность, рассчитанная при пороге intersection over union (IoU), равном 0,50.

mAP50-95 – среднее значение средней точности, рассчитанное при различных пороговых значениях IoU, варьирующихся от 0,50 до 0,95.

В таблице 3.3 представлены показатели работы модели YOLOv12 для классификации заболеваний.

Таблица 3.3 – Оценки работы модели YOLOv12

Заболевание	Изображение	P	R	mAP50	mAP50-95
Все	899	0.685	0.751	0.766	0.759
Миопия	94	0.897	0.957	0.957	0.952
Гипертоническая ретинопатия	89	0.637	0.730	0.762	0.753
Диабетическая ретинопатия	50	0.604	0.397	0.472	0.468
Глаукома	161	0.722	0.773	0.793	0.790
Катаракта	160	0.751	0.825	0.885	0.871
Возрастная макулярная дегенерация	98	0.595	0.796	0.787	0.780
Здоровый	247	0.589	0.777	0.706	0.702

Лучшая точность достигается для класса Миопии (mAP50 = 0.957) и Катаракты (mAP50 = 0.885). Худший показатель у класса Диабетическая ретинопатия (mAP50 = 0.472).

Можем заметить, что для обеих моделей сохраняется закономерность в достижении лучшей и худшей точности определения классов. Лучшие результаты наблюдаются в выявлении класса Миопии, что может быть связано с большим отличием в изображениях глаз других заболеваний и глаза, имеющего данное заболевание. На основании этого можем сделать вывод о

том, что для обучения важным является количество изображений, при малом количестве изображений модель не хватает данных для обучения, что приводит к низкой точности определения. Данный эффект может так же наблюдаться из-за расчета оценки, так как при малом количестве изображений не правильное определении пары изображений приводит к резкому снижению точности.

Для анализа этого можно использовать матрицу ошибок. Матрица ошибок (confusion matrix) – таблица с 4 различными комбинациями прогнозируемых и фактических значений. Прогнозируемые значения описываются как положительные и отрицательные, а фактические – как истинные и ложные.

На рисунках 3.2 и 3.3 изображены матрицы ошибок для моделей EfficientNetB6 и YOLOv12 соответственно.

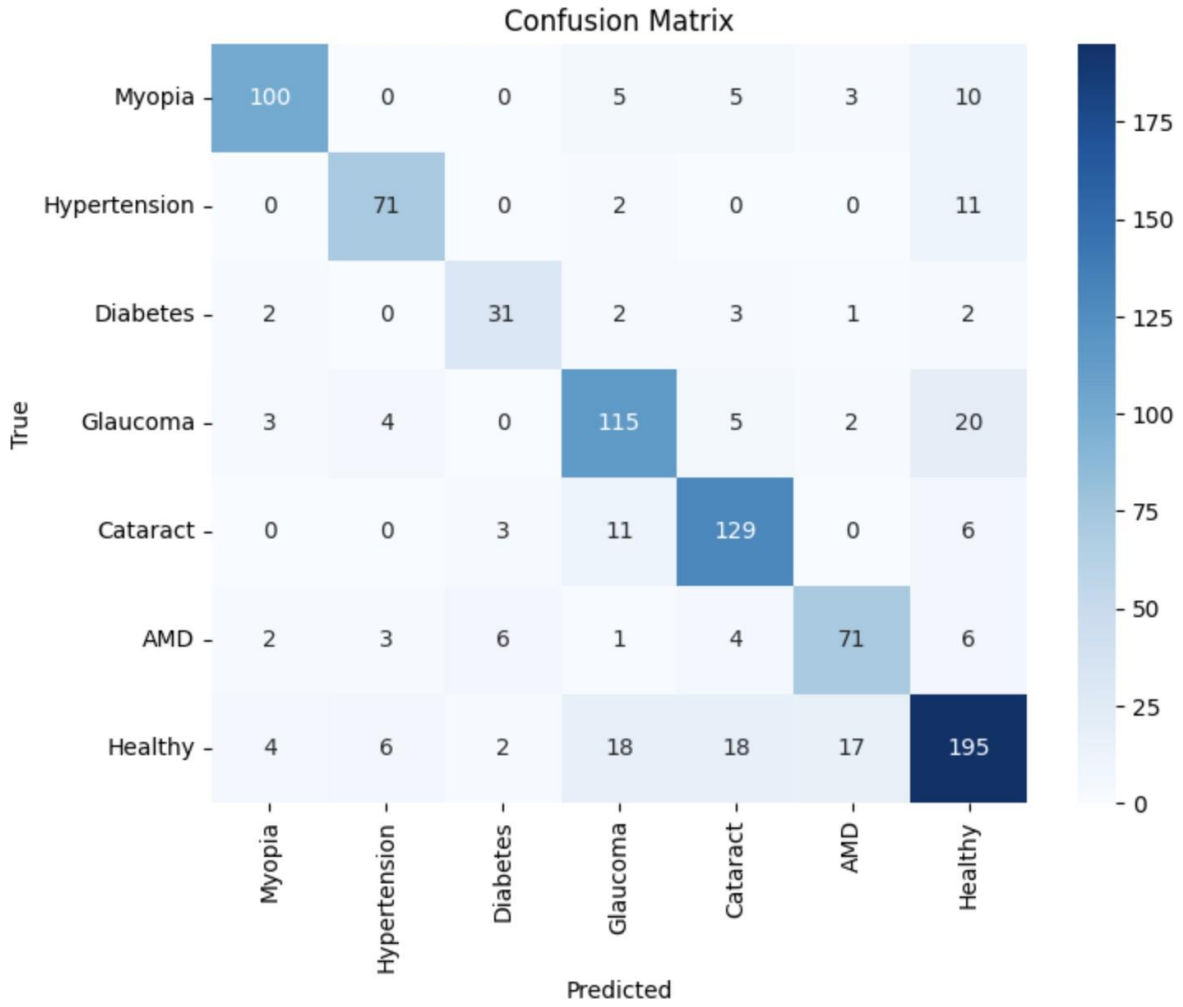

Рисунок 3.2 – Матрица ошибок для модели EfficientNetB6

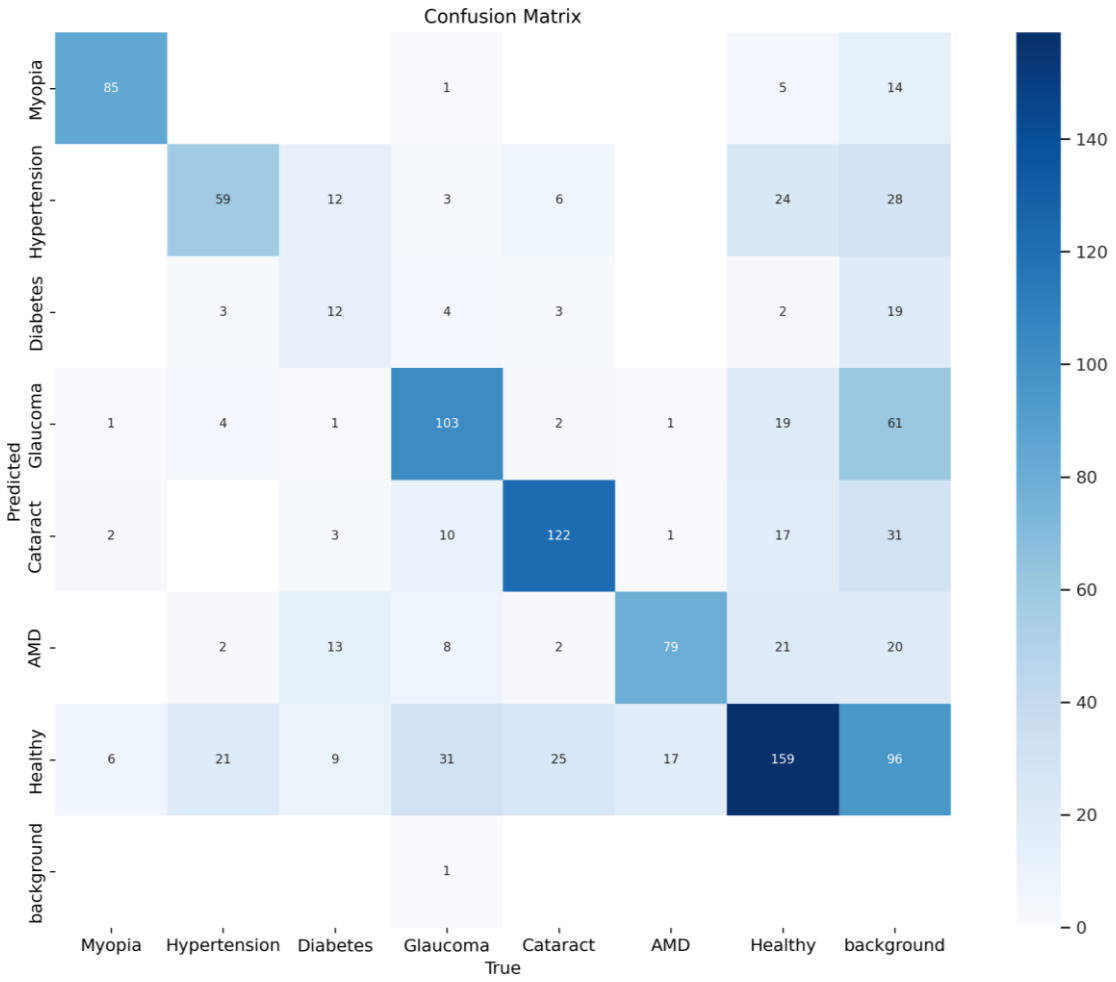

Рисунок 3.3 – Матрица ошибок для модели YOLOv12

Истинно положительные (True Positives, TP): количество объектов, которые модель правильно классифицировала как положительные.

Ложно положительные (False Positives, FP): количество объектов, которые модель ошибочно классифицировала как положительные. То есть, это случаи, когда модель указала на наличие заболевания, а на самом деле пациент здоров.

Истинно отрицательные (True Negatives, TN): Количество объектов, которые модель правильно классифицировала как отрицательные. Например, здоровые пациенты, которых модель также правильно определила, как здоровых.

Ложно отрицательные (False Negatives, FN): Количество объектов, которые модель ошибочно классифицировала как отрицательные. Это случаи, когда пациент на самом деле болен, но модель не смогла это выявить.

Попробуем улучшить результаты моделей, для этого создадим модель для выделения диска зрительного нерва. Это позволит нам рассмотреть наиболее информативную зону глаза.

Для анализа того, какие изображения были распознаны не верно была прописана функция вывода изображений, на которых метка классов не совпадает с предсказанной меткой. Результат работы такой функции представлен на рисунке 3.4.

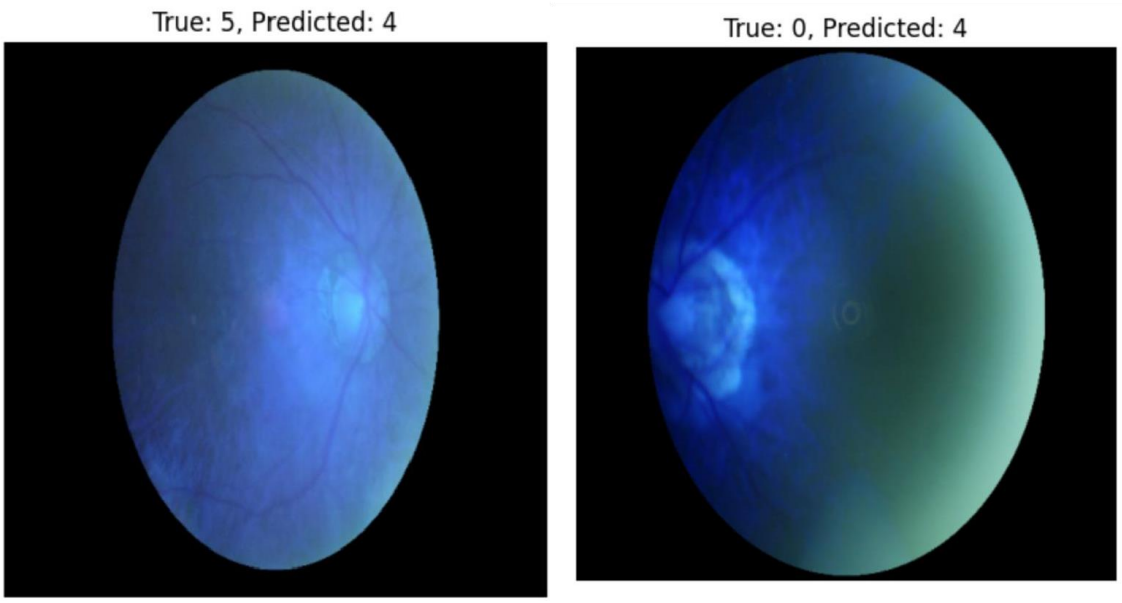

Рисунок 3.4 – Изображения, для которых истинные метки не совпали с предсказанными

Визуальный анализ ошибочно классифицированных изображений может помочь выявить общие паттерны среди неверно классифицированных изображений, такие как нечеткость снимков, присутствие шумов и артефактов. Анализ результатов показал, какие виды аугментации данных наиболее эффективны для улучшения производительности модели и уменьшения количества ошибок. Результаты анализа можно использовать для разработки стратегии дальнейшего улучшения модели, включая создание новых наборов данных, улучшение предобработки и аугментации.

3.5 Реализация сегментации изображений

Модель U-Net определим следующим образом: в качестве энкодера используем предобученную архитектура ResNet-34. Будем использовать веса, предобученные на базе данных ImageNet, что обеспечивает модели начальные представления о базовых структурах и формах, что помогает быстрее и точнее обучаться на задаче сегментации, особенно с небольшим количеством данных. Модель принимает на вход трехканальное изображение.

Для обучения модели будем использовать датасет с Kaggle, состоящий из изображений глазного дна и масок к ним. Пример изображения и маски к нему можно увидеть на рисунке 3.5.

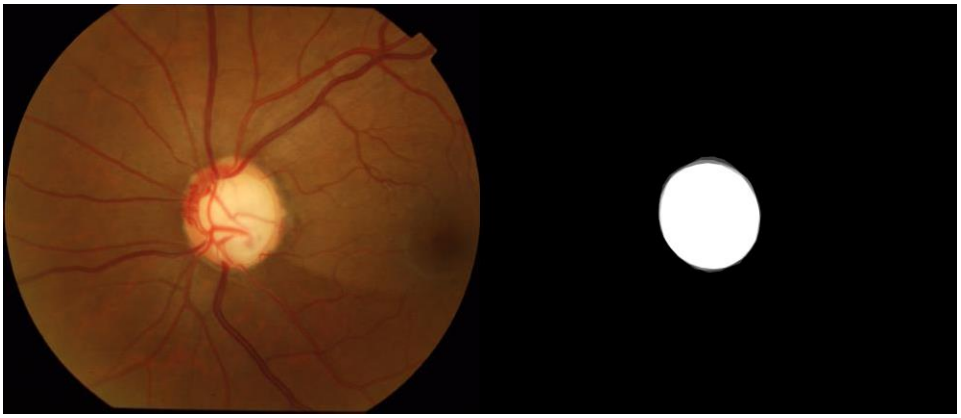

Рисунок 3.5 – Изображений глазного дна и маска

Модель принимает batch_size=8, данные перемешиваются перед каждой эпохой. Используется бинарная кросс-энтропия для сравнения предсказаний и истинных масок. Модель обучалась в течении 15 эпох. Точность сегментации составила 94 %.

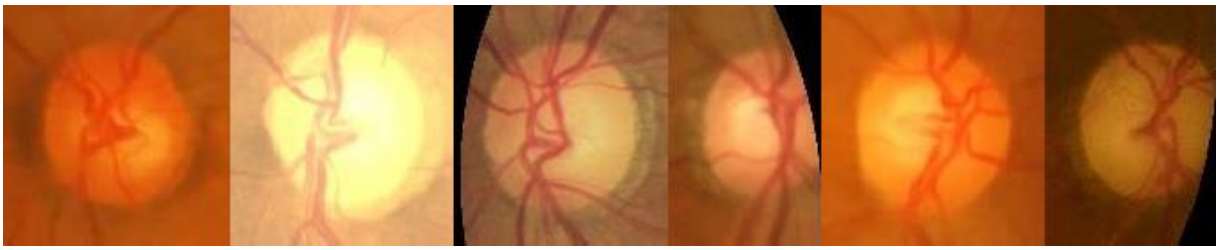

Рисунок 3.6 – Изображения, полученные с помощью сегментации

Данная модель была сохранена и применена для сегментации изображений рассматриваемого нами для задачи классификации. Была дополнительно применена функция выделения области в 20 пикселей вокруг диска зрительного нерва, с целью уменьшения вероятности потери данных в окрестностях диска. Результат работы данной функции и модели представлен на рисунке 3.6.

3.6 Реализация веб-приложения

Веб-приложение было написано с использованием библиотеки streamlit. Также были использованы:

1. TensorFlow для загрузки классификационной модели
2. OpenCV для обработки изображений
3. ONNX Runtime для работы с ONNX-моделью
4. стандартные библиотеки Python (NumPy, os, random и т. д.)

Также задаются 7 классов, которым соответствует каждая метка заболевания, например, «Миопия», «Гипертоническая ретинопатия» и так далее.

Это приложение предоставляет два режима работы: классификации заболевания по снимку глазного дна или сегментации изображения с выделением на изображении диска зрительного нерва. Выбор режима работы осуществляется пользователем. Используется боковое меню (sidebar) Streamlit, чтобы пользователь мог выбрать между режимом «Классификация» и «Сегментация». В зависимости от выбора вызывается соответствующая функция обработки.

На рисунке 3.7 можно увидеть первоначальный интерфейс, который видит пользователь, заходя в веб-приложение.

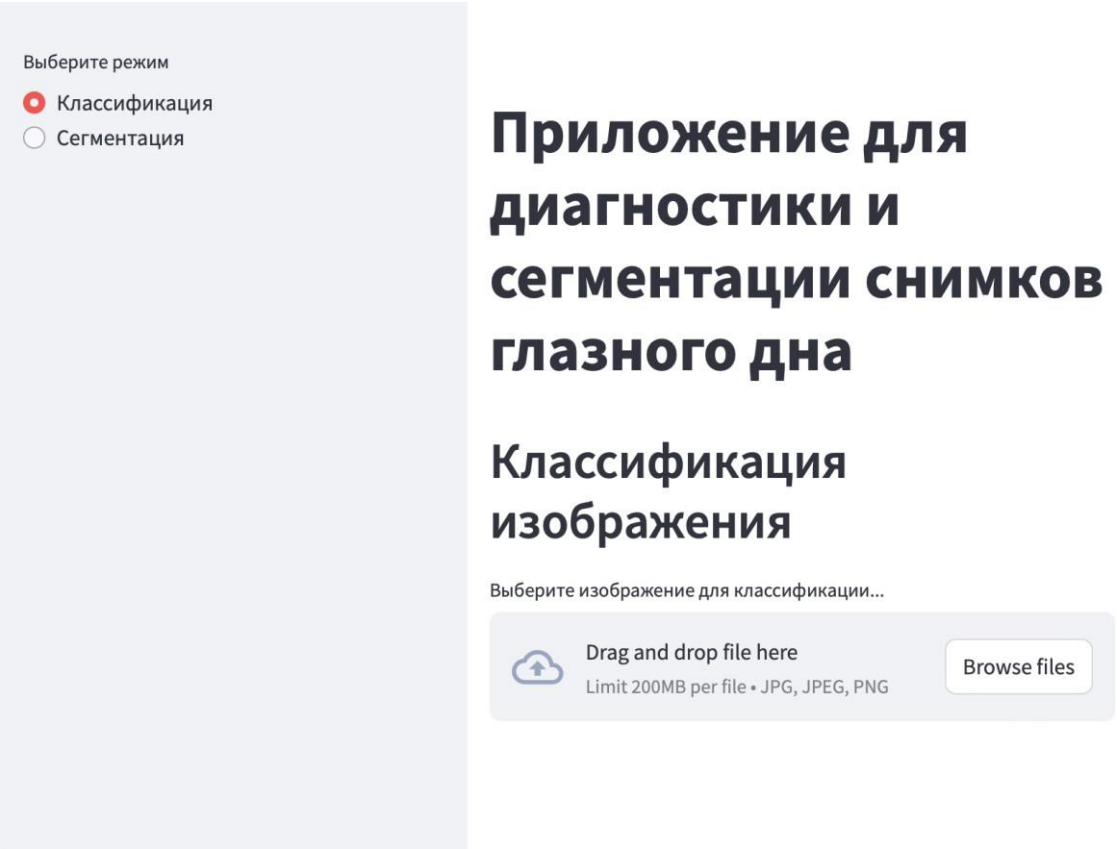

Рисунок 3.7 – Изображение интерфейса во время начала работы

При выборе решения задачи классификации пользователь загружает изображение, которое предварительно обрабатывается для дальнейшего анализа моделью, после чего передается в классификационную модель, загружаемую через TFSMLayer. Модель возвращает вероятностное распределение по 7 классам заболеваний и классу Здоровый. На основании вероятностного распределения выбирается наиболее вероятный класс,

отображаемый на экране вместе с процентными значениями для каждого класса. Пример вывода можно видеть на рисунке 3.8.

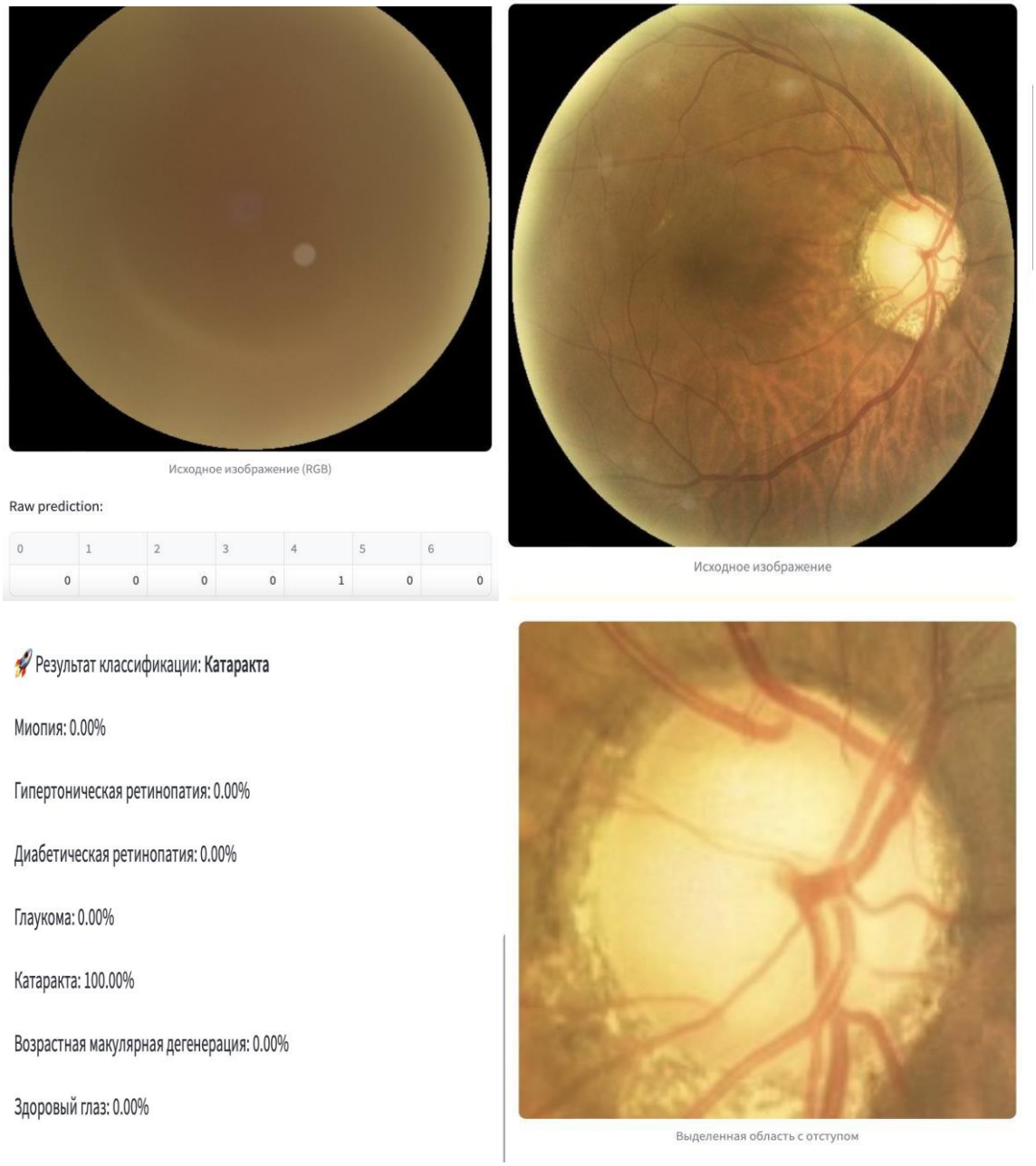

Рисунок 3.8 – Вид отображения результата работы классификации и сегментации

При решении задачи сегментации изображения пользователь загружает изображение, затем выполняется изменение его размера (для соответствия входному размеру модели). Затем осуществляется поиск сегмента и выделение найденной области с добавлением небольшого отступа. В результате отображается выделенная область. Пример вывода работы сегментации можно увидеть на рисунке 3.8.

Код реализации веб-приложения представлен в Приложении А.

3.7 Выводы

В данной главе представлена практическая реализация исследования. Был описан процесс обучения моделей, и результаты, полученные в ходе обучения, рассмотрены метрики, которые помогают корректнее интерпретировать и проанализировать результаты. Это позволило сделать вывод о важности количества данных для обучения нейронной сети, а также о том, какие заболевания лучше распознаются нейронными сетями, на основе закономерности в полученных результатах для обеих рассмотренных моделей.

Была также создана модель для сегментации изображений на основе U-Net, что в дальнейшем позволит изучить влияние выделения областей глаза, таких как диск зрительного нерва, для улучшения качества классификации заболеваний на снимках глазного дна. Модель показала точность сегментации в 94 процента.

Также описан процесс создания приложения для практической демонстрации работы моделей.

ЗАКЛЮЧЕНИЕ

В данной работе проводился анализ признаков заболеваний на изображениях глазного дна методами машинного обучения. Для ознакомления с проблемой выявления болезней глаз были рассмотрены основные особенности заболеваний, ориентируясь на которые, можно выделить признаки заболеваний на снимках глазного дна.

Были рассмотрены существующие нейронные сети и проведен их анализ, для выбора наиболее подходящих для решения нашей задачи. В качестве сетей для классификации было принято решение использовать сети EfficientNetB6 и YOLOv12. С ними проводились эксперименты по выявлению заболеваний на изображениях глазного дна. Для этого был отобран датасет содержащий снимки глаз 5 тысяч пациентов, размеченные заболеваниями, рассмотренными в первой главе. Был проведен анализ датасета, для определения того, какая подготовка данных необходима для качественного обучения сети. Для увеличения разнообразия в данных применялась аугментация на лету.

Анализ результатов исследования показал, что нейронные сети могут использоваться для классификации заболеваний, однако стоит уделять особое внимание качеству, количеству и правильной подготовке данных, так как эти аспекты имеют большое влияние на обучение сети.

Были рассмотрены также возможности для улучшения осуществления детекции заболеваний. Была применена сегментация диска зрительного нерва, для дальнейшего исследования влияния выделения информативных областей глаза на анализ признаков заболеваний. Для создания сегментационной модели использовалась архитектура нейронной сети U-NET. Модель обучалась на датасете, содержащем снимки глазного дна и маски к ним. Точность обучения составила 94 %.

Было разработано веб-приложение, которое дает возможность практического применения исследования. Оно позволяет продемонстрировать работу моделей. Так, было разработано два режима работы “Классификация” и “Сегментация”. Для создания приложения были рассмотрены существующие подходы и выбран фреймворк Streamlit.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Волков, В. В. Глазные болезни. Основы офтальмологии / В. В. Волков [и др.] под общ. ред. В. Г. Копаевой. – 4-е изд. – М. : Офтальмология. – 2018. – DOI: 10.25276/978-5-903624-36-2

2. Гуревич, И. Б. Новый математический метод автоматизации анализа флюоресцентных ангиограмм глазного дна человека / И. Б. Гуревич [и др.]. // Вестник офтальмологии. – 2021. – Т. 137, №3. – С. 49-57. – DOI: 10.17116/oftalma202113703149

3. Емельянов, С. О. Методы аугментации обучающих выборок в задачах классификации изображений / С. О. Емельянов [и др.] // Сенсорные системы. – 2018. – Т. 32, № 3. – DOI: 10.1134/S0235009218030058

4. Свистунова, К. И. Классификация глазных заболеваний по снимкам глазного дна с помощью нейронной сети Efficientnet / К. И. Свистунова, С. В. Абламейко // BIG DATA и анализ высокого уровня = BIG DATA and Advanced Analytics : сборник научных статей XI Международной научно-практической конференции, Минск, 23–24 апреля 2025 года / редкол.: В. А. Богуш [и др.]. – Минск : БГУИР, 2025. – С. 168-173

5. Нероев, В. В. Разработка прототипа сервиса для диагностики диабетической ретинопатии по снимкам глазного дна с использованием методов искусственного интеллекта / В. В. Нероев, А. А. Брагин, О. В. Зайцева // Национальное здравоохранение. – 2021. – Т. 2, №. 2. – С. 64-72. – DOI: 10.47093/2713-069X.2021.2.2.64-72

6. Сикорский, О. С. Обзор свёрточных нейронных сетей для задачи классификации изображений / О. С. Сикорский // Новые информационные технологии в автоматизированных системах – М., 2017 – №20. – С. 37-42.

7. Старовойтов, В. В. Оценка качества цифровых изображений сетчатки / В. В. Старовойтов, Ю. И. Голуб, М. М. Лукашевич // Системный анализ и прикладная информатика. – 2021. – №. 4. – С. 25-38

8. Хайкин, С. Нейронные сети: полный курс / С. Хайкин – 2-е изд. – М. : Издательский дом «Вильямс». – 2006. – 1104 с.

9. Himbitskaya, E., Svistunova, K Enhancing Fundus Image Classification With Semantic Segmentation Based Attention Mask = Улучшение Диагностики Изображений Глазного Дна С Использованием Маски Алгоритма Внимания На Основе Семантической Сегментации / E. Himbitskaya, K. Svistunova, G. Karapetyan, A. Nedzved, S. Ablameyko // Open Semantic Technologies for Intelligent Systems (OSTIS): сборник научных трудов / редкол.: В. В. Голенков [и др.]. – Минск: БГУИР, 2025. – С. 261–267.

10. Tan, M. EfficientNet: Rethinking model scaling for convolutional neural networks // M. Tan, Q. Le // International conference on machine learning. – PMLR, 2019. – P. 6105-6114.

11. U-Net: нейросеть для сегментации изображений // Режим доступа: https://neurohive.io/ru/vidy-nejrosetej/u-net-image-segmentation/ – Дата доступа: 20.05.2025

12. Badrinarayanan, V. Segnet: A deep convolutional encoder-decoder architecture for image segmentation / V. Badrinarayanan, A. Kendall, R. Cipolla // IEEE transactions on pattern analysis and machine intelligence. – 2017. – Т. 39, №. 12. – P. 2481-2495. – DOI: 10.1109/TPAMI.2016.2644615

13. YOLO12: обнаружение объектов, ориентированных на внимание // Режим доступа: https://docs.ultralytics.com/ru/models/yolo12/#overview – Дата доступа: 20.05.2025

ПРИЛОЖЕНИЕ А

КОД ДЛЯ РЕАЛИЗАЦИИ ВЕБ-ПРИЛОЖЕНИЯ

```
import streamlit as st
import tensorflow as tf
import numpy as np
import cv2
from PIL import Image
import onnxruntime as ort
import os
import random

# Для классификации
disease_classes = [
    "Миопия",
    "Гипертоническая ретинопатия",
    "Диабетическая ретинопатия",
    "Глаукома",
    "Катаракта",
    "Возрастная макулярная дегенерация",
    "Здоровый глаз"
]

onnx_model_path = "diskSegmentation.onnx"

# Режим 1: Классификация изображения

@st.cache_resource
def load_classification_model():
 # Загрузка модели
    input_layer = tf.keras.Input(shape=(512, 512, 3)
    tf_sm_layer = tf.keras.layers.TFSMLayer("ALLEfficientNet_saved",
call_endpoint="serving_default")
    output_layer = tf_sm_layer(input_layer)
    model = tf.keras.Model(inputs=input_layer, outputs=output_layer)
    return model

# Загрузка классификационную модель
classification_model = load_classification_model()

def preprocess_image_classification(image, target_size=(512, 512)):
    if image.mode != "RGB":
```

```
        image = image.convert("RGB")
    image = image.resize(target_size)
    image_array = np.array(image, dtype=np.float32) / 255.0
    image_array = cv2.cvtColor(image_array, cv2.COLOR_RGB2BGR)
    image_array = np.expand_dims(image_array, axis=0)
    return image_array

def classification_mode():
    st.header("Классификация изображения")
    uploaded_file = st.file_uploader("Выберите изображение для
классификации...", type=["jpg", "jpeg", "png"])
    if uploaded_file is not None:
        image = Image.open(uploaded_file)
        st.image(image, caption="Исходное изображение (RGB)",
use_column_width=True)
        processed_image = preprocess_image_classification(image, target_size=(224,
224))
        # Выполняем предсказание
        prediction = classification_model.predict(processed_image)["output_0"]
        st.write("Raw prediction:", prediction) predicted_class_index =
np.argmax(prediction, axis=1)[0]
        predicted_disease = disease_classes[predicted_class_index]
        st.write(f" Результат классификации: **{predicted_disease}**")
        # Вывод вероятностей для всех классов
        probabilities = prediction[0] * 100
        for i, prob in enumerate(probabilities):
            st.write(f"{disease_classes[i]}: {prob:.2f}%")

# Режим 2: Сегментация изображения

@st.cache_resource
def load_segmentation_model():

    try:
        sess = ort.InferenceSession(onnx_model_path)
        return sess
    except Exception as e:
        st.error(f"Ошибка загрузки сегментационной модели: {e}")
        return None

# Загружаем сегментационную модель
segmentation_session = load_segmentation_model()
if segmentation_session is not None:
    input_name = segmentation_session.get_inputs()[0].name
```

```
    output_name = segmentation_session.get_outputs()[0].name

def segment_disk_with_margin_from_array(image, margin=10):
    original_image = image.copy() # сохраняем исходное изображение
    # Меняем размер для сегментации до 352×352
     resized_image = cv2.resize(original_image, (352, 352))
     normalized_image = resized_image.astype('float32') / 255.0
   input_image = np.expand_dims(normalized_image, axis=0)  # форма (1, 352, 352, 3)

    try:
        result = segmentation_session.run([output_name], {input_name: input_image})[0]
    except Exception as e:
        st.error(f"Ошибка при выполнении инференса сегментационной модели: {e}")
        return None, None
    # В оригинальном коде результат имеет вид [0, :, :, 0]
    result = result[0, :, :, 0]
    mask = (result > 0.5).astype(np.uint8)
    # Масштабируем маску обратно до размера исходного изображения
    mask_resized = cv2.resize(mask, (original_image.shape[1], original_image.shape[0]))
    contours, _ = cv2.findContours(mask_resized, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    if contours:
        largest_contour = max(contours, key=cv2.contourArea)
        x, y, w, h = cv2.boundingRect(largest_contour)
        x_start = max(x - margin, 0)
        y_start = max(y - margin, 0)
        x_end = min(x + w + margin, original_image.shape[1])
        y_end = min(y + h + margin, original_image.shape[0])
        disk_region_with_margin = original_image[y_start:y_end, x_start:x_end]
        return mask_resized, disk_region_with_margin
    else:
        st.warning("Сегментация не удалась.")
        return mask_resized, None

def segmentation_mode():
    st.header("Сегментация изображения")
    uploaded_file = st.file_uploader("Выберите изображение для сегментации...", type=["jpg", "jpeg", "png"])
    if uploaded_file is not None:
        # Загружаем изображение через PIL и преобразуем в numpy (RGB)
```

```
        image = Image.open(uploaded_file)
        st.image(image, caption="Исходное изображение",
use_column_width=True)
        image_np = np.array(image)
        # Выполняем сегментацию (функция принимает numpy-массив)
        mask, disk_region = segment_disk_with_margin_from_array(image_np,
margin=20)
        if disk_region is not None:
            st.image(disk_region, caption="Выделенная область с отступом",
use_column_width=True)
        else:
            st.error("Не удалось выделить область диска.")

# MAIN: Выбор режима через Sidebar

def main():
    st.title("Приложение для диагностики и сегментации снимков глазного дна")
    mode = st.sidebar.radio("Выберите режим", ("Классификация",
"Сегментация"))

    if mode == "Классификация":
        classification_mode()
    elif mode == "Сегментация":
        segmentation_mode()

if __name__ == "__main__":
    main()
```