Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра интеллектуальных информационных технологий

Налетов Илья Вячеславович

Исследование и разработка методов обнаружения роста бактерий в микробиологических анализах на основе обработки фотоизображений.

ДИПЛОМНАЯ РАБОТА

Научный руководитель:
Попов Иван Сергеевич

Москва, 2021

Оглавление

Аннотация

Данная работа посвящена исследованию задачи автоматизированного определения роста бактерий на основе фотоизображений образцов с анализами в процессе их созревания, а так же сопровождающих их неграфических параметров, для применения в качестве дополнения к традиционному способу визуальной оценки специалистами-микробиологами в ручном режиме.

В рамках исследований был произведен обзор предметной области, в ходе которого, архитектура трансформер была выбрана в качестве основной архитектуры решения поставленной задачи. Предложенный в работе метод показал схожую со специалистами точность обнаружения на доступном массиве размеченных исторических данных. Для улучшения качества классификации были рассмотрены различные методы предобработки изображений.

Введение

В случае недомогания, человек, чаще всего отправляется к врачу, во многих случаях врач направляет его на сдачу необходимых анализов, пациент сдает их и ждет результатов. Ускорение сроков их выдачи является платной услугой во многих клиниках. Конечно же, очевидно, что любая болезнь лечится тем хуже, чем позже она выявлена.

Вывод такой, хочется иметь возможность узнавать о наличии вредоносных бактерий в организме человека как можно раньше.

До сих пор в современной медицине подавляющая масса анализов обрабатывается вручную.
В большинстве лабораторий все устроено следующим образом - большое количество лаборантов, ручной посев анализов (помещения биоматериала в специальную среду), хранение, ведение их учета, ручная утилизация, мониторинг всех стадий развития бактерий и выдача результатов.

При ручном, неавтоматизированном, подходе можно выявить множество проблем, такие как:

- острая нехватка персонала
- ошибки, связанные с человеческим фактором (30% анализов утилизируются по причине ошибок при посеве)
- и основная – физическая ограниченность человеческих возможностей, как следствиедолгая выдача результатов анализов

Все большее количество клиник приходят к необходимости автоматизации разных этапов обработки анализов.
На помощь приходят автоматизированные системы посева, выполняющие следующие функции:

- засеивание бактерий с помощью особой технологии с использованием магнитного шарика
- поддержка необходимых условий для развития бактерий
- фотографирование образцов с требуемой периодичностью

- хранение, очистка и ведение учета всех образцов, утилизация

В среднем за восьмичасовой рабочий день лаборант может засеить анализы в 800 чашек, автоматизированные системы засева способны работать без остановки и способны засеивать 300 чашек в час.

Как сказано выше, установка указанной системы решает большое количество проблем. Такая автоматизированная система берет на себя все процессы связанные с обслуживанием чашек Петри, но все еще остается этап, который требует вмешательства специалистов, а именно, анализ результатов. Врач анализирует изображения, которые создает система, делая выводы о том, что развилось или не развилось в данном среде.

Рисунок 1: *Визуализация возможных путей анализа бактерий*

После помещения анализов в среду возможны несколько исходов.
Через некоторое время, разное для разных бактерий, становится понятно, вырастет ли что-то в этой среде. В случае отрицательного ответа происходит утилизация анализов.
Если же обнаружен рост бактерий, то врач классифицирует их, а в случае выявления вредоносных отправляет их на анализ в масс-спектрометр для получения финального вывода.

Обнаружение наличия роста по имеющимся фотоизображениям – основная задача в данной дипломной работе работе.

Постановка задачи

Основная задача: разработать метод устанавливающий наличие или предсказывающий отсутствие роста бактерий по сериям фотоизображений образца, зафиксированным в последовательные промежутки времени.

Подзадачи, которые необходимо решить в рамках основной:

- Задача раннего обнаружения роста бактерий
- Задача ранней установки отсутствия роста
- Обнаружение и фильтрация разного рода артефактов на изображениях

Обзор

Обзор статей по классификации изображений

Большое количество статей посвящено анализу медицинских изображений. В первую очередь это касается рентгеновских изображений и анализа внутренних органов.
Не большое количество авторов посвятило себя работе в области классификации бактерий. Среди всего многообразия статей только три представленные ниже статьи решают схожую задачу и предлагают интересные подходы к решению.

1. Bacterial Colony Counting with Convolutional Neural Networks in Digital Microbiology Imaging

   Описание статьи:

   Подсчет бактериальных колоний на микробиологических пластинах является трудоемкой, подверженной ошибкам, но, тем не менее, важной количественной задачей в лабораториях клинической микробиологии. В своей работе авторы исследуют возможность найти эффективные решения вышеуказанной проблемыпутем разработки и тестирования двух различных подходов к машинному обучению. Первый из них основан на извлечении полного набора признаков вручную, которые в дальнейшем используются при обучении SVM. Второй основан на проектировании и настройке архитектуры глубокого обучения сверточных нейронных сетей. Чтобы проверить предлагаемые методы оценки
   бактериальной нагрузки, авторы создали и публично опубликовали полностью маркированную большую и репрезентативную базу данных как отдельных, так и агрегированных бактериальных колоний, извлеченных из обычных клинических лабораторных пластин. Подходы к улучшению набора данных также были экспериментально протестированы для оптимизации производительности. Подходк глубокому обучению с большим отрывом превзошел подход, основанный на ручном формировании признаков, став предпочтительным решением для задачи количественной оценки цифровых изображений микробиологии.

2. An Automated Deep Learning Approach for Bacterial Image ClassificationОписание статьи:

   Автоматизированное распознавание и классификация видов бактерий по микроскопическим изображениям имеют важное значение в клинической микробиологии. Классификация бактерий обычно проводится вручную биологамис использованием различных форм и морфологических характеристик видов бактерий. Ручное систематизирование типов бактерий по микроскопическим изображениям занимает много времени и является сложной задачей даже для опытных биологов. В этом исследовании был предложен подход к классификации, основанный на автоматизированном глубоком обучении, для классификации бактериальных изображений по различным категориям. Предварительно обученная архитектура CNN ResNet-50 была использована для классификации цифровых изображений бактерий по 33 категориям. Метод дообучения был использован для ускорения процесса обучения сети и повышения эффективности классификации. Предложенный метод достиг средней точности классификации 99,2%.
3. Early-detection And Classification of Live Bacteria Using Time-lapse Coherent Imagingand Deep Learning

   Описание статьи:

   Ранняя идентификация патогенных бактерий в продуктах питания, воде и жидкостях организма очень важна и в то же время сложна из-за сложности образцов и большого объема образцов, которые необходимо быстро проверить. Существующие методы скрининга, основанные на подсчете пластин или молекулярном анализе, представляют собой некоторые компромиссы в отношении времени обнаружения, точности/чувствительности, стоимости и сложности подготовки образцов. Авторы данной статьи представляют вычислительную систему обнаружения живых бактерий, которая раз в 30 минут делает фотографиюбактерий внутри агаровой пластины и анализирует эти временные ряды с использованием глубоких нейронных сетей для быстрого обнаружения роста бактерий и классификации соответствующих видов. Эффективность системы была продемонстрирована быстрым обнаружением бактерий кишечной палочки в пробах воды, что сократило время обнаружения роста бактерий более чем на 12 ч по сравнению с аналитическими методами, одобренными Агентством по охране окружающей среды. Эксперименты также подтвердили, что этот метод успешно обнаруживает 90% бактериальных колоний в течение 7-10 ч и >95% в течение 12 ч

с точностью 99,2-100% и правильно идентифицирует их виды в течение 7,6–12 ч с точностью 80%.

Обзор статей по предобработке изображений

В ходе решения задачи стало очевидно, что большое количество дефектов и артефактов на изображениях мешают ее решению с необходимой точностью. Удалению и поиску дефектов на изображениях посвящено множество статей. Проанализированы и взяты в разработки следующие работы:

1. Clearing the Skies: A deep network architecture for single-image rain removalОписание статьи:

   Авторы статьи представляют нейросетевую архитектуру под названием DerainNetдля удаления дождя с изображений. Поскольку авторы не обладали изображениями, соответствующими реальным изображениям дождя, они синтезируют изображения с дождем для обучения. В отличие от других распространенных стратегий, которые увеличивают глубину или ширину сети, встатье используются знания о предметной области обработки изображений для изменения целевой функции и улучшения передачи данных с помощью CNN скромного размера. Хотя DerainNet обучается на синтетических данных, обнаружено, что обученная сеть очень эффективно себя показывает при работе среальными изображениями.
2. A Global Optimization Method for Specular Highlight Removal from A Single Image Описание статьи:

   Наличие бликов является критической проблемой как для естественных, так и для медицинских изображений, таких как изображения, полученные с помощью лапароскопов, что может привести к ошибочному визуальному отслеживанию, реконструкции стерео и сегментации изображений. Авторы статьи предлагают метод глобальной оптимизации для удаления зеркальных бликов с одного изображения на основе модели дихроматического отражения. В дополнение к введению модифицированной цветности освещения предлагаемый метод состоит из двух новых этапов: один для оценки диффузной цветности путем коррекции оттенка и насыщенности на выделенных областях, а другой для оценки коэффициентов диффузного и зеркального отражения с использованием выпуклойоптимизации с двойной регуляризацией. Доказано, что оцененная диффузная цветность приближается к истинной диффузной цветности, и предлагаемый

алгоритм оптимизации гарантированно находит оптимальные коэффициенты диффузии. Экспериментальные результаты показывают, что предложенный метод позволяет эффективно удалять зеркальные блики как с естественных изображений,так и с эндоскопических изображений с сохранением деталей текстуры.

3. Veiling Glare in High Dynamic Range ImagingОписание статьи:
   Способность камеры записывать изображение с высоким динамическим диапазоном, будь то один снимок или последовательность, ограничена наличием маскирующих бликов - тенденцией ярких объектов в сцене уменьшать контраст везде в поле зрения. Блик - это глобальный эффект освещения, возникающий в результате многократного рассеяния света внутри корпуса камеры и оптики объектива. Измеряя отдельно прямые и косвенные компоненты внутрикамерного переноса света, можно увеличить максимальный динамический диапазон, которыйможет записывать конкретная камера. В этой статье авторы количественно оценивают наличие маскирующих бликов и связанных с ними оптических артефактов для нескольких типов цифровых камер и описывают два метода их удаления: деконволюцию с помощью измеренной функции распространения
   бликов и прямое-косвенное (direct-indirect) разделение переноса объектива с использованием структурированной окклюзионной маски.

Существующие подходы анализа изображений

1. ***Сравнение по перцептивному хэшу***
   Перцептивные хэш - алгоритмы описывают класс функций для генерации сравнимых хэшей. Характеристики изображения используются для генерацииндивидуального (но не уникального) отпечатка, и эти отпечатки можно сравнивать друг с другом.

   Перцептивные хэши — это другая концепция по сравнению с криптографическими хэш-функциями вроде MD5 и SHA1. В криптографии каждый хэш является случайным. Данные, которые используются для генерации хэша, выполняют роль источника случайных чисел, так что одинаковые данные дадут одинаковый результат, а разные данные — разный результат. Из сравнения двух хэшей SHA1 на самом деле можно сделать только два вывода. Если хэши отличаются, значит,
   данные разные. Если хэши совпадают, то и данные, скорее всего, одинаковые

(поскольку существует вероятность коллизий, то одинаковые хэши не гарантируют совпадения данных). В отличие от них, перцептивные хэши можно сравнивать между собой и делать вывод о степени различия двух наборов данных.

Все алгоритмы вычисления перцептивного хэша обладают одинаковыми базовыми свойствами: картинки можно изменять в размере, менять соотношение сторон и даже слегка менять цветовые характеристики (яркость, контраст и т.д.), но они всё равно совпадают по хэшу. И самое главное: такой алгоритм очень быстрый!

Если вам нужно сравнить две картинки, то просто строите хэш для каждой из них и подсчитываете количество разных битов (это расстояние Хэмминга). Нулевое расстояние означает, что это, скорее всего, одинаковые картинки (или вариации одного изображения). Далее, чем выше расстояние, тем выше отличие изображений.

2. *HOG и анализ с помощью методов машинного обучения*

Построение признакового пространства с помощью HOG.

Основной идеей алгоритма является допущение, что внешний вид и форма объекта на участке изображения могут быть описаны распределением градиентов интенсивности или направлением краев. Реализация этих дескрипторов может бытьпроизведена путём разделения изображения на маленькие связные области, именуемые ячейками, и расчетом для каждой ячейки гистограммы направлений градиентов или направлений краев для пикселей, находящихся внутри ячейки.

Комбинация этих гистограмм и является дескриптором. Для увеличения точности локальные гистограммы подвергаются нормализации по контрасту. С этой целью вычисляется мера интенсивности на большем фрагменте изображения, который называется блоком, и полученное значение используется для нормализации. Нормализованные дескрипторы обладают лучшей инвариантностью по отношениюк освещению.

Дескриптор HOG имеет несколько преимуществ над другими дескрипторами. Поскольку HOG работает локально, метод поддерживает инвариантность геометрических и фотометрических преобразований, за исключением ориентации

объекта. Подобные изменения появятся только в больших фрагментах изображения.

После построения дескриптора мы можем использовать его в качестве признаков для обучения любого выбранного алгоритма машинного обучения.
Основным, ведущим методом машинного обучения являются деревья решений и ансамбли над ними (случайный лес и градиентный бустинг на деревьях).

Деревья решений:

Дерево решений — метод представления решающих правил в определенной иерархии, включающей в себя элементы двух типов — узлов (node) и листьев (leaf). Узлы включают в себя решающие правила и производят проверку примеров на соответствие выбранного атрибута обучающего множества.

Примеры попадают в узел, проходят проверку и разбиваются на два подмножества:
первое — те, которые удовлетворяют установленное правило;
второе — те, которые не удовлетворяют установленное правило.

Далее к каждому подмножеству снова применяется правило, процедура повторяется. Это продолжается, пока не будет достигнуто условие остановки алгоритма. Последний узел, когда не осуществляется проверка и разбиение, становится листом.

Лист определяет решение для каждого попавшего в него примера. Для дерева классификации — это класс, ассоциируемый с узлом, а для дерева регрессии — соответствующий листу модальный интервал целевой переменной. В листе содержится не правило, а подмножество объектов, удовлетворяющих всем правилам ветви, которая заканчивается этим листом.

Пример попадает в лист, если соответствует всем правилам на пути к нему. К каждому листу есть только один путь. Таким образом, пример может попасть только в один лист, что обеспечивает единственность решения.

В основе критерия разбиения лежит информационная энтропия:

$$H = -\sum_{i=1}^{n} \frac{N_i}{N} \log\left(\frac{N_i}{N}\right)$$

где *n* — число классов в исходном подмножестве, *Ni* — число примеров i-го класса, *N* — общее число примеров в подмножестве.

Энтропия рассматривается как мера неоднородности подмножества по представленным в нем классам. И даже если классы представлены в равных долях, а неопределенность классификации наибольшая, то энтропия тоже максимальная. Логарифм от единицы будет обращать энтропию в ноль, если все примеры узла относятся к одному классу.

Если выбранный атрибут разбиения Aj обеспечивает максимальное снижение энтропии результирующего подмножества относительно родительского, его можно считать наилучшим.

Так же часто используется критерий Джини в качестве критерия для разбиения:

$$\mathrm{Gini}(Q) = 1 - \sum_{i=1}^{n} p_i^2$$

где *Q* — результирующее множество, *n* — число классов в нем, *pi* — вероятность i-го класса (выраженная как относительная частота примеров соответствующего класса).

Идея случайного леса состоит в том, чтобы построить много решающих деревьев, по разным правилам, и в итоге усреднить их ответы.

Идея градиентного бустинга состоит в том, чтобы стоить каждый новый алгоритм, как уточнение предыдущего. Градиентный бустинг на данный момент показывает лучшие результаты в большинстве задач машинного обучения.

3. Convolutional Neural Network.

Сверточные нейронные сети основаны на такой математической операции как свертка и всегда являлись передовым решением в задачах анализа изображений.

Двумерная свертка (2D convolution) — это довольно простая операция: начинаем с ядра, представляющего из себя матрицу весов (weight matrix). Ядро “скользит” над

двумерным изображением, поэлементно выполняя операцию умножения с той частью входных данных, над которой оно сейчас находится, и затем суммирует все полученные значения в один выходной пиксель.

Ядро повторяет эту процедуру с каждой локацией, над которой оно “скользит”, преобразуя двумерную матрицу в другую все еще двумерную матрицу признаков. Признаки на выходе являются взвешенными суммами (где веса являются значениями самого ядра) признаков на входе, расположенных примерно в том же месте, что и выходной пиксель на входном слое.

Независимо от того, попадает ли входной признак в “примерно то же место”, он определяется в зависимости от того, находится он в зоне ядра, создающего выходные данные, или нет. Это значит, что размер ядра сверточной нейронной сети определяет количество признаков, которые будут объединены для получениянового признака на выходе.

4. ***Transformer***

Вывод

Поставленная задача не имеет готового решения, а значит, требуется разработать собственный подход, позаимствовав идеи, описанные в статьях на схожую тематику.

В рамках данной работы необходимо сравнить все вышеуказанные методы и выявить тот, который покажет себя лучше всего. Акцент в первую очередь будет стоять на сравнении сверточных нейросетей и нейросетей архитектуры трансформер.

При решении многих задач, построение собственной нейросети не является обязательным – возможным подходом оказывается использование готовых архитектур, которые демонстрируют хорошую точность при решении классификационных задач.

Предобработка изображений является важным этапом разработки решения. Обучение на предобработанных фото способно увеличить качество модели, по сравнению с работой с сырыми данными.

Исследование и построение решения

Используемый подход

Классификационная нейросеть на вход получает предобработанное изображение, а на последний линейный слой конкатенируются неграфические признаки. Выход нейросети - вероятность того, что на данном изображении наблюдается рост бактерий.

Ранее использовался подход, при котором на вход нейросети подавалась разность изображений в разные моменты времени. Разработки в данном направлении в текущий момент приостановлены, так как часть бактерий уже на первой фотографии обладают ростом, а значит мы должны уметь делать выводы, в лучшем случае, имея только одно изображение.

Особенности задачи (постановка)

1. Задача предполагает минимизацию количества ошибок второго рода
   В силу того, что данная задача связана со здоровьем людей, количество ошибок второго рода необходимо свести к минимуму, возможно, даже за счет некоторого понижения точности модели.
2. Большое количество артефактов на изображениях
   В ходе исследований стало понятно, что на изображениях присутствуют разногорода дефекты, которые мешают обучению нейросети. Такие как: блики, просвечивающиеся надписи, отвалившиеся шарики для засеивания бактерий, конденсат, царапины и прочие мелкие артефакты.
3. Работа с большим объемом изображений, в высоком разрешении.
   Изначальный размер изображений 2000x2000. Весь датасет, уже приведенный к нужному виду – около 85 тысяч изображений (всего изображений около 250тыс., но в силу разметки мы не можем использовать все).

Особенности задачи (решение)

1. Увеличение штрафа модели за ошибки второго рода
   Для борьбы с ошибками второго рода были проведены эксперименты с изменением лосс-функции, в которой был повышен штраф за них. Исследовалась зависимость величины штрафа и метрик accuracy и recall. За счет подбора значения коэффициентаудалось практически полностью избавиться от ошибок второго рода(значение recall 0.994), при несильном падении точности (около 0.005).
2. Изучены статьи по удалению бликов и борьбе с артефактами нафотоизображениях.

Предобработка изображений пока еще в стадии изучения. Была воспроизведена одна из статей, представленных выше [4]. Подход, использованный в статье, может быть хорошо использован, например, для удаления бликов, конденсата илицарапин с изображений.

3. Отдельно хотелось бы задержаться на пункте с обработкой большого количества данных. В силу особенностей архитектуры модели, на данный момент изображение 2000x2000 преобразуется в изображение 224x224. Но так как мы работаем с большим количеством изображений (размеры датасетов будут указаны ниже), а так же применяем аугментацию данных – появилась необходимость добавления оптимизационных решений в pipeline.

   Оптимизация:

   - DDP
     Было добавлено DDP, то есть возможность параллельно обучать модель на нескольких видеокартах
   - Mixed precision
     Перевод основной части операций в fp16. То есть обычно модели работают с данными в формате float32, если мы переведем большинство операций на float16 это сильно облегчит вычисления.
   - Cupy
     Cupy это библиотека, которая проводит свои вычисления на видеокарте, что значительно быстрее. Сделано было следующим образом: был взят код библиотеки для аугментации Albumentations, и используемые при аугментации функции были переписаны на cupy.

Этапы построения решения

То с чего началось построение решения – был анализ всех имеющихся данных, выявление их особенностей, распределений, объемов. При первом этапе знакомства с поставленной задачей были обнаружены значительные проблемы в разметке. А именно: мы не могли утверждать, что данные помеченные, как выросшие, действительно являются таковыми, мы обладали лишь знанием, что в данном анализе что-то выросло, но в какой среде и на каком фото достоверно не было известно. В связи с этим был проведен анализ данных и выявлено, что в большинстве случаев, если рост бактерии и был обнаружен, то он чаще происходил в bloodAgar среде. Было решено начать разработку архитектуры решения

задачи с построения нейросети, которая обрабатывала бы только изображения со средой bloodAgar. CromAgar был оставлен до появления новой разметки.

Был построен baseline, который сравнивал изображения по их перцептивному хэшу. Данный подход очень плохо работал с изображениями такого разрешения и реагировал на любые минимальные изменения в фото, такие как освещение, царапины, поворот изображения, поэтому этот самый простой аналитический метод было решено оставить.

В связи с тем, что требовалось построить модель, которая бы анализировала наличие роста бактерий, было решено подавать в модель не одно изображение, а разность фото, стоящих рядом во временной цепи. Долгое время этот подход являлся основным. Было построено большое количество функций, для приведения данных к нужному виду.

Объемы данных были очень большие, каждое изображение поставлялось в разрешении 2000x2000, локальной компьютер не был способен обрабатывать такие объемы информации, поэтому был получен доступ к серверу nvidia DGX-2, который обладает обширной оперативной памятью, а главное предоставляет свои вычислительные мощности в виде GPU. Работа с GPU значительно ускоряет обучение как, например, lightGBM, так и любых нейросетевых архитектур.

Следующим подходом в решении было тестирование построения признакового пространства с помощью HOG и обучение lightGBM на этих данных. Этот подход, аналогично перцептивному хэшу не показал удовлетворительных результатов, поэтому было решено перейти к построению сверточных нейросетей.

Велись исследования влияния разрешения на качество моделей, так же была попытка работать с черно-белыми изображениями, но такой подход только ухудшал показатели. Но уже на тот момент собственная архитектура показывала точность модели около 0.92, что является низким показателем с точки зрения задачи, как задачи работы со здоровьем людей, но достаточно высоко для просто задачи бинарной классификации.

С какого-то момента стало понятно, что необходима предобработка изображений, в связи с большим количество артефактов, а так же в связи с тем, что за пределами чашки-Петри, находилась еще справочная информация, которая модели не требовалась. С этого момента начали применять маскирование изображений, для удаления лишней информации с фото.

Исследования велись с разрешениями изображений 200x200 и 500x500. Большие размеры было сложно поместить в оперативную память.

В связи с тем, что изображения круглые, для уменьшения переобучения моделей было решено использовать аугментацию поворотом на случайный угол от 1 до 360 градусов, что на каком-то этапе давало прирост точности.

Уже на том этапе становилось понятно, что собственная архитектура обладаем малым количеством параметров, для такого объема данных, в связи с этим было решено попробовать подход fine tuning с уже готовыми нейросетевыми архитектурами. В итоге оказалось, что данных так много, что вполне возможно самостоятельно обучать эти большие модели, без заморозки каких-либо слоев. Лучше всех себя показала модель Xception [7] с точностью около 0.958.

После того, как было решено использовать модель Xception был проведен рефакторинг кода, приведения файлов к хорошей структуре, для демонстрации и легкого понимания, при необходимости, новыми разработчиками. Для автоматизации процессов был создан файл config, изменяя лишь параметры которого можно было перестраивать модели и получать необходимые цифры. Был создан репозиторий на github для анализа версий. В общем была проведена полная гигиена всей дипломной работы.

Анализ изображений, на которых ошибалась нейросеть выявил большое количество ошибок в разметке, а так же большое количество артефактов, что в очередной раз указало на необходимость нахождения решения по предобработке фото. Указанная выше точность стала на некоторое время непреодолимой планкой, а так же работа с chromAgar не могла стартовать, в связи с плохой разметкой.

В момент ожидания новой разметки изучались статьи по предобработке данных, в первую очередь по удалению бликов с изображений. Была воспроизведена статья [4] и некоторая часть статьи [6].

Появление знания о том, на каком именно изображении врач обнаружил рост бактерии дало новый виток в разработке. Был изменен подход представления данных. Теперь нейросеть стала получать на вход не разность изображений, а каждое изображение самостоятельно, в связи с тем, что для некоторых бактерий возможен рост уже с первых

фото (первое фото для некоторых бактерий делается через несколько часов, а не в момент засеивания).

Пришлось привести датасет к новому виду, разбить на датасет для раннего обучения и для обучения на достоверных данных (далее описано подробнее). Так же появилась возможность работать со средой chromAgar, так же как и с bloodAgar. Для каждой среды была построена своя нейросеть со схожими архитектурами.

На новой разметке была достигнута точность 0.987 для cromAgar и 0.976 для bloodAgar. Оставшийся небольшой процент ошибок, был вызван небольшим процентом ошибок врачей в разметке, а так же наличием артефактов, как и раньше.

Новой задачей для анализа стала задача раннего обнаружения роста, особый датасет был протестирован на моделях, обученных на достоверных данных, точность заметно упала – 0.953 и 0.947 chromAgar и bloodAgar соответственно. Для повышения точности раннего обучения было решено попробовать подход при котором в модель добавляются дополнительно неграфические признаки, помимо фотоизображений, а так же построение нового датасета, разбитого по часам, а так же, возможное, перестроение бинарной архитектуры в модель, которая предсказывает три параметра (да, нет, не знаю), для уменьшения количества ошибок второго рода, при таком подходе модель не обязана будет давать итоговый ответ уже на первом изображении, она может дождаться второго момента времени и более уверенно там ответить, все равно, в среднем, уменьшив время ответа на вопрос, выросло ли что-то в этом анализе или уже не вырастет.

В первую очередь было решено провести эксперименты по добавлению в модель дополнительной информации, помимо изображения. На этом этапе так же было принято решение перейти с Keras на Pytorch и полностью переписать pipeline, для того, чтобы сделать его более гибким и масштабируемым.

Pipeline был переведен на Pytorch, появилась модульность и решение стало легко изменяемым. Далее начались эксперементы со Swin Transformer, а так же с добавлением неграфических признаков в модель.
Теперь было принято решение сделать одну общую модель на все среды, и текущая Balanced-Accuracy стала составлять: 0.987.
Так же к этому моменту количество данных выросло с 7 месяцев до 11, что так же

определенно помогло сети увеличить качество работы.

Неграфические данные

До того, как добавлять параметры в нейросеть было необходимо отобрать из имеющихся те, которые могли бы нести потенциальную ценность и в целом проанализировать качество данных.
По каждому изображению в датасете мы имели по 42 неграфических признака.
Часть данных сразу была отброшена, так как несла вспомогательную/техническую информацию и не несла в себе полезных данных. После этого были построены корреляции Пирсона и Спирмена с целевой метрикой, а так же на оставшихся данных была обучена модель LightGBM, для получения feature_importance каждого признака.
По итогам из 42 параметров, как потенциально полезные были отобраны только 9, при этом и они не особо оправдали наши надежды.

Таблица 1: *Влияние неграфических признаков на целевую метку.*

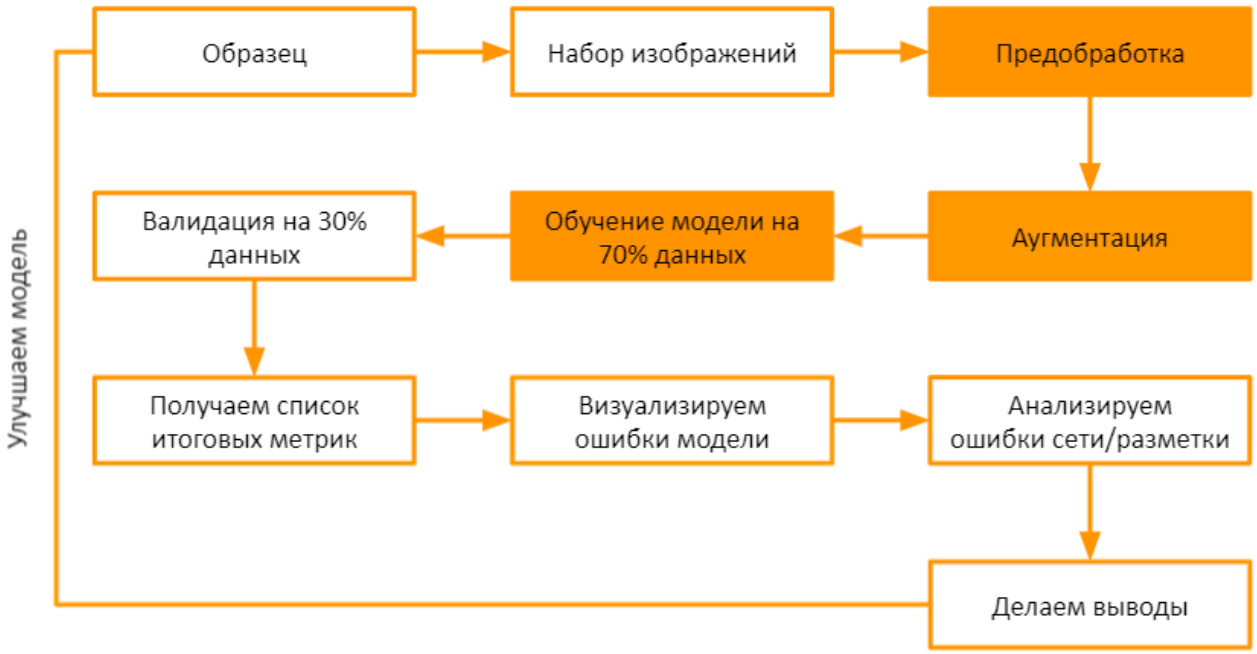
	feature	importance	pearson	spearman
0	ДатаПоступленияБиоматериала	1183	-0.03	-0.03
1	ВремяВыполнения	1175	-0.03	-0.02
2	ПациентДатаРождения	932	-0.06	-0.05
3	Биоматериал	908	-0.11	-0.13
4	Kiestra_AS_ID	354	-0.27	-0.38
5	Kiestra_CS_ID	317	-0.37	-0.31
6	Kiestra_SCAN_NR	242	0.48	0.50
7	ПациентПол	167	-0.10	-0.09
8	Kiestra_MEDIA_NAME	142	-0.09	-0.14

ДатаПоступленияБиоматериала – дата поступления биоматериала для анализа
ВремяВыполнения – когда был проведен анализ
ПациентДатаРождения – дата рождения пациента
Биоматериал - какой биоматериал взят на анализ

Kiestra_AS_ID - айди, который выставляется врачом и он обозначает последовательность анализов которые надо провести с той или иной чашкой. (Конкретная последовательность нас особо не интересует. Т.к. мы исследуем только один анализ - обнаружение роста посредством визуального осмотра врача и дальнейшего анализа в масспектрометре).
Kiestra_CS_ID - айди, который выставляется системой посева и отражает настройки камеры.
SCAN_NR - номер скана в последовательности.
ПациентПол – пол пациента
Kiestra_media_name - название среды, на который развиваются колонии бактерий.

Архитектура решения

Рисунок 2: *Архитектура предложенного решения задачи раннего обнаружения роста бактерий.*



Используемые метрики

Для сравнения изображений:

1) PSNR:

$$PSNR = 10\log_{10}\left(\frac{MAX_I^2}{MSE}\right) = 20\log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right)$$

2) MSE:

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}|I(i,j) - K(i,j)|^2$$

3) SSIM

$$\text{SSIM}(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

Для оценка качества классификации:

	$y = 1$	$y = 0$
$\hat{y} = 1$	True Positive (TP)	False Positive (FP)
$\hat{y} = 0$	False Negative (FN)	True Negative (TN)

1) Accuracy:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

2) Precision:

$$precision = \frac{TP}{TP + FP}$$

3) Recall:

$$recall = \frac{TP}{TP + FN}$$

4) F1-score: $\beta = 1$

$$F_\beta = (1 + \beta^2) \cdot \frac{precision \cdot recall}{(\beta^2 \cdot precision) + recall}$$

Для обучения модели:

1) Binary Cross-entropy:

$$BCE = -\frac{1}{N}\sum_{i=0}^{N} y_i \cdot log(\hat{y}_i) + (1 - y_i) \cdot log(1 - \hat{y}_i)$$

В силу особенностей данной задачи для обучения использовалась взвешенная функция потерь (weighted BCE), с весами равными отношению классов 0 и 1. Ключевой метрикой выбрана recall, в связи с тем, что требуется обнаружить все случаи, когда обнаружен рост бактерии, даже пускай, с потерей точности.

Практическая часть

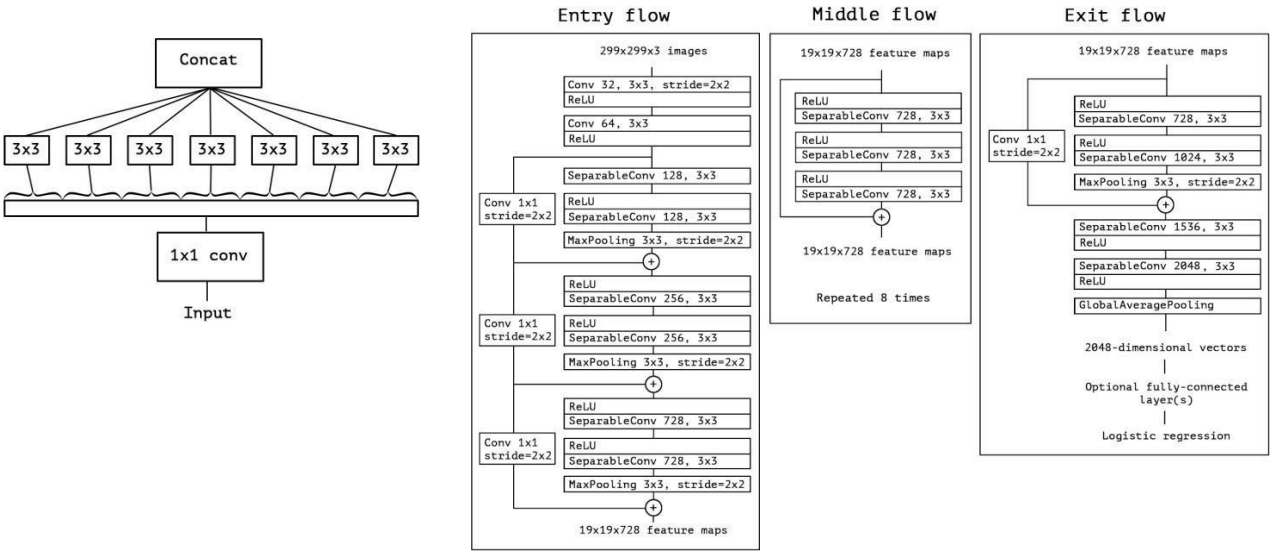
Для решения поставленной задачи был выбран язык Python. Для построения нейросетей изначально использовалась библиотека Keras, так как это самая удобная библиотека для построения нейросетей, ее функционал слегка ограничен по сравнению с другими (tensorflow, pytorch), но его вполне хватало, для решения данной задачи. В последствии был произведен переход на pytorch для расширения возможностей использования современных методов.

Как уже говорилось ранее, для решения поставленной задачи были выбраны две архитектуры Xception как представитель сверточных нейросетей и Swin Transformer, как представитель архитектуры трансформер. В экспериментах можно увидеть итоги сравнения этих методов.

При изначальном подходе была попытка построить собственную нейросеть, но потом, как предложено в статье [2], было решено воспользоваться готовыми архитектурами, хорошо показавшими себя при решении других задач анализа изображений. Архитектура Xception, которая была заинициализирована весами после обучения на ImageNet, но не замораживалась, а обучалась с нуля. В силу большого количества данных, даже такая нейросетевая архитектура как Xception (22910480 параметров) смогла спокойно обновлять веса, не ощущая нехватки знаний.

Архитектура Xception:

Рисунок 3: *Блок Depthwise separable convolution и архитектура сети Xception.*



Чтобы улучшить качество распознавания своих сетей, исследователи старались добавлять в сети больше слоёв, однако со временем пришло понимание, что иногда ограничения производительности попросту не позволяют обучать и использовать настолько глубокие сети. Это стало мотивацией для использования depthwise separable convolutions и создания архитектуры Xception.

Представим, что мы взяли стандартный свёрточный слой с C2 фильтрами размера 3x3, на вход которому подается тензор размерности M*M*C1, где M — это ширина и высота тензора, а C1 — количество каналов.

Что делает такой слой? Он сворачивает одновременно все каналы исходногосигнала C2 разными свёртками. На выходе у такого слоя получается тензор размерности (M−2)*(M−2)*C2.

Вместо этого сделаем последовательно два шага:

1. Свернём исходный тензор 1x1 свёрткой, подобно тому как мы делали в блоке Inception, получив тензор M*M*C2. Эта операция называется pointwise convolution

2. Свернём каждый канал по отдельности 3x3 свёрткой (при этом размерность не изменится, так как мы сворачиваем не все каналы вместе, как в обычном свёрточном слое). Эта операция называется depthwise spatial convolution

Архитектура Swin Transformer:

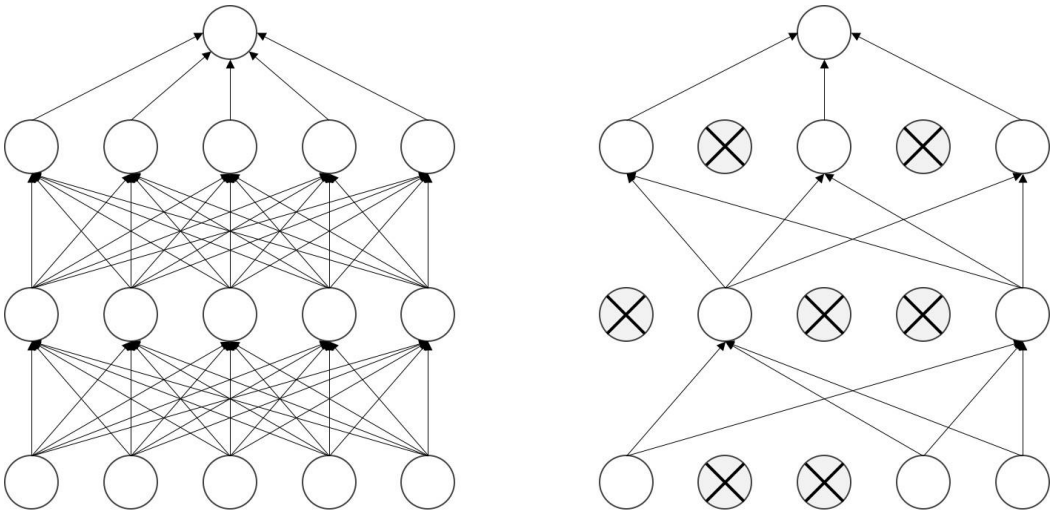
Работа с архитектурой нейросети

Dropout

Для уменьшения переобучения моделей в архитектуру были добавлены Dropout слои. Цель этого метода – снизить специализацию каждого отдельного нейрона. Именно в этом корень проблемы переобучения.

Сети для обучения получаются с помощью исключения из сети (dropping out) нейронов с вероятностью p, таким образом, вероятность того, что нейрон останется в сети, составляет q=1−p. “Исключение” нейрона означает, что при любых входных данных или параметрах он возвращает 0.

Рисунок 5: *Визуализация подхода Dropout.*

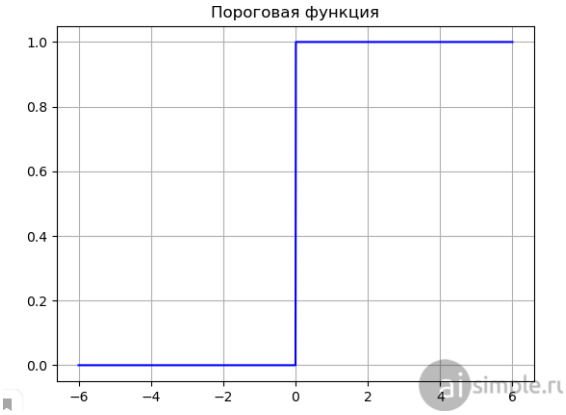


Функции активации

При работе с нейросетями огромную роль играет выбор функции активации. О влиянии функций активации написано много статей, приведем формулы и графики самых часто используемых из них:

1) Пороговая

$$f(x) = \begin{cases} 1, & x \geq a \\ 0, & x < a \end{cases}$$

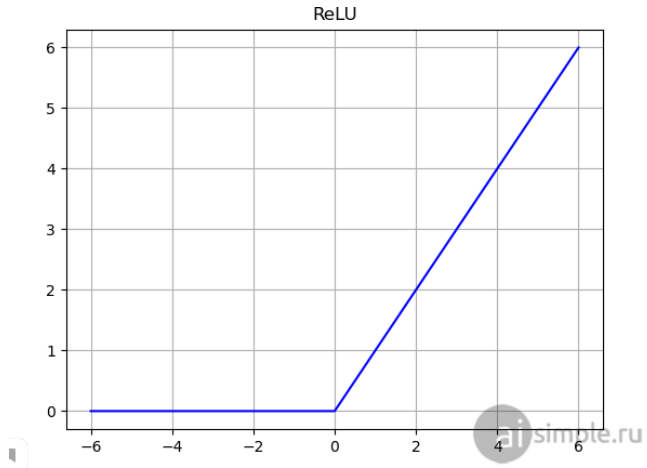


2) Линейная

$$f(x) = c * x$$

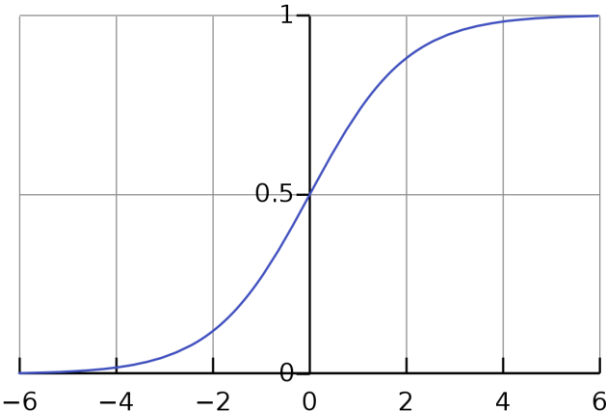
3) ReLU

$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases}$$



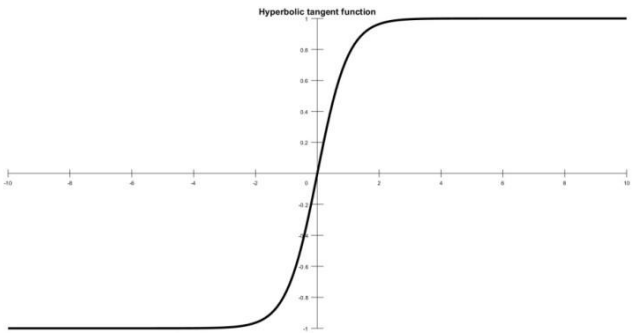
4) Сигмоида

$$f(x) = \frac{1}{1 + e^{-x}}$$



5) Гиперболический тангенс

$$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$



В большинстве задач анализа изображений разработчики отдают свое предпочтение функции ReLU, так как она благоприятно влияет на градиент (не происходит его быстрого затухания), помогая большим моделям обучаться и воспринимать большее количество данных.

Данные и разметка

В задачах анализа данных основное влияние на качество их решения в первую очередь оказывает имеющийся датасет.

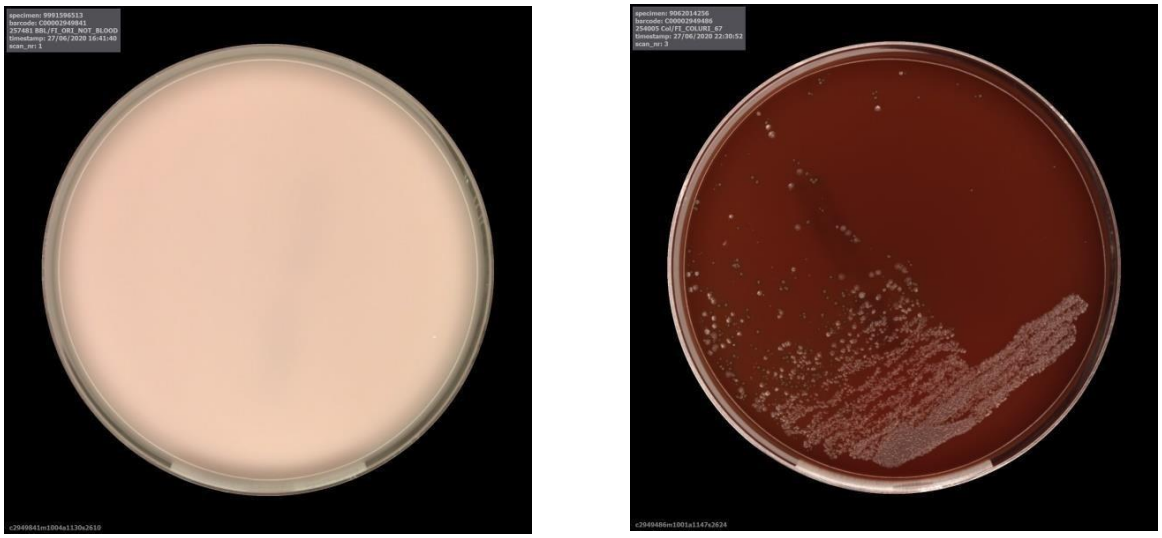
В данном случае у нас имеется множество изображений чашек Петри, разделенных на выросшие и невыросшие. Каждый образец фотографируется с помощью BD Kiestra через разные промежутки времени, то есть у нас по каждому образцу имеется папка, в которой лежат изображения в моменты t_0, t_1, t_2 и тд.

Процесс разметки происходит следующим образом. Врач смотрит на изображение в момент t_0, например, не может сделать вывод. И далее анализирует имеющиеся изображения до того момента, пока он не будет точно готов сделать вывод о том, что здесь что-то выросло. Мы обладаем знанием, в какой момент врач вынес окончательный вердикт. На основе этого вердикта мы и создаем датасет. Если врач, например, делает окончательный вывод в момент времени t_2, то мы t_0 и t_1 помечаем как датасет для раннего обнаружения, а t_2 и дальнейшие моменты как датасет с достоверными данными. И добавляем к датасету все невыросшие изображения.

Рисунок 6: *Схема разделения всего временного ряда на два датасета (для раннегообнаружения и «достоверные данные»).*

У нас есть данные за одиннадцать месяцев, суммарно достоверных изображений около 160 тысяч. Соотношение классов невыросшие/выросшие: 1.1/1
Так же все изображения разделяются по типам сред, в которых развиваются бактерии. Среды бывают белые (хромогенные) и красные (на основе крови животных).

Рисунок 7: *Изображения исходных данных. Слева «невыросший» chromAgar, справа «выросший» bloodAgar.*



Исходный размер изображений 2000x2000, для того чтобы обработать изображения такого размера нужны огромные размеры оперативной памяти и очень много времени. Для того, чтобы ускорить процесс обучения и не создавать лишней нагрузки на кластер, изображения изначально были уменьшены до разрешения 500x500, а так же были обрезаны по краям чашек Петри. При работе со Swin Transformer они составляют 224x224.

Таблица 2: *Описание данных, параметры*

Количество «достоверных данных»	160 тысяч изображений. 11 месяцев
Количество данных для «раннего обнаружения»	90 тысяч изображений. 11 месяцев
Объем всех данных	456Гб
Соотношение классов (0:1)	1.1/1
Исходный размер изображений	2000x2000
Итоговый размер изображений	500x500
Средняя длина временного ряда	3.09 изображений
Вывод врача (в среднем)	изображение 2.7

В среднем длина одной временной цепочки равна 3.09 изображений, врач обычно делает вывод достаточно поздно (3 изображения это 18 - 32 часа с момента засеивания анализа, что уже достаточно долго). При раннем обнаружении мы стараемся делать выводы уже по первому изображению, но при таком подходе может быть уже недостаточно только одних изображений. В дальнейшем есть планы добавить дополнительные знания модели и слегка перестроить ее архитектуру.

Проведение экспериментов

Эксперименты в рамках курсовой работы:

Таблица 3: *Результаты экспериментов для задач раннего обнаружения и «имитации врача». Для модели Xception (результаты курсовой работы).*

Среда	**Описание**	**Accuracy**	**Recall**
ChromAgar	«Имитация врача». Обучение на 6 месяцах. Валидация на 1 месяце. Штраф взвешенный. Достоверные данные.	**0.987**	0.987
BloodAgar	«Имитация врача». Обучение на 6 месяцах. Валидация на 1 месяце. Штраф одинаковый. Достоверные данные.	0.976	0.984

ChromAgar	«Раннее обнаружение». Обучение на 6 месяцах. Валидация на 7 месяцах. Штраф одинаковый. Раннее обнаружение.	**0.953**	0.953
BloodAgar	«Раннее обнаружение». Обучение на 6 месяцах. Валидация на 7 месяце. Штраф одинаковый. Раннее обнаружение.	0.947	0.947
ChromAgar	«Имитация врача». Обучение на 6 месяцах. Валидация на 1 месяце. Штраф повышенный для ошибок второго рода. Достоверные данные.	0.982	**0.994**

Метрика Accuracy представлена с весами равными отношению невыросших к выросшим (1.1/1).

При обучении на достоверных данных использовались около 47 тыс. изображений для обучения и 8тыс. для валидации.

При раннем обнаружении модель, обученная на достоверных данных, валидировалась на 30 тыс. данных, в которых специалисты еще не сделали окончательных выводов. В связи с тем, что часть изображений выросших в датасете «ранее обнаружение» не имеет каких-либо признаков роста, как следствие совершается большое количество ошибок. Итог: делается вывод о том, что необходимо добавить в нейросеть дополнительные знания, помимо изображения, а так же научить нейросеть отвечать не только выросло/не выросло, а так же «не знаю», тем самым проимитировав действия врача, но при этом научиться делать выводы все равно раньше, чем специалист.

При обучении месяца в модель поступали постепенно, на каждом месяце происходило обучение на 5 эпохах (при более высоких значениях происходило переобучение модели). Использовался оптимизатор Adam [17].

Обучение происходило на сервере nvidia DGX-2, что помогло значительно уменьшить время обучения за счет использования GPU для обработки изображений и работы с нейросетями. При среднем размере изображения 500x500 обучение одной эпохи занимало около 200 секунд на самых больших месяцах. В среднем время обучения одной нейросети занимает около 1-1.5 часов.

Было обнаружено, что модель для ChromAgar обучается лучше и показывает более

устойчивые результаты, чем для BloodAgar при всех проводимых экспериментах. Скорее всего, это связано в первую очередь с меньшим количеством артефактов на изображениях данной среды.

За счет добавления повышенного штрафа за ошибки второго рода, удалось сократить количество ошибок второго рода до минимума (recall 0.994), при небольшой потере точности, такая модель уже вполне способна «соперничать» со специалистами, так как случаи, когда болезнь не будет обнаружена практически отсутвуют.

Таблица 4: *Зависимость значений метрик от коэффициента штрафа. Модель chromAgar.*

Коэффициент	Accuracy	Precision	Recall
0.1	0.9784	0.9929	0.9703
0.5	0.9856	0.9926	0.9829
1.0	0.9850	0.9928	0.9815
1.5	**0.9876**	**0.9938**	0.9850
3.0	0.9876	0.9934	0.9854
5.0	0.9854	0.9861	0.9892
50.0	0.9851	0.9855	0.9894
200.0	0.9820	0.9761	0.9939
1000.0	0.9687	0.9547	**0.9941**

К < 1 - больший штраф за ошибки первого рода
К = 1 - одинаковый штраф
К > 1 - больший штраф за ошибки второго рода

В ходе анализа изображений, на которых ошибалась нейросеть, были выявлены ошибки в разметке, которые вместе с присутствующими артефактами занижают качество модели. Решением этой проблемы является постепенная переразметка данных, за счет обнаружения ошибок с помощью нейросети.

Было разработано несколько скриптов для обучения нейросетей, для обучения и

валидации на одной месяце, на всех месяцах, для разных сред, для раннего обнаружения и для достоверных данных. Так же было написано несколько Jupyter Notebook для анализа тех изображений, на которых ошибается нейросеть, так же для предобработки изображений, и большой ноутбук для приведения имеющихся данных к нужному виду, чтобы оформить датасет в том виде, в котором он описан ранее.

Данные были упакованы в numpy файлы, и именно они уже поступали на сервер, работа велась с ними, а не с изначальными фотоизображениями. Для работы с сервером использовались такие приложения как WINSCP и Putty, а так же возможности командной строки. На сервере nvidia DGX-2 вся работа велась с помощью контейнеров singularity, в которых находились все необходимые библиотеки для работы с данными.

Эксперименты в рамках дипломной работы:

1. LightGBM (только неграфические параметры):
   Balanced Accuracy = 0.6324
2. Лучшее решение:
   Swin-B Transformer (изображение + неграфические параметры):
   Balanced Accuracy = 0.987
   Confusion matrix:

true \ model	не выросло	выросло
не выросло	**24651 (98.49%)**	377(1.51%)
выросло	244(1.08%)	**22431(98.92%)**

Заключение

По итогам дипломеной работы были достигнуты следующе результаты.

1. Pipeline переписан и переведен с Keras на Pytorch для создания модульности и простоты воспроизведения результатов.
2. Обучена модель, основанная на SOTA решении в рамках многих задач CV, Swin-Transformer, в архитектуру модели добавлены неграфические признаки.
3. Выявлены врачебные ошибки (833 изображения) в существующей разметке.
4. Исследовано влияние неграфических признаков на целевую переменную.
5. Разработан экспериментальный стенд, отображающий текущие показатели метрик и визуализирующий ошибки нейросети.
   Объем 1100 строк кода

Рисунок 8: *Примеры врачебных ошибок. Модель определила рост бактерий, а изображения были помечены как «невыросшие».*

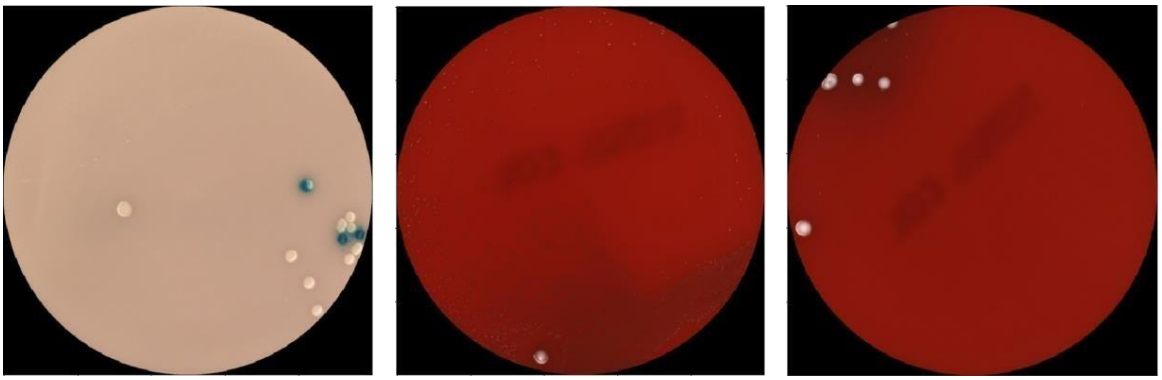
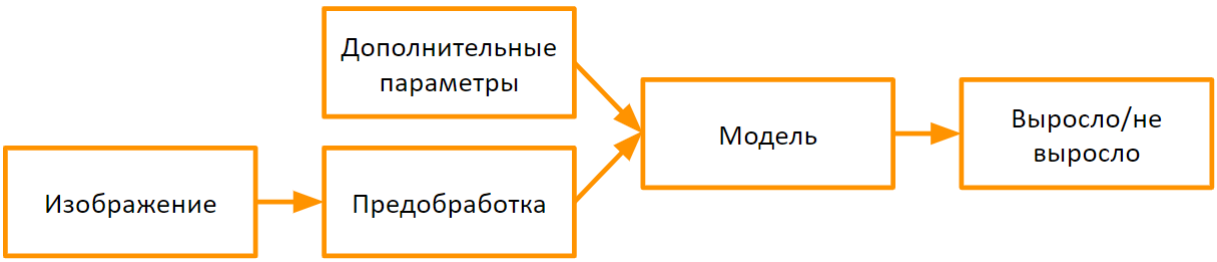


Рисунок 9: *Структура демонстрационного стенда*



Дальнейшие планы

1. Добавление сегментации области, где обнаружен рост бактерий.
2. Создание Hybrid-Swin-Transformer для увеличения размера входного изображения.
3. Подробные исследования в сторону раннего обнаружения и как следствие:
4. Расчет минимально необходимого времени повторного фотографирования для задачи оценки наличия изменений в росте бактерий.
5. Перевод модели с "выросло/не выросло" на "выросло/не выросло/ не знаю" для имитации принятий решений как врач.

Список литературы

[1]. Alessandro Ferrari, Stefano Lombardi, Alberto Signoroni, “Bacterial colony counting with Convolutional Neural Networks in Digital Microbiology Imaging”

[2]. M.Talo, “An Automated Deep Learning Approach for Bacterial Image Classification”

[3]. Hongda Wang, Hatice Ceylan Koydemir, Yunzhe Qiu, Bijie Bai, Yibo Zhang, Yiyin Jin, Sabiha Tok, Enis Cagatay Yilmaz, Esin Gumustekin, Yair Rivenson, Aydogan Ozcan,

“Early-detection and classification of live bacteria using time-lapse coherent imaging anddeep learning”

[4]. Xueyang Fu, Jiabin Huang, Xinghao Ding, Yinghao Liao, John Paisley “Clearing theSkies: A deep network architecture for single-image rain removal”

[5]. Eino-Ville Talvala, Andrew Adams, Mark Horowitz, Marc Levoy, “ Veiling Glare in High Dynamic Range Imaging”

[6]. Wenyao Xia, Elvis C. S. Chen, Stephen Pautler, and Terry M. Peters, Fellow, “A Global Optimization Method for Specular Highlight Removal from A Single Image”

[7]. F. Chollet, Xception: Deep learning with depthwise separable convolutions, in: Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, 2017.

[8]. Weihong Ren, Jiandong Tian, Yandong Tang, “ Specular Reflection Separation With Color-Lines Constraint”

[9]. Qingxiong Yang, Jinhui Tang, Narendra Ahuja, “Efficient and Robust Specular Highlight Removal”

[10]. Antonio Rogriguez-Sanchez, Daly Chea, George Azzopardi, Sebastian Stabinger, “A deep learning approach for detecting and correcting highlights in endoscopic images”

[11]. Dongsheng An∗ , Jinli Suo∗ , Xiangyang Ji, Haoqian Wang and Qionghai Dai, “Fast and High Quality Highlight Removal from A Single Image”

[12]. Qingxiong Yang, Shengnan Wang, Narendra Ahuja, “Real-Time Specular Highlight Removal Using Bilateral Filtering”

[13]. Pan, S.J. and Yang, Q., “A survey on transfer learning”

[14]. Nie, D., Shank, E.A. and Jojic, V., “A deep framework for bacterial image segmentation and classification.”

[15]. Mohamed, B.A. and Afify, H.M., “Automated classification of Bacterial Images extracted from Digital Microscope via Bag of Words Model.”

[16]. Huang, G., Liu, Z., van der Maaten, L. & Weinberger, K. Q. “Densely Connected Convolutional Networks.”

[17]. Kingma, D. P. & Ba, J. “Adam: A Method for Stochastic Optimization”
[18]. A. Ferrari, A. Signoroni, “Multistage classification for bacterial colonies recognition on solid agar images”
[19]. Gilles Louppe, “Understanding Random Forests: From Theory to Practice”
[20]. Nasip, Ö.F. and Zengin, K., “Deep Learning Based Bacteria Classification.”
[21]. K. He, X. Zhang, S. Ren, and J. Sun. “Deep residual learning for image recognition”