

АННОТАЦИЯ

Развитие области медицинской визуализации — важная задача при растущей потребности в автоматизированной, быстрой и эффективной диагностике. Традиционно количество клеток крови определяется с помощью гемоцитометра при использовании дополнительного лабораторного оборудования и химических соединений, но данный метод занимает много времени и является трудоемким. В работе исследовано применение методов машинного обучения к задаче идентификации и классификации клеток крови для увеличения скорости распознавания без ухудшения качества.

ABSTRACT

The development of the field of medical imaging is an important task with the growing need for automated, fast and efficient diagnostics. Traditionally, the number of blood cells is determined using a hemocytometer using additional laboratory equipment and chemical compounds, but this method is time consuming and laborious. The paper studies the application of machine learning methods to the problem of identification and classification of blood cells to increase the recognition rate without compromising quality.

СОДЕРЖАНИЕ

АННОТАЦИЯ	2
ВВЕДЕНИЕ	3
1 ПРЕДПРОЕКТНОЕ ИССЛЕДОВАНИЕ	5
1.1 Анализ предметной области	5
1.2 Глубокие нейронные сети	7
1.3 Сверточные нейронные сети	15
1.4 Машинное обучение в системах классификации клеток костного мозга	
18	
2 МАТЕРИАЛЫ И МЕТОДЫ	22
2.1 Материалы	22
2.2 Предлагаемые методы и подходы	25
2.2.1 Фреймворк глубокого обучения	25
2.2.2 Предобученные модели на ImageNet	28
2.2.3 Интерпретационное машинное обучение	30
2.2.4 Модель обнаружения объектов YOLO	32
2.2.5 Клиент-серверная архитектура	36
2.2 Инструменты реализации	38
3 ПРОЕКТНОЕ РЕШЕНИЕ	40
3.1 Концептуальная модель и алгоритмы	40
3.2 Модель базы данных	45
3.3 Архитектура системы	47
3.5 Взаимодействие с пользователем	48
4 ЭКСПЕРИМЕНТ И РЕЗУЛЬТАТЫ	52
4.1 Подготовка данных	52
4.2 Результаты экспериментов	54
ЗАКЛЮЧЕНИЕ	59
СПИСОК ЛИТЕРАТУРЫ	61

ВВЕДЕНИЕ

Клинический анализ крови позволяет выявлять различные заболевания на основе отклонений от референсных значений показателей компонентов клеточного состава крови. В гематологии особенно важно соотношение различных типов клеток при случае диагностики гематологических заболеваний: гемобластозов (например, лимфом и лейкозов), анемий и других патологий. Помимо стандартного забора периферической крови, в случае обнаружения существенных отклонений от референсных значений показателей клеточного состава, также используется анализ клеток красного костного мозга, что позволяет точнее установить заболевание и причины его возникновения.

Дифференциальный подсчет клеток в аспирате красного костного мозга и мазках периферической крови имеет решающее значение для классификации гематологических заболеваний. Ручной подсчет клеток в камере Горяева считается стандартом, но он трудоемок, требует много времени и приводит к ошибкам, вызванных человеческим фактором. Цифровая визуализация патологии, основанная на алгоритмах машинного обучения, представляет собой многообещающую новую технологию для этой решения существующих проблем ручного анализа клеток. При этом методы глубокого обучения позволяют обнаруживать клетки с высокой точностью.

Цель выпускной квалификационной работы работы заключается в реализации методов машинного обучения для создания модели классификации клеток крови и костного мозга, а также проведении экспериментов с изображениями клеток крови и костного мозга.

Задачи работы:

1. Проанализировать и выбрать подходы к реализации методов классификации.
2. Реализовать алгоритмы и методы выбранных методов машинного обучения для классификации клеток крови и костного мозга.
3. Создать базу данных с изображениями клеток крови и костного мозга.
4. Провести эксперименты по классификации клеток крови и клеток гемопоэтического ряда красного костного мозга путем анализа их гистологических снимков выбранными методами машинного обучения, а также провести эксперименты по выявлению признаков, которые используются моделью классификации.
5. Выбрать методы с наилучшим качеством распознавания клеток крови и клеток гемопоэтического ряда красного костного мозга путем проведения сравнительного анализа методов.
6. Реализовать клиент-серверное приложение и графический интерфейс пользователя для взаимодействия с системой.

1 ПРЕДПРОЕКТНОЕ ИССЛЕДОВАНИЕ

1.1 Анализ предметной области

Обнаружение и распознавание объектов в данных — задача, которая появляется в различных областях науки и инженерии. Обычно, данная задача решается статистическими методами. Эмпирически выявлено, что данный подход предоставляет наилучшие результаты в большинстве задач обнаружения объектов. Это связано с развитием машинного обучения и анализа данных, и, в особенности, с развитием глубокого обучения, в последнее десятилетие. Машинное обучение активно эксплуатирует описываемый подход, и представляет собой междисциплинарную область, которая объединяет математическую статистику, теорию оптимизации, компьютерную инженерию, численные методы и инженерию данных [1].

Методы машинного обучения универсальны, но требуют знаний в предметной области, в которой эти методы используются. Знания в предметной области необходимы, чтобы можно было получать и делать безошибочные объективные выводы из полученных после использования методов результатов. Адекватный к предметной области выбор признаков, способ сбора, обработки и хранения данных позволяет достичь наилучших результатов для решения поставленной задачи [2].

Данные для распознавания объектов, как для любых других задач машинного обучения, могут содержать как размеченные, так и неразмеченные объекты. Для размеченных данных используются методы обучения с учителем, для неразмеченных — методы обучения без учителя. Также существуют промежуточные парадигмы — методы с частичным привлечением учителя и методы обучения с подкреплением.

При распознавании объектов методы обучения с учителем и без учителя можно свести к методам классификации и кластеризации. Классификация — это способ использования готовой натренированной модели машинного обучения с учителем для распределения данных по заранее определенным категориям. Кластеризация — это использование моделей машинного обучения для обнаружения кластеров, в которых объекты наиболее похожи друг на друга.

1.2 Глубокие нейронные сети

Классификация с несколькими метками предполагает прогнозирование двух и более меток классов. В отличие от обычных задач классификации, в которых метки классов являются взаимоисключающими, для классификации с несколькими метками требуются специализированные алгоритмы машинного обучения, которые поддерживают прогнозирование нескольких взаимно не исключающих классов или «меток».

Глубокие нейронные сети — пример алгоритма, который изначально поддерживает задачи классификации с несколькими метками. Модели нейронных сетей для задач классификации с несколькими метками можно легко определить и оценить с помощью различных библиотек.

Глубокое обучение является частью более широкого семейства методов машинного обучения, основанных на искусственных нейронных сетях с репрезентативным обучением. Обучение может быть контролируемым, частично контролируемым или неконтролируемым.

Архитектуры глубокого обучения, такие как глубокие нейронные сети, нейронные сети на графах, рекуррентные нейронные сети и сверточные нейронные сети, применялись в таких областях, как компьютерное зрение, распознавание речи, обработка естественного языка, машинный перевод, биоинформатика, дизайн лекарств, медицина, проверка материалов и настольных игр, где они дали результаты, сравнимые, а в некоторых случаях и превосходящие возможности экспертов [3].

Искусственные нейронные сети (ИНС) были вдохновлены обработкой информации и распределенными коммуникационными узлами в биологических системах. При этом, конечно, ИНС имеют отличия от

биологического мозга. В частности, нейронные сети, как правило, статичны и символичны, в то время как биологический мозг большинства живых организмов является динамическим и пластичным [4].

ИНС основана на наборе связанных блоков или узлов, называемых искусственными нейронами, которые в общих чертах моделируют нейроны в биологическом мозге. Каждое соединение, как синапсы в биологическом мозге, может передавать сигнал другим нейронам. Искусственный нейрон, который получает сигнал, затем обрабатывает его и может сигнализировать подключенным к нему нейронам. «Сигнал» в соединении — это действительное число, и выходной сигнал каждого нейрона вычисляется некоторой нелинейной функцией суммы его входов. Связи называются ребрами. Нейроны и ребра обычно имеют вес, который корректируется по мере обучения. Вес увеличивает или уменьшает силу сигнала в соединении. Нейроны могут иметь такой порог, что сигнал отправляется только в том случае, если совокупный сигнал пересекает этот порог. Обычно нейроны объединены в слои. Разные слои могут выполнять разные преобразования на своих входах. Сигналы проходят от первого слоя (входной) к последнему (выходному), возможно, после многократного прохождения слоев (Рисунок 1).

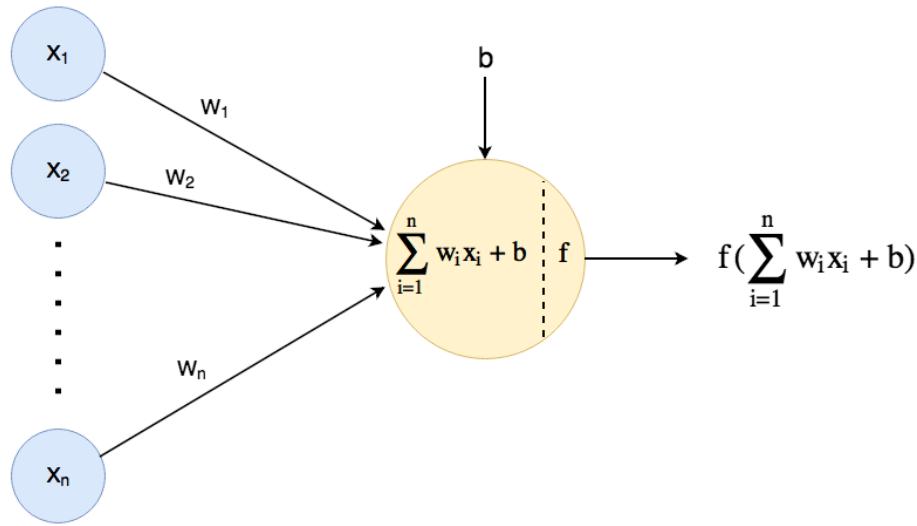


Рисунок 1. Иллюстрация принципа работы функции активации в одном нейроне

Прилагательное «глубокий» в глубоком обучении относится к использованию нескольких уровней в сети. Ранние работы показали, что линейный перцептрон не может быть универсальным классификатором, но сеть с неполиномиальной функцией активации с одним скрытым слоем неограниченной ширины имеет свойства универсального классификатора.

Глубокое обучение — это современная разновидность методов машинного обучения, которая связана с неограниченным количеством слоев ограниченного размера, что допускает практическое применение и оптимальную реализацию, при этом сохраняется теоретическая универсальность в различных условиях. В глубоком обучении слоям также разрешается быть неоднородными и широко отклоняться от биологических коннекционистских моделей ради эффективности, обучаемости и понятности, откуда и возникла «структурированная» часть [5].

В искусственных нейронных сетях функция активации узла определяет вывод этого узла с учетом ввода или набора входов. Стандартную интегральную схему можно рассматривать как цифровую сеть функций активации, которая может быть «ВКЛ» (1) или «ВЫКЛ» (0), в зависимости от входа. Это похоже на линейный персепtron в нейронных сетях. Однако только нелинейные функции активации позволяют таким сетям решать нетривиальные задачи, используя лишь небольшое количество узлов, и такие функции активации называются нелинейностями.

$$\begin{aligned}
 X &= \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} & W &= \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \\ W_{31} & W_{32} \\ W_{41} & W_{42} \end{bmatrix} & b &= \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \\
 \text{Input Data} & & \text{Randomly Initialized Weight Matrix} & & \text{Randomly Initialized bias Matrix}
 \end{aligned}$$

Рисунок 2. Концептуальное представление структуры используемых данных

Наиболее распространенные функции активации можно разделить на три категории: функции гребня, радиальные функции и функции складывания.

На Рисунках 3 и 4 изображена функция активации, которая используется больше всего в практике работы с нейронными сетями, однако данная функция иногда приносит не самые оптимальные результаты.

$$\phi(\mathbf{v}) = \max(0, a + \mathbf{v}' \mathbf{b})$$

Рисунок 3. Функция ReLu()

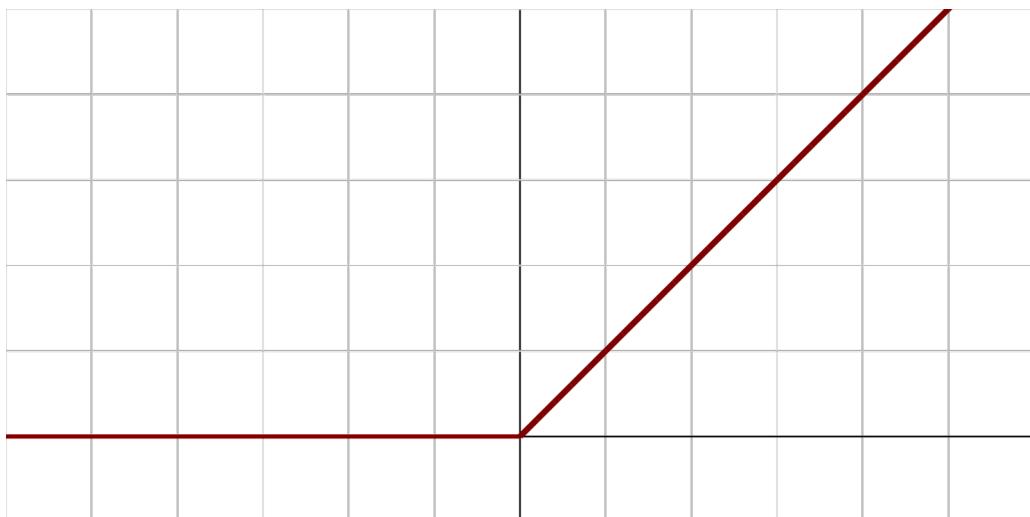


Рисунок 4. График функции активации ReLu()

Обучение модели означает поиск хороших значений для всех весов и смещений из помеченных примеров. При обучении с учителем алгоритм машинного обучения строит модель, исследуя множество примеров X и пытаясь найти модель, которая минимизирует потери при поиске истинного значения Y . Другими словами, происходит поиск функции $h: x \rightarrow y$, именуемой гипотезой. Этот процесс называется минимизацией эмпирического риска. На Рисунке 5 представлено математическое выражение минимизации эмпирического риска. Функция $L()$ – функция потерь (от слова loss). Функция $P()$ – совместное распределение множества X и Y . Соответственно итоговое выражение представляет собой поиск математического ожидания от разницы между предсказанным и истинным значением [6].

$$R(h) = \mathbf{E}[L(h(x), y)] = \int L(h(x), y) dP(x, y).$$

Рисунок 5. Математическое выражение минимизации эмпирического риска.

Потеря – это наказание за плохой прогноз. То есть потеря – это число, показывающее, насколько плохим был прогноз модели на одном примере. Если прогноз модели идеален, потери равны нулю; в противном случае потери больше. Цель обучения модели, как уже было сказано, – найти набор весов и смещений, которые в среднем имеют низкие потери во всех примерах. Например, на Рисунке 6 слева показана модель с высокими потерями, а справа – модель с низкими потерями.

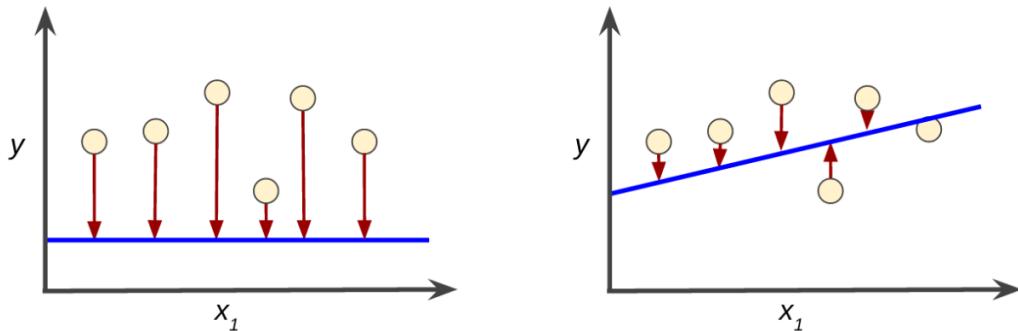


Рисунок 6. Влияние двух функций потерь с различными уровнями потерь

Наиболее известной функцией потерь является функция квадратичной ошибки, изображенная на Рисунке 7.

$$MSE = \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n}$$

Рисунок 7. Квадратичная функция потерь

Квадратичная функция потерь обычно используется в задачах линейной регрессии, однако не является универсальной, поэтому существует еще ряд различных функций.

Ключевой алгоритм для обучения современных ИНС — алгоритм обратного распространения ошибки. Этот алгоритм позволяет достичь оптимального распределения весов внутри нейронной сети за счет использования алгоритмов оптимизации, например таких алгоритм градиентного спуска или метод стохастического градиентного спуска. Результаты алгоритма предоставляют возможности к поиску разделительной гиперплоскости в многомерном пространстве признаков при классификации объектов.

Нейронную сеть можно представить как комбинацию композиций функций активации и матричного умножения (Рисунок 8).

$$g(x) := f^L(W^L f^{L-1}(W^{L-1} \dots f^1(W^1 x) \dots))$$

Рисунок 8. Математическое представление ИНС

Обратное распространение вычисляет градиент в весовом пространстве нейронной сети с прямой связью по отношению к функции потерь. На Рисунке 9 представлено математическое выражение алгоритма обратного распространения ошибки. Первое выражение представляет собой скалярное произведение градиента функции потерь от нейрона a с производной функции активации от результата сложения всех значений нейронного последнего слоя L. Дальнейшие выражения представляют собой продолжение операций на последующих нейронах ИНС [7].

$$\delta^L = \nabla_a C \odot \sigma'(z^L) \quad (\text{BP1})$$

$$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l) \quad (\text{BP2})$$

$$\frac{\partial C}{\partial b_j^l} = \delta_j^l \quad (\text{BP3})$$

$$\frac{\partial C}{\partial w_{j,k}^l} = a_k^{l-1} \delta_j^l \quad (\text{BP4})$$

Рисунок 9. Математическое выражение алгоритма обратного распространения ошибки

1.3 Сверточные нейронные сети

В глубоком обучении сверточная нейронная сеть (CNN или ConvNet) представляет собой класс глубокой нейронной сети, наиболее часто применяемой для анализа визуальных образов. Они также известны как инвариантные к сдвигу или пространственно-инвариантные искусственные нейронные сети (SIANN), основанные на архитектуре ядер свертки или фильтров с общим весом, которые скользят по входным объектам и обеспечивают эквивалентные отклики перевода, известные как карты признаков. Большинство сверточных нейронных сетей только эквивариантны, а не инвариантны, по отношению к трансляции. У них есть приложения для распознавания изображений и видео, рекомендательных систем, классификации изображений, сегментации изображений, анализа медицинских изображений, обработки естественного языка, интерфейсов мозг-компьютер и финансовых временных рядов [8].

CNN — это регуляризованные версии многослойных перцептронов. Многослойные перцептроны обычно означают полностью связанные сети, то есть каждый нейрон в одном слое связан со всеми нейронами следующего слоя. «Полная связность» этих сетей делает их склонными к переобучению данных. Типичные способы регуляризации или предотвращения переобучения включают: штрафные параметры во время обучения (например, снижение веса) или ограничение возможности подключения (пропущенные соединения, выпадение и т. д.). CNN используют другой подход к регуляризации: они используют преимущества иерархической структуры данных и собирают паттерны возрастающей сложности, используя более мелкие и простые паттерны, полученные в их фильтрах (Рисунок 5).

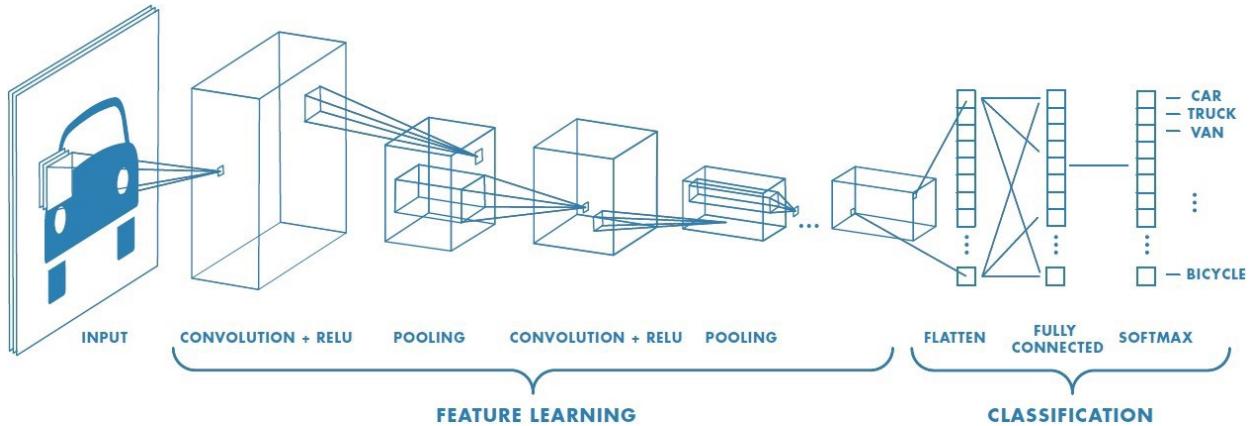


Рисунок 5. Концептуальная модель сверточной нейронной сети

Основная операция в сверточных нейронных сетях — операция свертки. Данная операция описана на Рисунке 6.

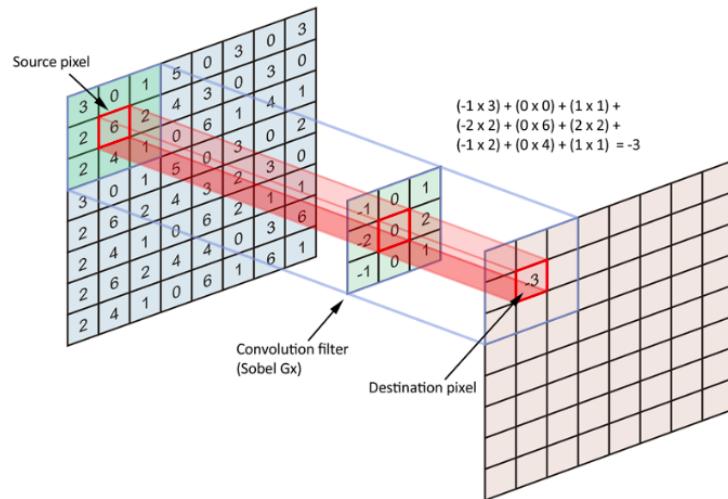


Рисунок 6. Логика работы операции свертки на матрице

В результате использования операции свертки на каждом слое сверточной нейронной сети, на выходе каждого слоя появляются карты признаков (feature map) (Рисунок 7) [9].

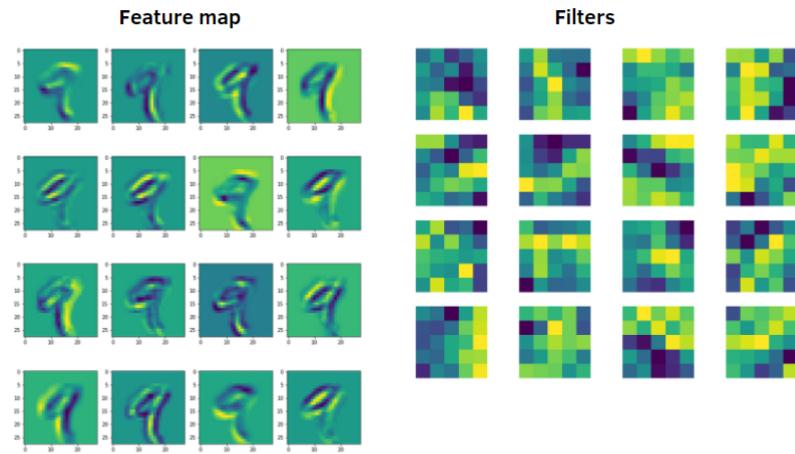


Рисунок 7. Результат использования фильтров (сверток) на изображение с числом девять

Пулинг — еще одна базовая операция в сверточных нейронных сетях. Такая операция позволяет понизить размерность данных на последующих слоях сети, что позволяет повысить производительность и уменьшить количество потребляемых ресурсов (Рисунок 8).

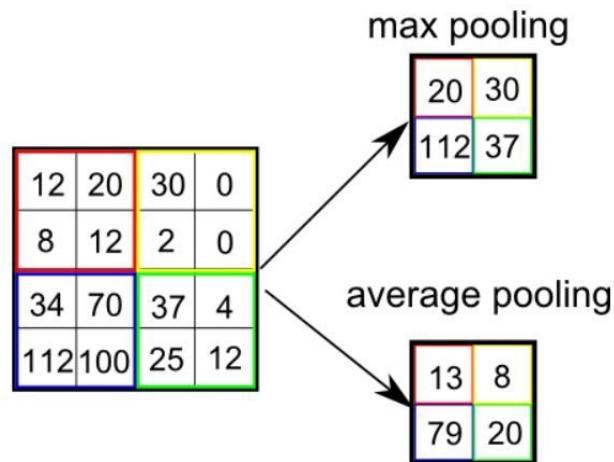


Рисунок 8. Максимальный пулинг и усредненный пулинг

1.4 Машинное обучение в системах классификации клеток костного мозга

Точность классификации естественных изображений за последние несколько лет значительно улучшилась благодаря все более широкому использованию сверточных нейронных сетей (CNN) [10]. Тем временем эта технология также применялась для интерпретации различных медицинских изображений, включая обнаружение митоза в гистологических срезах рака молочной железы, обнаружение рака кожи, оценку маммограммы, и цитологическую классификацию в периферической крови. Однако, успешное использование CNN для классификации изображений обычно зависит от наличия достаточного количества высококачественных данных – изображений и высококачественных аннотаций, доступ к которым может быть затруднен из-за затрат, связанных с получением меток медицинскими экспертами. Это особенно важно в таких ситуациях, как цитоморфологическое исследование костного мозга, где нет технического золотого стандарта и необходимы специалисты-исследователи, чтобы разметить данные для обучения нейронной сети.

На основании CNN Кермани и соавторы создали диагностический инструмент для скрининга пациентов с распространенными излечимыми заболеваниями сетчатки, приводящими к слепоте, который продемонстрировал эффективность классификации, сравнимую с эффективностью экспертов-людей. Используя CNN, можно установить сопоставление взаимосвязей между изображением клеток костного мозга и классификацией, чтобы реализовать эффективное распознавание морфологии костного мозга [11].

В 2017 была опубликована статья [12], в которой авторы представили подход для обнаружения и автоматического подсчета красных кровяных клеток (эритроцитов) на изображениях мазков крови. Для анализа использовались изображения мазков крови, окрашенных по Романовскому-Райту, здоровых и больных онкологией пациентов из Индийского Медицинского Колледжа Кастворбы и Атласа Гематологии. Авторы не предоставили сведений о количестве изображений, а также не предоставили ссылок на исходные источники данных, что предварительно понижает корректность разработанного подхода [13], [14]. Полученные изображения были преобразованы в цветовое пространство LAB. Авторы смотивировали преобразование тем, что LAB пространство ближе к восприятию человеческого глаза, однако не предоставили доказательств о связи данного цветового пространства к качеству обнаружения и классификации каких-либо объектов на изображении. После изменения цветового пространства изображения были преобразованы в оттенки серого. Данное преобразование понижает размерность признакового пространства, тем самым может привести к понижению качества классификации [15]. Авторы также использовали бинаризацию изображений на основе метода порога Оцу и эквализации значений каждого пикселя, что также понижает размерность признакового пространства и потенциально снижает точность обнаружения и классификации объектов на изображениях. Сегментация изображений проводилась при помощи метода преобразования водораздела (watershed transform). Изображения были далее отфильтрованы медианным фильтром 2x2 для снижения шума. На области полученных изображений был использован алгоритм для вырезания белых кровяных клеток при помощи

алгоритма сегментации k-medoids и выделенных свойств изображений. Для подсчета эритроцитов использовалось круговое преобразование Хафа.

В статье [16] 2018 года был проведен обзор методов обработки изображений и методов машинного обучения для морфологического анализа кровяных клеток: эритроцитов, тромбоцитов и лейкоцитов. Авторы статьи обратили внимание на необходимость рассматривать более специфичные типы клеток крови, которые зависят от степени дифференцировки и наличия патологии, а также предложили подход, в котором классификация проходит в несколько этапов: от групп клеток к специфичных клеткам данного типа. Многие работы, рассмотренные в статье, добились высокой точности, и основным методом был метод опорных векторов. Авторы также обратили внимание на необходимость создания единого хранилища данных изображений кровяных клеток, что позволило бы улучшить качество систем обнаружения и классификации клеток крови.

В статье [15] авторы предлагают новую сеть глубокого обучения для помощи в обнаружении и классификации эритроидных и миелоидных клеток на гистопатологических изображениях биопсии костного мозга. Чтобы повысить точность обнаружения и классификации, предлагаемая нейронная сеть включает новый механизм выбора окрестности, который учитывает область, окружающую центр обнаруженной клетки.

В другой работе [17] авторы провели сравнение классического отбора признаков при помощи методов обработки изображений, и автоматического – на основе сверточной нейронной сети AlexNet [18]. Авторы сообщили, что сверточная нейронная сеть отбирает лучшие признаки, что повышает конечную точность классификации по сравнению с классическими методами.

В статье [19] 2019 года для обнаружения и подсчета кровяных клеток: эритроцитов, лимфоцитов и тромбоцитов, авторы использовали модель обнаружения объектов YOLO [20] и достигли точности 96%.

В исследовании авторы использовали алгоритм обнаружения клеток, основанный на быстрой регионной сверточной сети (Faster Region-Based Convolutional Network).

Хотя CNN превзошли все классификаторы, полагающиеся на созданные вручную функции в широком диапазоне задач, структура их выходных данных обычно не поддается прямой интерпретации человеком. Для устранения этого недостатка были разработаны различные методы распознавания признаков, на основании которых модель классифицирует объекты. Таким образом признаки, выбранные сетью для классификации изображений, кажутся надежными и устойчивыми по отношению к некоторому шуму меток, которого нельзя избежать в методе, управляемом данными, полагающимся на экспертные аннотации.

Таким образом, создание эффективного алгоритма и обучение на большом количестве изображений может сломать зависимость анализа морфологии клеток костного мозга от человеческого распознавания и реализовать интеллектуальное распознавание.

2 МАТЕРИАЛЫ И МЕТОДЫ

2.1 Материалы

Гемоцитобласт — мультипотентная стволовая клетка, от которой могут дифференцироваться остальные типы клеток [21].

Все клетки крови делятся на три линии:

- Красные кровяные тельца, также называемые эритроцитами, являются клетками, переносящими кислород. Эритроциты функционируют и попадают в кровь. Количество ретикулоцитов, незрелых красных кровяных телец, дает оценку скорости эритропоэза.
- Лимфоциты — это краеугольный камень адаптивной иммунной системы. Они происходят от общих лимфоидных предшественников. Лимфоидная линия состоит из Т-клеток, В-клеток и естественных клеток-киллеров. Это лимфопоэз.
- Клетки миелоидного происхождения, которые включают гранулоциты, мегакариоциты и макрофаги, происходят от обычных миелоидных предшественников и участвуют в таких разнообразных ролях, как врожденный иммунитет и свертывание крови. Это миелопоэз.
- Гранулопоэз (или гранулоцитопоэз) — это процесс созревания гранулоцитов, за исключением тучных клеток, которые являются гранулоцитами с экстрамедуллярным созреванием.
- Мегакариоцитопоэз — это кроветворение мегакариоцитов.

Для анализа и разработки системы используется набор данных, взятый из открытых источников. Первый набор данных взят из архива изображений,

связанных с онкологическими заболеваниями (Cancer Image Archive), и представляет собой набор изображений различных типов клеток красного костного мозга, окрашенных по Романовскому, и включает 170000 изображений 21 типа клеток [10]. Набор данных был собран в Институте Мюнхенской лаборатории по изучению лейкемии с 961 пациента (Рисунок 8). Второй набор данных представляет собой набор изображений клеток периферической крови, состоящий из 17092 изображений 9 типов клеток. Набор был собран в Госпитальной Клинике Барселоны (Рисунок 9).

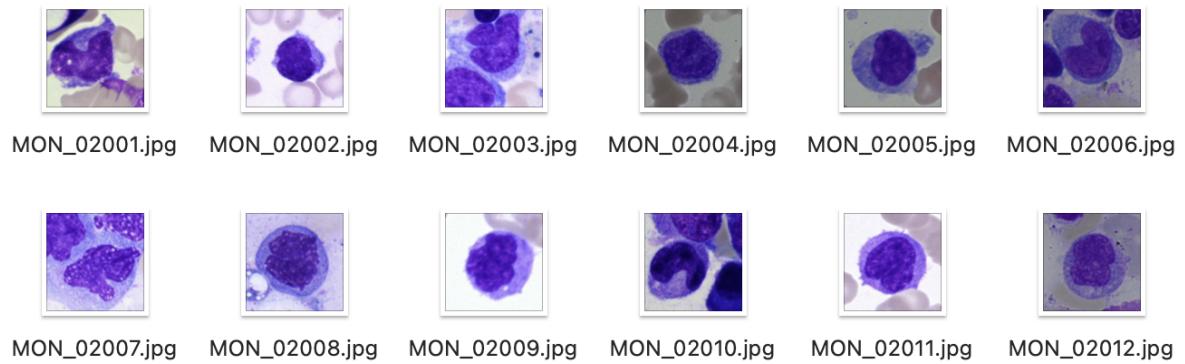


Рисунок 9. Изображения моноцитов из Cancer Image Archive

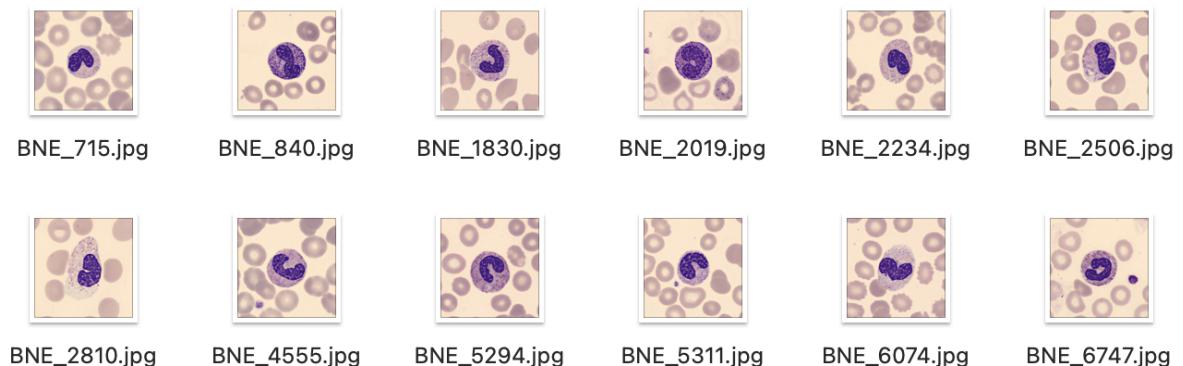


Рисунок 10. Изображения нейтрофилов из набора данных
Госпитальной Клиники Барселоны

Все наборы данных были обработаны и подготовлены для дальнейшего использования. В итоге получился набор со следующими 10 классами:

- Нейтрофилы
- Моноциты
- Бласти
- Клетки плазмы
- Гранулоциты
- Эритробласти
- Тромбоциты
- Лимфоциты
- Патогенные клетки (Лейкемия, разрушенные клетки и аномальные клетки)
- Другие клетки (неклассифицируемые клетки)

2.2 Предлагаемые методы и подходы

2.2.1 Фреймворк глубокого обучения

Для правильного выбора необходимо сравнить плюсы и минусы различных решений, оценить их пределы возможностей и узнать о лучших вариантах использования. TensorFlow был разработан Google и реализован на C++ [22]. Оболочка, которая может запускаться поверх TensorFlow, Theano или CNTK. Поддерживает широкий спектр слоев нейронных сетей, таких как сверточные слои, рекуррентные или плотные. PyTorch — преемник библиотеки Torch, написанной на Lua. Фреймворк был разработан Facebook и использовался Twitter, Salesforce, Оксфордским Университетом и многими другими компаниями. Следующий фреймворк, CNTK (Microsoft Cognitive Toolkit) представляет из себя открытый фреймворк для глубокого обучения, созданный для работы с действительно большими данным с поддержкой Python, C++, C# и Java [23]. Caffe — фреймворк, реализованный на C++, имеющий интерфейс на Python. Фреймворк поддерживает CNN и сети прямого распространения, а также подходит для тренировки моделей, обработке изображений и усовершенствования существующих нейронных сетей [24]. Theano — проект с открытым исходным кодом, основной разработчик которого — группа машинного обучения в Монреальском университете. 28 сентября 2017 года было объявлено о прекращении работы над проектом после выхода релиза 1.0, при этом обещано сохранение его минимальной поддержки в течение одного года [25].

В Таблице 1 представлен сравнительный анализ одних из самых популярных фреймворков для глубокого обучения.

Свойство	Caffe	Theano	TensorFlow	PyTorch	CNTK	Keras
Базовый язык	C++	Python	C++	Pytho	C++	Python
API	C++ Python	Python	C++ Python	Python	C++, C# Python	Python
Многоядерные CPU	+	+	+	+	+	+
GPU	+	+	+	+	+	+
Распределенное обучение	+	-	+	+	+	+
Разработчик	Центр компьютерного зрения и обучения Беркли	Университет Монреаля	Google	Facebook	Microsoft	Франсуа Шолле
Открытые коды	+	+	+	+	+	+
Обученные сети	+	-	+	+	+	+

Таблица 1 – Сравнительный анализ фреймворков глубокого обучения

Широкое распространение практического применения нейронных сетей является возможным благодаря наличию большого количества готовых решений для обучения глубоких нейронных сетей, фреймворков, позволяющих быстро и эффективно разрабатывать новые архитектуры, а также из-за возможности использования современных многоядерных процессоров, ускорителей вычислений GPU [26].

PyTorch — фреймворк машинного обучения для языка Python с открытым исходным кодом, созданный на базе Torch. Используется для решения различных задач: компьютерное зрение, обработка естественного языка. Разрабатывается преимущественно группой искусственного интеллекта Facebook. Также вокруг этого фреймворка выстроена экосистема, состоящая из различных библиотек, разрабатываемых сторонними командами: PyTorch Lightning и Fast.ai, упрощающие процесс обучения

моделей, Pyro, модуль для вероятностного программирования, от Uber, Flair, для обработки естественного языка и Catalyst, для обучения моделей глубокого обучения [8].

PyTorch предоставляет две основные высокоуровневые модели:

- Тензорные вычисления (по аналогии с NumPy) с развитой поддержкой ускорения на GPU
- Глубокие нейронные сети на базе системы autodiff

Таким образом, данный фреймворк был признан самым подходящим под цели и задачи работы.

2.2.2 Предобученные модели на ImageNet

Трансферное обучение (transfer learning) — концепция из области машинного обучения, которая представляет собой переиспользование результатов обучения из одной доменной области к другой. Например, знания, полученные при обучении распознаванию автомобилей, можно применить при попытке распознать грузовики. С практической точки зрения повторное использование или передача информации из ранее изученных задач для изучения новых задач может значительно повысить эффективность выборки агента обучения с подкреплением. Трансферное обучение является гибким, позволяя напрямую использовать предварительно обученные модели в качестве предварительной обработки извлечения признаков и интегрировать их в совершенно новые модели [11].

База данных ImageNet — проект по созданию и сопровождению массивной базы данных аннотированных изображений, предназначенная для отработки и тестирования методов распознавания образов и машинного зрения. По состоянию на 2022 год в базу данных было записано около десяти миллионов URL с изображениями, которые прошли ручную аннотацию для ImageNet, в аннотациях перечислялись объекты, попавшие на изображение, и прямоугольники с их координатами [18].

Аннотации на уровне самих изображений показывают наличие или отсутствие объекта данного класса (например, «на картинке имеется тигр» или «на картинке нет тигров»). На уровне объекта в аннотацию включается прямоугольник с координатами видимой части объекта. ImageNet использует вариант семантической сети WordNet для категоризации объектов, которая достаточно детализирована, например, породы собак представлены 120

классами. Каждому узлу сети WordNet сопоставлены сотни или тысячи изображений, но в среднем на 2016 год — около 500 изображений. На 2022 год в ImageNet 14 197 122 изображения, разбитых на 21 841 категорию [27].

Библиотека Pytorch имеет расширение для работы с предобученным на ImageNet нейронным сетям — torchvision. Расширение включает в себя набор различных моделей сверточных нейронных сетей с сохраненными весами внутри. В данной работе анализируется точность работы классификаторов следующих моделей: ResNet [28], MobileNetV3 [29], EfficientNetV2 [30], CoAtNet [31].

2.2.3 Интерпретационное машинное обучение

В последнее время предпринимаются попытки использования нейронных сетей, в особенности сверточных нейронных сетей, для обнаружения и классификации в медицинской сфере, однако, данные методы содержат общую для всей нейронных сетей проблему черного ящика. Для решения данной проблемы используется интерпретационное машинное обучение – направление в машинном обучении, ставящее цель определить поведение нейронной сети в процессе обучения и работы нейронной сети на анализируемых объектах. Методы интерпретационного машинного обучения привносят в систему обнаружения и классификации предсказуемость и интерпретируемость. Кроме этого, интерпретационные методы позволяют определить какие признаки на изображениях – наиболее важные, это особенно полезно при анализе количества и морфологического состояния клеток крови и костного мозга, другими словами, автоматизированная интерпретация позволяет получить новые знания о структуре анализируемых объектов и то, на что необходимо обращать внимание человеку, чтобы правильно отнести объект к нужному классу [32].

С увеличением сложности модели и, как следствие, отсутствием прозрачности, методы интерпретации модели становятся все более важными. Понимание моделей является как активной областью исследований, так и областью практического применения машинного обучения в различных отраслях.

Алгоритмы интерпретации разделены на три группы: первичная атрибуция, атрибуция слоев и атрибуция нейронов, которые определяются следующим образом:

- Первичная атрибуция: оценивает вклад каждой входной функции в выходные данные модели.
- Атрибуция слоя: оценивает вклад каждого нейрона в данном слое в выходные данные модели.
- Атрибуция нейронов: оценивает вклад каждой входной функции в активацию определенного скрытого нейрона.

Интегрированные градиенты представляют собой интеграл градиентов относительно входных данных на пути от заданной базовой линии до входных данных. Интеграл можно аппроксимировать с помощью квадратурного правила суммы Римана или Гаусса-Лежандра. Формально его можно описать следующим образом:

$$\text{IntegratedGrads}_i(x) := (x_i - x'_i) \times \int_{\alpha=0}^1 \frac{\partial F(x' + \alpha \times (x - x'))}{\partial x_i} d\alpha$$

Интегрированные градиенты по i -му измерению входных данных X .
Альфа — это коэффициент масштабирования. Уравнения скопированы из
оригинальной статьи.

Краеугольными камнями этого подхода являются две фундаментальные аксиомы, а именно чувствительность и инвариантность реализации.

Метод окклузий — это основанный на возмущении подход к вычислению атрибуции, включающий замену каждой смежной прямоугольной области заданной базовой линией/эталоном и вычисление разницы в результатах. Для функций, расположенных в нескольких регионах (гиперпрямоугольники), соответствующие выходные различия усредняются для вычисления атрибуции для этой функции. Затенение наиболее полезно в

таких случаях, как изображения, где пиксели в непрерывной прямоугольной области, вероятно, будут сильно коррелированы.

2.2.4 Модель обнаружения объектов YOLO

Современная область обнаружения объектов включает в себя набор концепций, которые необходимо определить перед описанием модели. .

Более ранние архитектуры для обнаружения объектов состояли из двух отдельных этапов — сети предложений регионов, которая выполняет локализацию объектов, и классификатора для обнаружения типов объектов в предлагаемых регионах. В вычислительном отношении они могут быть очень дорогими и поэтому не подходят для реальных приложений реального времени. Однократная детекция (single-shot detection) инкапсулирует как задачи локализации, так и задачи обнаружения в одну прямую работу сети, что приводит к значительно более быстрому обнаружению при развертывании на более легком оборудовании.

В задачах классификации изображений прогнозы основываются на окончательной сверточной карте признаков — наименьшем, но самом глубоком представлении исходного изображения. При обнаружении объектов карты признаков из промежуточных сверточных слоев также могут быть непосредственно полезны, поскольку они представляют исходное изображение в разных масштабах. Следовательно, фильтр фиксированного размера, работающий с разными картами признаков, сможет обнаруживать объекты разного размера.

Предварительные точки (priors) — предварительно вычисленные поля, определенные в определенных позициях на определенных картах объектов, с определенными пропорциями и масштабами. Они тщательно выбираются, чтобы соответствовать характеристикам ограничивающих рамок объектов в наборе данных.

Индекс Жаккара или Перекрытие Жаккара или Пересечение над объединением (IoU) измеряют степень или степень, в которой два блока перекрываются.

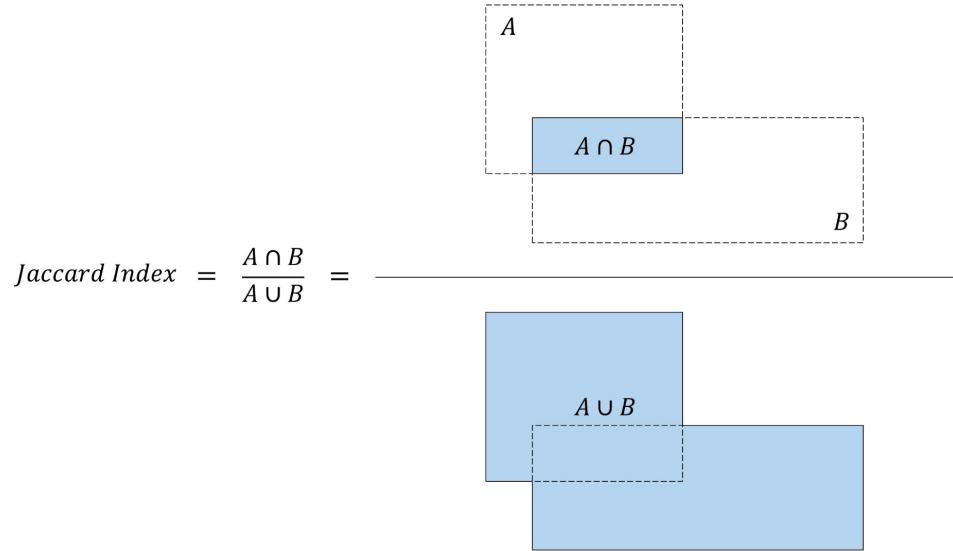


Рисунок 11. Индекс Жаккара

YOLO – это система обнаружения объектов. Разработана Джозефом Редмоном (Joseph Redmon). Преимущество YOLO над другими архитектурами — скорость. Модели семейства YOLO быстры и превосходят R-CNN (Region-Based Convolutional Neural Network) [33] и другие модели. Это позволяет добиться обнаружения объектов в режиме реального времени [20].

На момент первой публикации по сравнению с другими системами, такими как R-CNN и DPM (Deformable Part Model), YOLO добилась передового значения mAP (mean Average Precision). С другой стороны, YOLO испытывает трудности с точной локализацией объектов. Однако в новой версии были внесены улучшения в скорости и точности системы.

Другие архитектуры в основном использовали метод скользящего окна по всему изображению, и классификатор использовался для определенной области изображения (DPM). Также, R-CNN использовал метод предложения регионов (region proposal method). Описываемый метод сначала создает потенциальные bounding box'ы. Затем, на области, ограниченные bounding box'ами, запускается классификатор и следующее удаление повторяющихся распознаваний, и уточнение границ рамок.

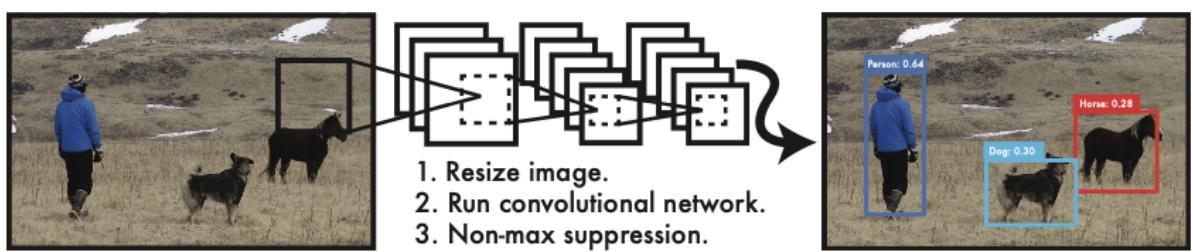


Рисунок 12. Система обнаружения YOLO

Обработка изображений с YOLO представляет собой последовательность шагов (Рисунок 12). Система:

- (1) изменяет размер входного изображения до 448×448 ,
- (2) запускает единственную сверточную сеть на изображении
- (3) ограничивает результирующие обнаружения достоверностью модели.

В любом заданном месте несколько априорных значений могут значительно перекрываться. Следовательно, предсказания, вытекающие из этих априорных значений, на самом деле могут быть дубликатами одного и того же объекта. Немаксимальное подавление (non-max suppression) — это средство удаления избыточных прогнозов путем подавления всех прогнозов, кроме прогноза с максимальной оценкой.

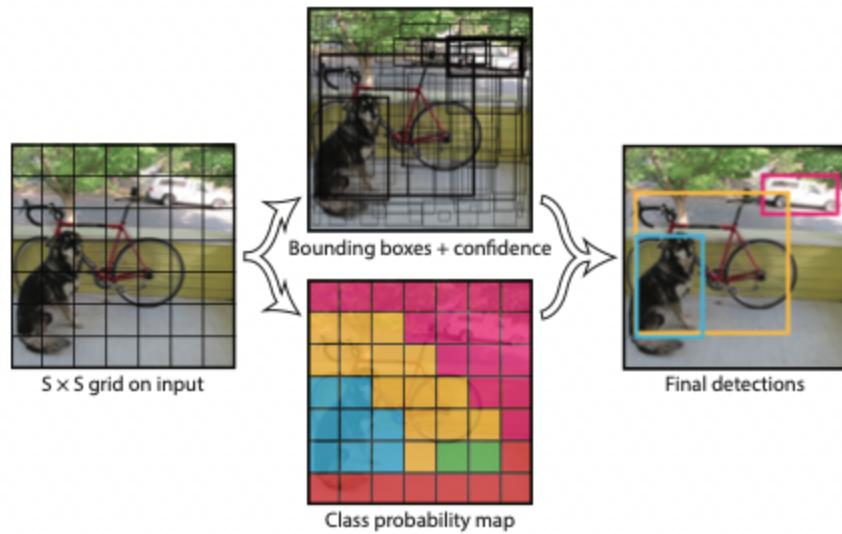


Рисунок 13. Модель YOLO

YOLO переосмыслила задачу обнаружения объектов в задачу регрессии. Она идет от пикселей изображения к координатам ограничивающих рамок и вероятностей классов. Тем самым, единая сверточная сеть предсказывает несколько ограничивающих рамок и вероятности классов для содержания этих областей (Рисунок 13).

Система YOLO моделирует обнаружение как проблему регрессии. Модель делит изображение на сетку $S \times S$ и для каждой ячейки сетки предсказывает B ограничивающих рамок, достоверность для этих рамок и вероятности класса C . Эти предсказания кодируются как тензор $S \times S \times (B * 5 + C)$.

2.2.5 Клиент-серверная архитектура

Клиент-серверная архитектура — вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг, называемыми серверами, и заказчиками услуг, называемыми клиентами. Фактически клиент и сервер — это программное обеспечение. Обычно эти программы расположены на разных вычислительных машинах и взаимодействуют между собой через вычислительную сеть посредством сетевых протоколов, но они могут быть расположены также и на одной машине. Программы-серверы ожидают от клиентских программ запросы и предоставляют им свои ресурсы в виде данных (например, передача файлов посредством HTTP, FTP, BitTorrent, потоковое мультимедиа или работа с базами данных) или в виде сервисных функций (например, работа с электронной почтой, общение посредством систем мгновенного обмена сообщениями или просмотр web-страниц во всемирной паутине). Поскольку одна программа-сервер может выполнять запросы от множества программ-клиентов, ее размещают на специально выделенной вычислительной машине, настроенной особым образом, как правило, совместно с другими программами-серверами, поэтому производительность этой машины должна быть высокой. Из-за особой роли такой машины в сети, специфики ее оборудования и программного обеспечения, её также называют сервером, а машины, выполняющие клиентские программы, соответственно, клиентами [34].

Преимущества:

- Отсутствие дублирования кода программы-сервера программами-клиентами.

- Так как все вычисления выполняются на сервере, то требования к компьютерам, на которых установлен клиент, снижаются.
- Все данные хранятся на сервере, который, как правило, защищён гораздо лучше большинства клиентов. На сервере проще организовать контроль полномочий, чтобы разрешать доступ к данным только клиентам с соответствующими правами доступа.

Недостатки:

- Неработоспособность сервера может сделать неработоспособной всю вычислительную сеть. Неработоспособным сервером следует считать сервер, производительности которого не хватает на обслуживание всех клиентов, а также сервер, находящийся на ремонте, профилактике и т. п.
- Поддержка работы данной системы требует отдельного специалиста — системного администратора.
- Высокая стоимость оборудования.

2.2 Инструменты реализации

Для реализации системы используются языки программирования высокого уровня Python версии 3.9 и JavaScript стандарта ECMAScript 2021 года.

Python — это высокоуровневый интерпретируемый язык программирования общего назначения. Философия его дизайна делает упор на удобочитаемость кода с использованием обязательных отступов. Python имеет динамическую типизацию и сборку мусора. Он поддерживает несколько парадигм программирования, включая структурированное (особенно процедурное), объектно-ориентированное и функциональное программирование [8].

Клиентская часть приложения использует язык разметки HTML, язык стилей CSS и стандартную библиотеку языка JavaScript. JavaScript — основной язык программирования, который используется для построения интерфейсов web-приложений. В качестве фреймворка клиентской части приложения используется React от компании Facebook.

Серверная часть приложения использует Python библиотеку FastAPI версии 0.70. FastAPI — это современная, быстрая и высокопроизводительная веб-инфраструктура для создания API-интерфейсов на основе стандартной аннотации типов Python. Система создана на основе стандартов: OpenAPI и JSON Schema.

Для доставки рабочего приложения используется инструмент виртуализации Docker. Система Docker позволяет запускать программы в изолированной среде — контейнере, что предоставляет возможности для портативности приложения.

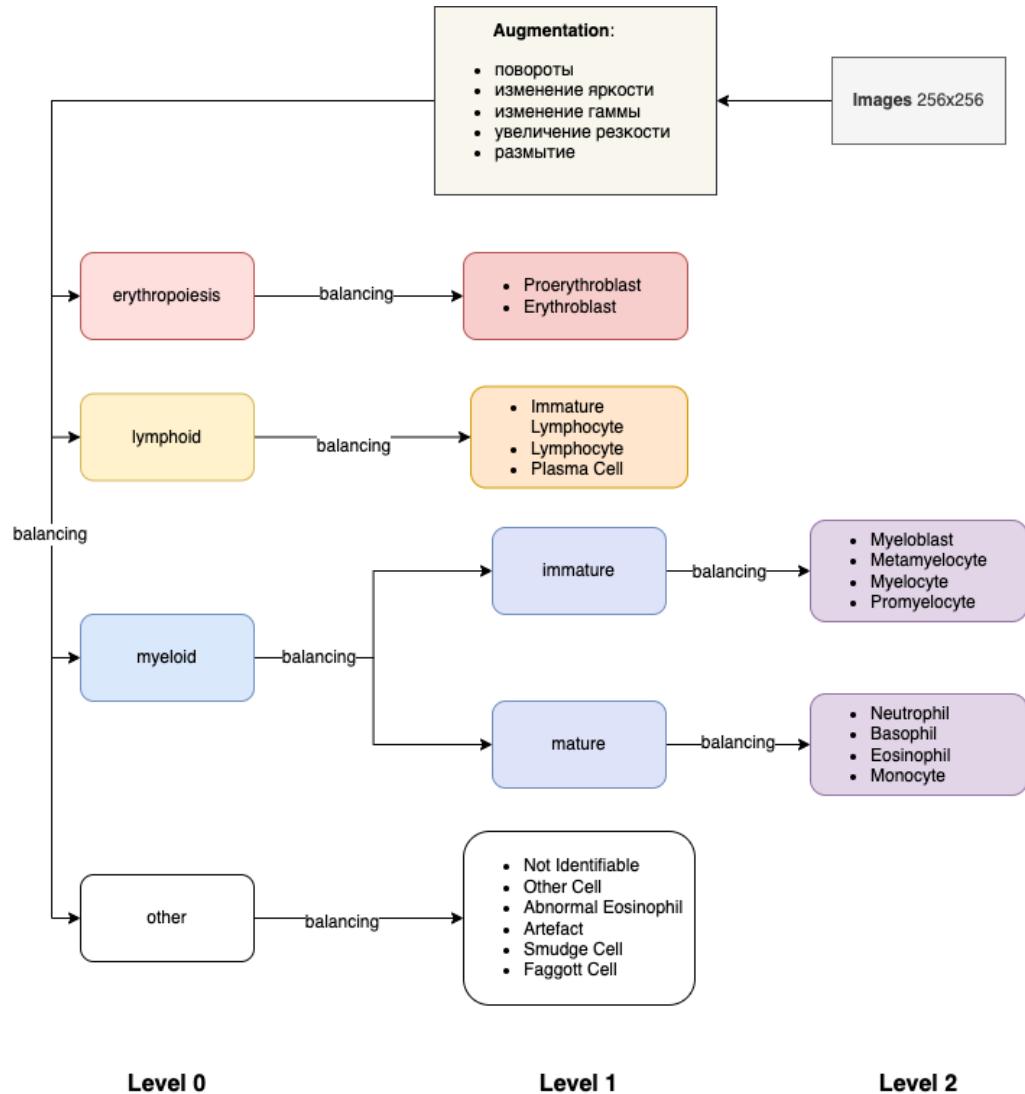
Обучение и анализ моделей проведен при помощи стандартной библиотеки глубокого обучения PyTorch версии 1.10 и расширения для работы со сверточными нейронными сетями torchvision версии 0.12.

Для методов интерпретационного машинного обучения используется библиотека Captum версии 0.5. Captum предоставляет современные алгоритмы, в том числе интегрированные градиенты, чтобы предоставить исследователям и разработчикам простой способ понять, какие функции влияют на выходные данные модели. Captum помогает исследователям машинного обучения легче внедрять алгоритмы интерпретируемости, которые могут взаимодействовать с моделями PyTorch.

3 ПРОЕКТНОЕ РЕШЕНИЕ

3.1 Концептуальная модель и алгоритмы

Логика классификации представляет собой набор шагов, представленный на Рисунке 14.



Level 0

Level 1

Level 2

Рисунок 14. Концептуальная модель тренировки модели

Перед началом тренировки модели происходит процесс аугментации, который состоит из случайных поворотов, изменений яркости, изменения

гаммы, увеличения резкости и размытия. Такой выбор формата аугментации обусловлен особенностями работы микроскопа [35].

Логика тренировочного процесса представляет собой древовидную структуру. Так как набор данных может быть несбалансированным, то для этого важно использовать балансировку классов, так как это напрямую влияет на качество классификации [36]. Для этого используется алгоритм, описанный на Рисунке 15. На вход подается графовая карта классов или типов клеток с учетом путей клеточной дифференцировки (Рисунок 16). Карта представляет собой вложенную древовидную структуру с названиями типов клеток на данном уровне. Алгоритм проходит по данной карте и на каждом уровне оценивает какое количество объектов находится на данном уровне дерева и производит балансировку на основе класса с минимальным количеством объектов, не меньше заданного порога, например, класса с количеством не меньше 1000 изображений. На каждом уровне происходит подготовка данных для обучения, валидации и тренировки с соотношением 8/1/1 соответственно.

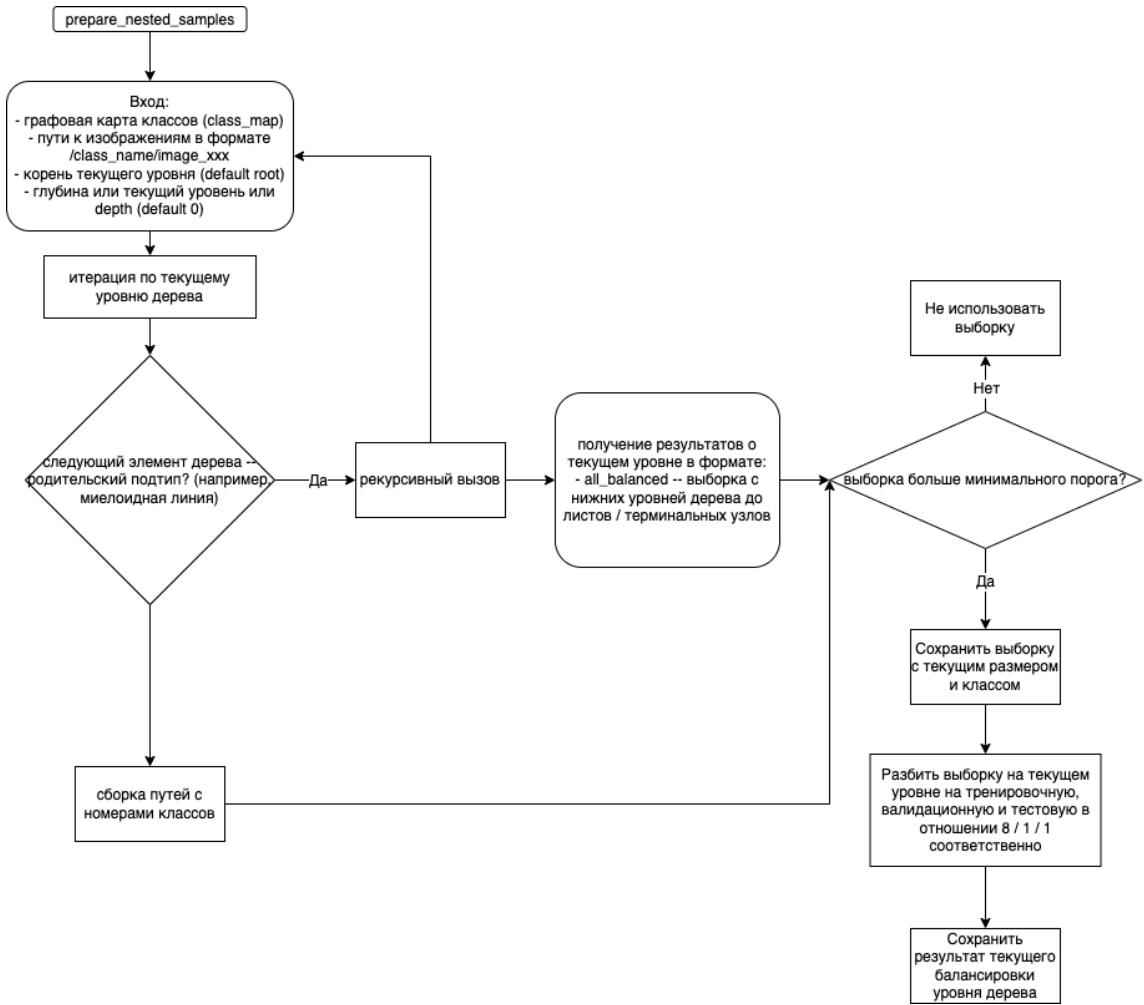


Рисунок 15. Алгоритм балансировки выборки на каждом уровне дерева
описания клеточной дифференцировки

```

CELL_TYPES = {
    'erythropoiesis': {
        'Proerythroblast': 'PEB',
        'Erythroblast': 'EBO',
    },
    'lymphoid': {
        'Immature Lymphocyte': 'LYI',
        'Lymphocyte': 'LYT',
        'Plasma Cell': 'PLM',
    },
    'myeloid': {
        'myeloid_immature': {
            'Myeloblast': 'BLA',
            'Metamyelocyte': 'MMZ',
            'Myelocyte': 'MYB',
            'Promyelocyte': 'PMO',
        },
        'myeloid_mature': {
            'Neutrophil': {
                'Band Neutrophil': 'NGB',
                'Segmented Neutrophil': 'NGS',
            },
            'Basophil': 'BAS',
            'Eosinophil': 'EOS',
            'Monocyte': 'MON',
        },
    },
    'abnormal': {
        'Not Identifiable': 'NIF',
        'Other Cell': 'OTH',
        'Abnormal Eosinophil': 'ABE',
        'Artefact': 'ART',
        'Smudge Cell': 'KSC',
        'Faggott Cell': 'FGC'
    }
}

```

Рисунок 16. Описание дерева типов и подтипов клеток крови на основе путей их дифференцировки

Далее осуществлялась классификация клеток на типы и подтипы. В последствии для каждого класса происходила тренировка нового классификатора на основе результатов предыдущего.

На этапе интерпретации используются уже натренированные модели. Для интерпретации необходимо выбрать одну из натренированных моделей, а также выбрать изображение, которое будет использоваться для интерпретации. После этого выбранная модель и изображения загружаются

на сервер и пользователю предлагается выбор метода интерпретации. После выбора происходит использование выбранного метода для интерпретации. На выходе пользователь получает обработанное изображение, которое показывает признаки, которые учитываются нейронной сетью при классификации (Рисунок 17).

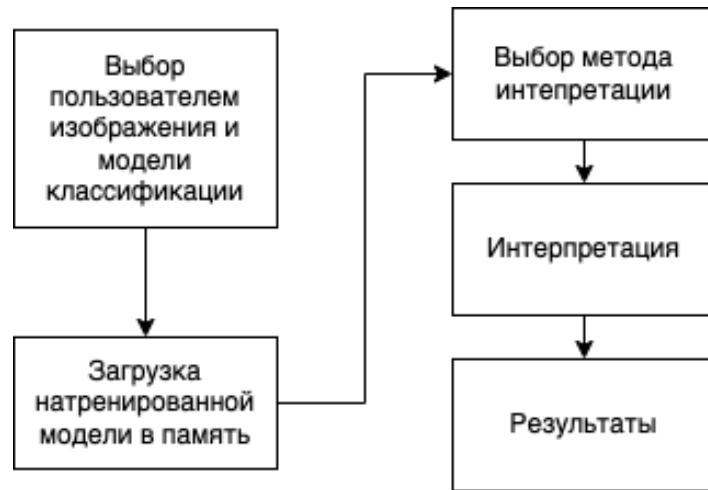


Рисунок 17. Концептуальная модель интерпретации работы модели классификации для выбранного изображения

3.2 Модель базы данных

База данных MySQL представляет собой множество элементов (таблиц), структурированное с целью одновременного исполнения запросов нескольких пользователей в конкретный момент времени. По сути, база данных MySQL представляет собой совокупность взаимосвязанных данных и содержит информацию о различных сущностях. На рисунке 18 приведена база данных MySQL, проектируемая для данной работы по классификации.

Сущность “Cells” необходима для хранения и просмотра информации о названиях типов клеток, их изображениях. Содержит поля: name (названия клеток), image (изображения клеток), type (тип клеток).

Сущность “ModelsTypes” необходима для хранения и просмотра информации о названиях натренированных моделей. Содержит поля: name (название модели).

Сущность “Models” необходима для хранения и просмотра информации о натренированных моделях. Содержит поля: name (название модели), dump (сохраненные веса модели), type (типы клеток для данной модели).

Сущность “CellTypes” необходима для хранения и просмотра информации о названиях типов клеток. Содержит поля: name (название типов клеток), code (сокращенное название типов клеток).

Сущность “ModelsToCellTypes” необходима для хранения и просмотра информации о связи названий типов клеток и соответствующих им моделях. Содержит поля: id_model (идентификатор модели), id_cell_type (идентификатор типов клеток).

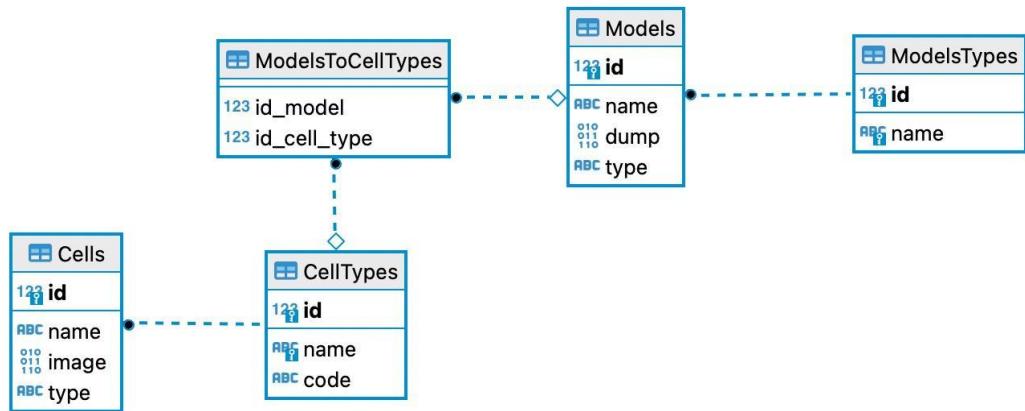


Рисунок 18. Схема данных

3.3 Архитектура системы

Система представляет собой клиент-серверное приложение, которое состоит из двух основных частей: сервера и клиента (Рисунок 19). Серверная часть реализована в виде программного обеспечения, которое позволяет обращаться на сервер по адресу. Обратиться можно на две входные точки: обучение моделей и тестирование моделей. При обучении моделей пользователю необходимо загрузить данные в специальном формате.

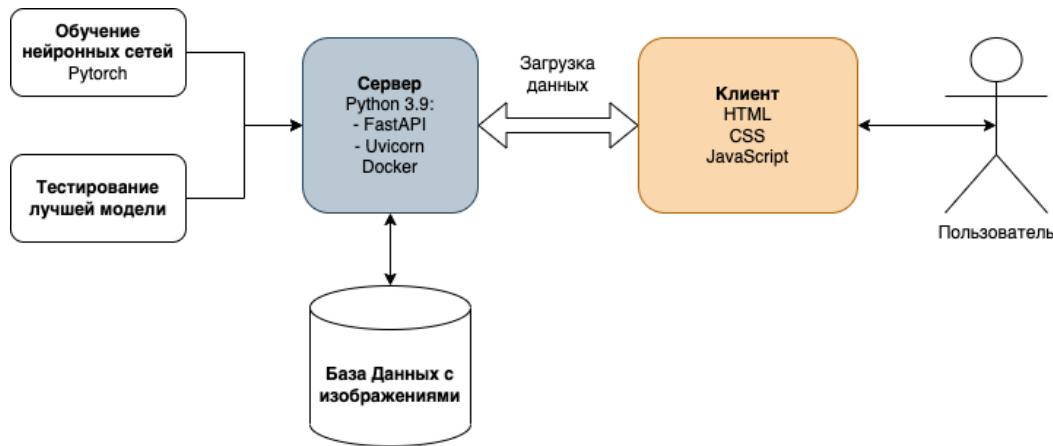


Рисунок 19. Архитектура системы

3.5 Взаимодействие с пользователем

Взаимодействие пользователя происходит через веб-браузер. Интерфейс реализован на основе HTML, CSS и JavaScript для взаимодействия с сервером.

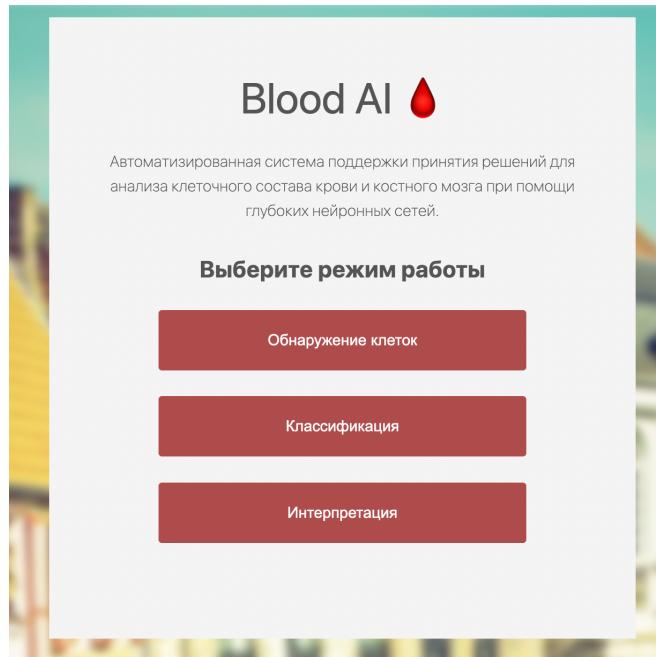


Рисунок 20. Стартовая страница программы

Перед использованием пользователю предлагается выбрать один из режимов работы системы: обнаружение клеток, классификация и интерпретация (Рисунок 20).

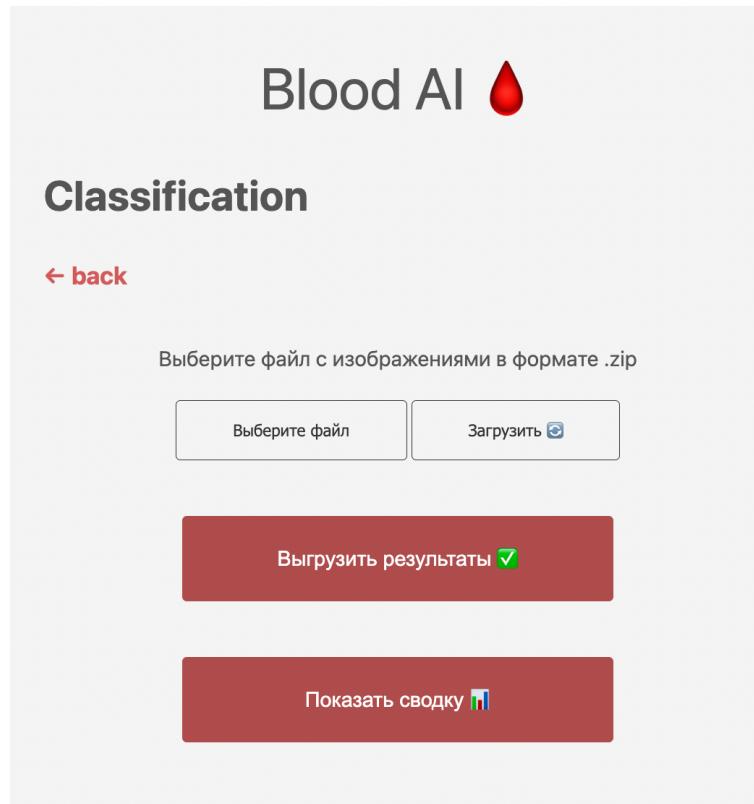


Рисунок 21. Интерфейс страницы для классификации

В режиме классификации (Classification) пользователю предлагается выбрать файл-архив с расширением .zip, в котором предполагается наличие изображений клеток крови и костного мозга (Рисунок 21).

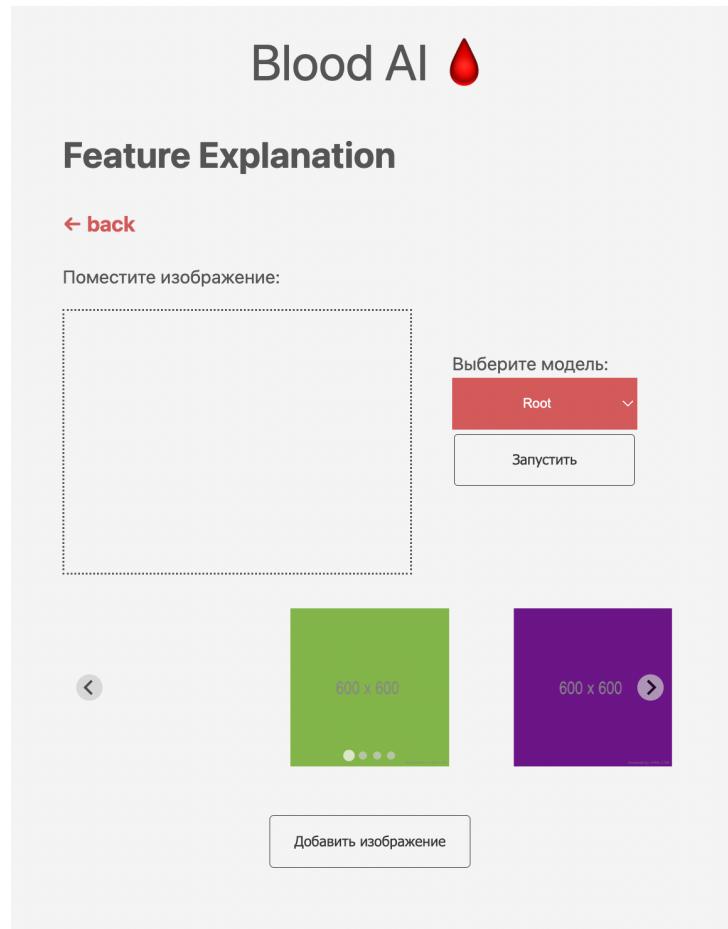


Рисунок 22. Интерфейс страницы для интерпретации признаков

Для работы методов интерпретационного машинного обучения (Feature Explanation) используется интерфейс на Рисунке 22. Для этого пользователю необходимо выбрать одну из обученных моделей на определенном уровне классификации на основе путей дифференцировки клеток крови и костного мозга, а также выбрать изображение, которого необходимо преобразовать при помощи методов интерпретационного обучения.

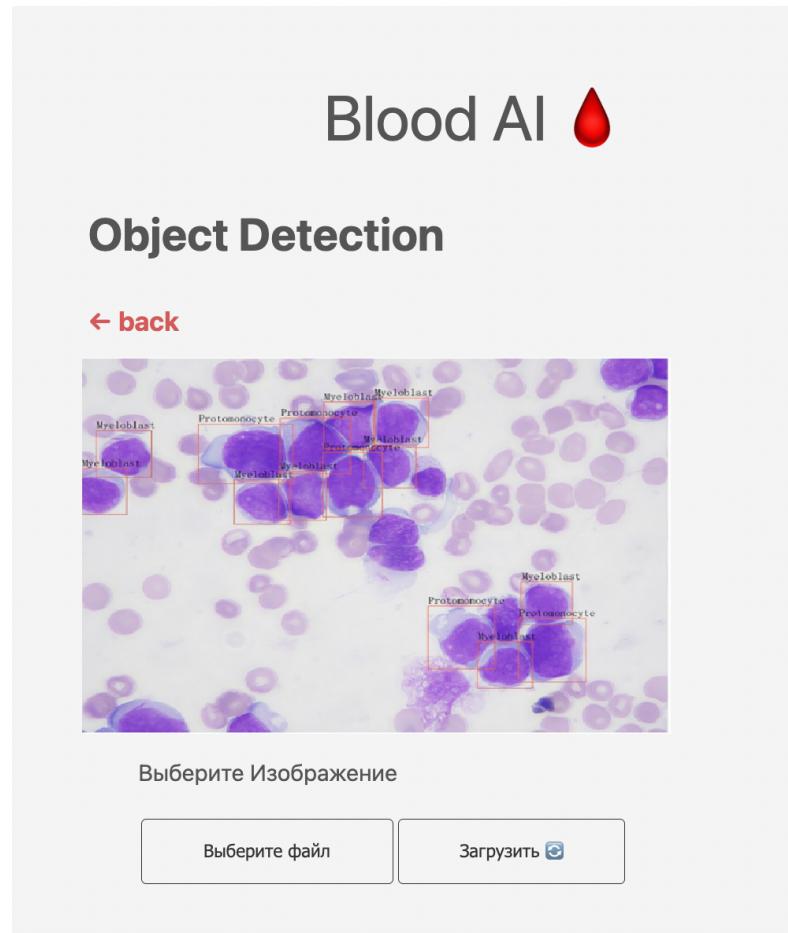


Рисунок 23. Интерфейс страницы для детекции клеток

На Рисунке 23 представлен интерфейс работы алгоритма обнаружения клеток на слайде с клетками костного мозга.

4 ЭКСПЕРИМЕНТ И РЕЗУЛЬТАТЫ

4.1 Подготовка данных

Изначальный набор данных представляет собой набор из двух источников. Каждый тип клеток имеет различное количество образцов, что означает несбалансированность выборок, поэтому было принято решения сбалансировать выборки на основе типа с наименьшим количеством образцов (Рисунок 24).

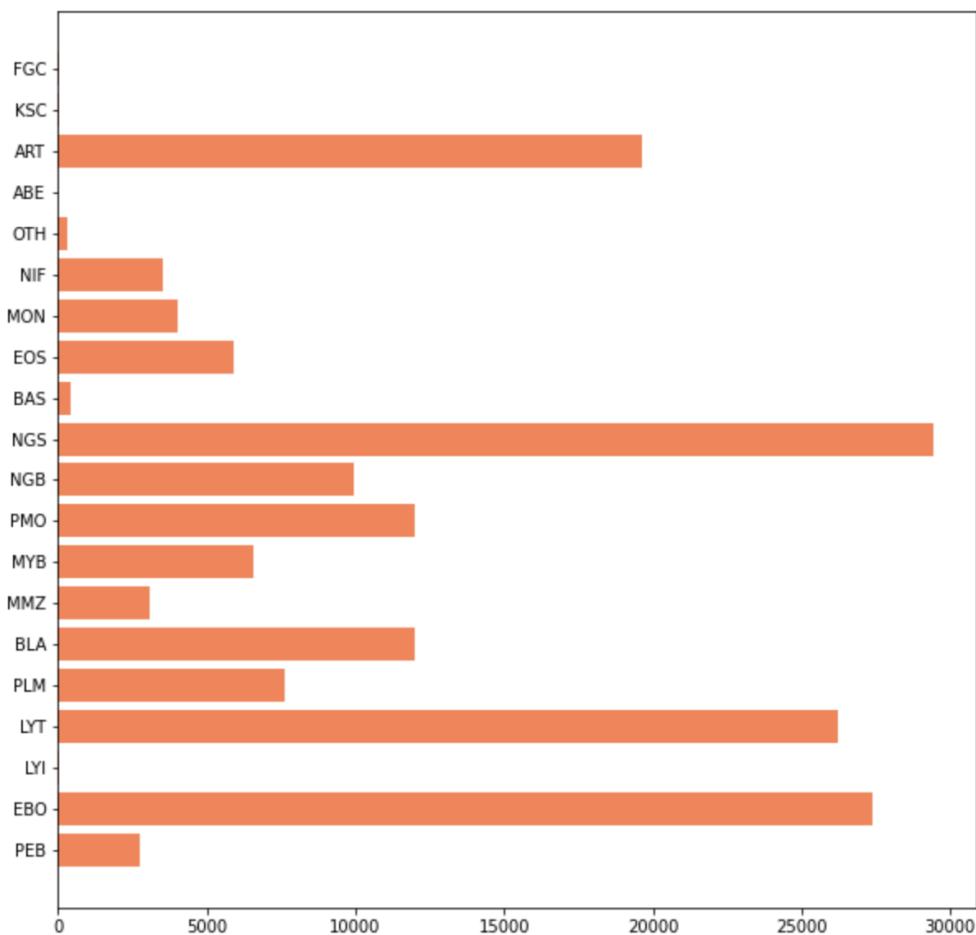


Рисунок 24. Несбалансированная выборка

```
[0, 1] erythropoiesis 2740
[0, 1] lymphoid 7629
[0, 1, 2, 3] myeloid_immature 3055
[0, 1] Neutrophil 9968
[0, 1, 2] myeloid_mature 4040
[0, 1] myeloid 12120
[0, 1] abnormal 3538
[0, 1, 2, 3] root 5480
```

Рисунок 25. Результаты балансировки на каждом уровне вложенности

На Рисунке 25 показан результат алгоритма балансировки на каждом уровне дерева классификации клеток.

Для расширения выборки использовалась аугментация со следующими параметрами (Рисунок 26):

- случайные повороты
- случайное изменение гаммы и контрастности
- случайное изменение размытости и резкости

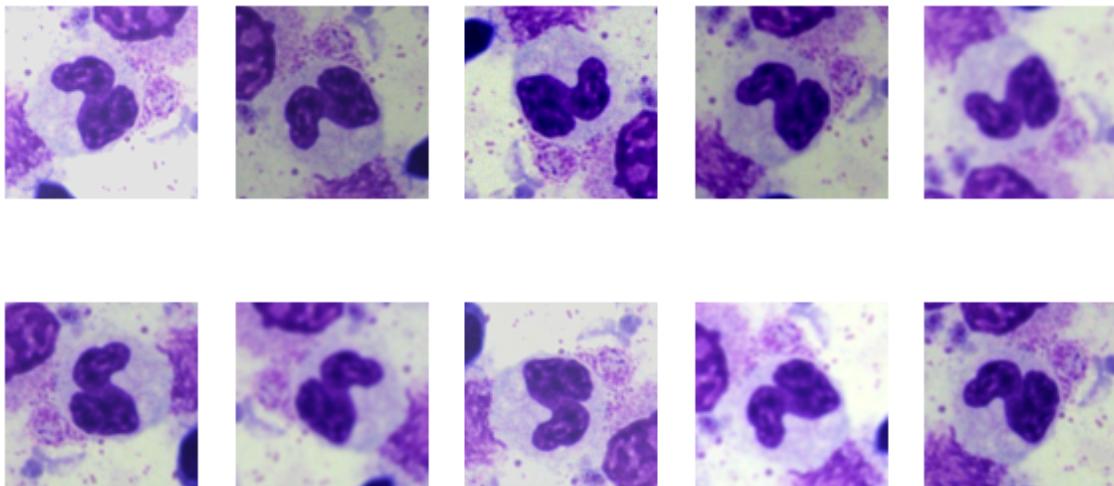


Рисунок 26. Аугментированные изображения

4.2 Результаты экспериментов

Для тренировки были выбраны следующие архитектуры сверточных нейронных сетей для классификации:

- ResNetXt50 (2019)
- MobileNetV3 (2019)
- EfficientNetV2 (2019)
- CoAtNet (2020)

Обучение моделей происходило в трех режимах: чистая архитектура без предобучения и аугментации, с предобучением, с предобучением и аугментацией с количеством эпох 10. Усредненные результаты по всем классам и подклассам изображены на Рисунке 27. Видно, что предобученные модели, которые обучались на аугментированных данных, показывают лучшие результаты в классификации. Это еще раз подтверждает, что для повышения точности необходимо использовать трансферное обучение и искусственно расширенную выборку.

чистая модель	с предобучением	с предобучением и аугментацией	Случай
81.7	87.5	89.5	корень дерева классификации
92.1	95.6	97.2	эритропоэз
95.9	97.2	97.9	лимфоидная линия
85.2	88.8	92.2	миелоидная линия
70.9	78.5	82.2	незрелая миелоидная линия
92.4	95.2	96.8	миелоидная зрелая линия
72.8	88.3	89.4	нейтрофилы
87.1	89.8	90.0	патологичные клетки
84.7	90.1	91.9	все

Рисунок 27. Усредненные результаты точности классификаторов на разных уровнях

На Рисунках 28, 29, 30 отображены графики процесса обучения классификаторов для различных классов для различных случаев в виде соотношения результатов функции потерь для тренировочной и валидационной выборок. Видно, что переобучение происходит на моделях без предобучения и аугментации, но в обратном случае результаты показывают корректный процесс обучения.

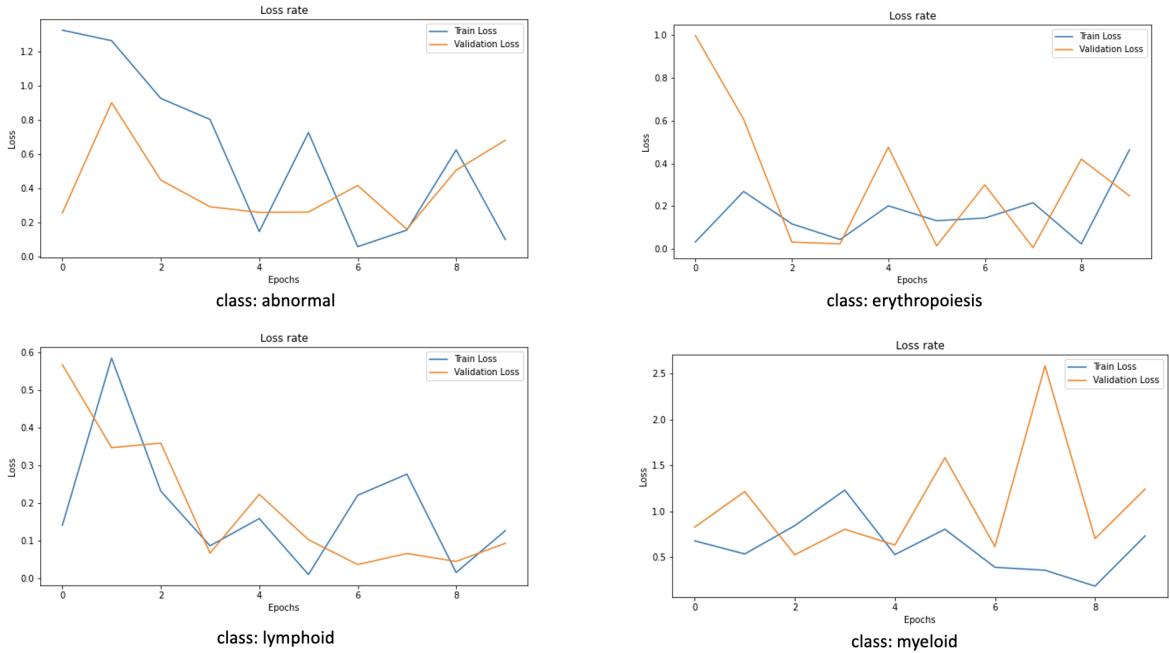


Рисунок 28. Процесс обучения: без предобучения и без аугментации

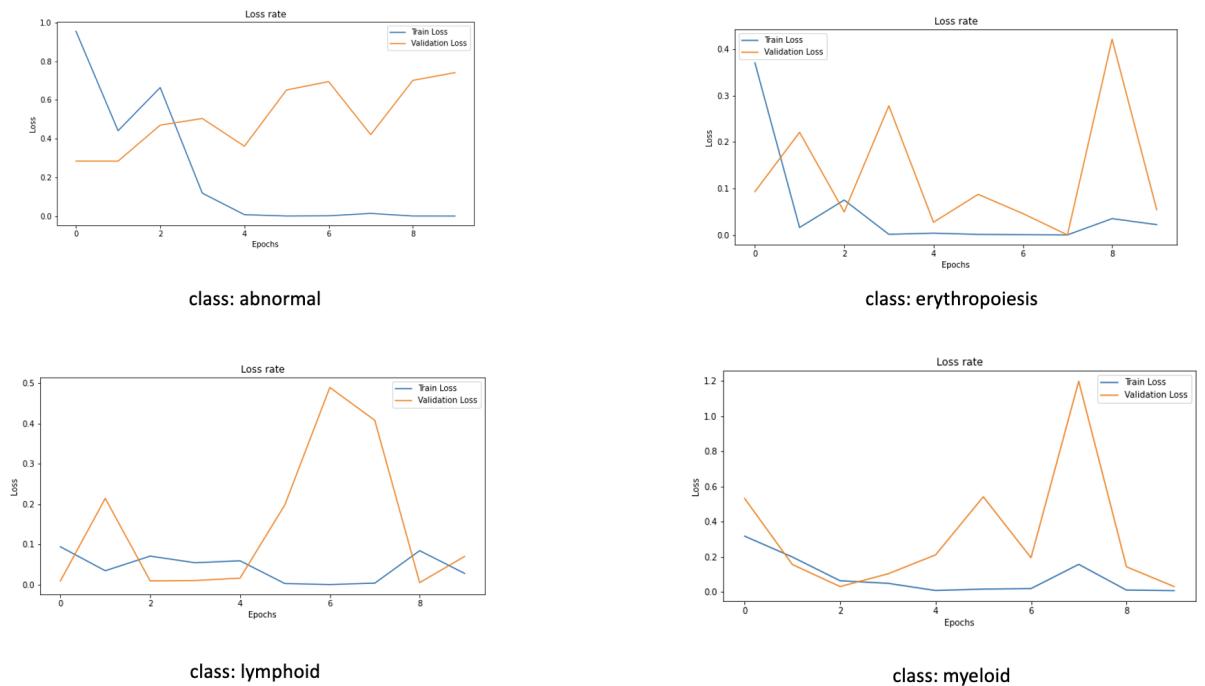


Рисунок 29. Процесс обучения: с предобучением и без аугментации

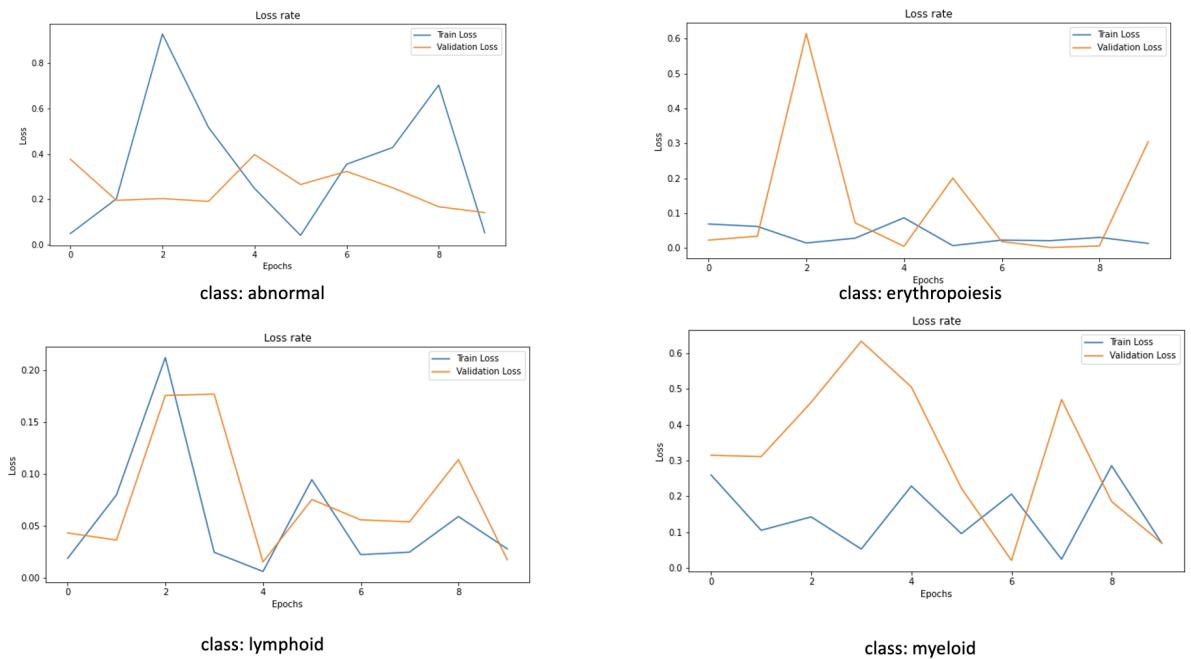


Рисунок 30. Процесс обучения: с предобучением и с аугментацией

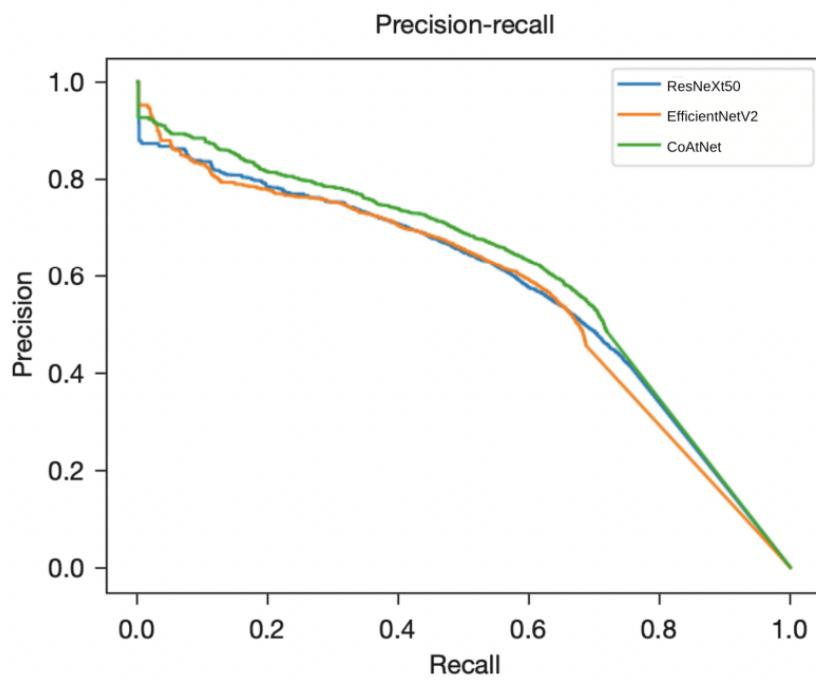


Рисунок 31. Отношение точности (precision) и чувствительности (recall)
для лучших моделей классификации

Рисунок 31 представляет собой график точности работы лучших моделей классификации. Важно отметить, что лучшие результаты показала модель CoAtNet, что также было подтверждено авторами данной архитектуры на конкурсе ImageNet.

На Рисунке представлены результаты работы двух методов интерпретационного машинного обучения: Метода Окклузий и Метода Интегрированных градиентов. На данных изображениях видно, что модель использует признаки, которые предоставляют ядра и форма клетки и ядра.

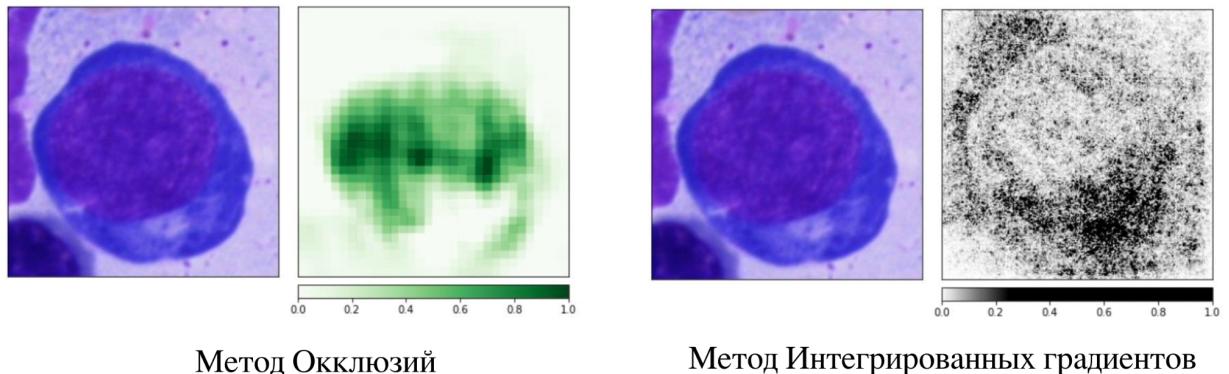


Рисунок 32. Результат работы методов интерпретационного обучения

На Рисунке 33 представлен результат работы алгоритма обнаружения на слайде изображения клеток крови и костного мозга. Можно заметить, что модель почти полностью определяет положение и тип клеток красного костного мозга.

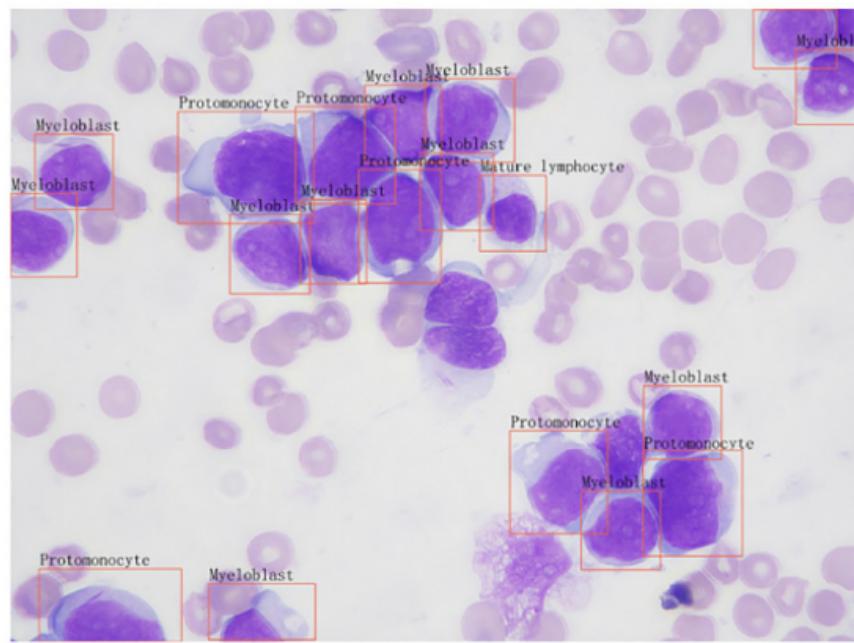


Рисунок 33. Результат работы алгоритма обнаружения клеток крови и костного мозга

ЗАКЛЮЧЕНИЕ

Основной результат данной работы — реализация алгоритмов машинного обучения и создание системы для обнаружения и классификации клеток крови и костного мозга, а также их подсчета выборке и на слайде изображения.

В данной работе были проанализированы и выбраны подходы к реализации методов классификации, интерпретации и обнаружения. Для классификации были выбраны архитектуры сверточных нейронных сетей: ResNetXt50, MobileNetV3, EfficientNetV2, CoAtNet. Для интерпретационного машинного обучения были выбраны: метод Окклузии и метод Интегрированных градиентов. Для обнаружения клеток на слайдах был выбран алгоритм YOLO, так как данная модель показала наилучшие результаты. В работе были реализованы выбранные алгоритмы и методы.

Для корректной работы с данными была создана база данных с изображениями клеток крови и костного мозга. Для работы с данными была разработана схема данных и выбрана СУБД MySQL для работы с со схемой и с изображениями.

В работе были проведены эксперименты по классификации клеток крови и клеток гемопоэтического ряда красного костного мозга путем анализа их гистологических снимков выбранными методами машинного обучения, а также провести эксперименты по выявлению признаков, которые используются моделью классификации. Кроме этого, были получены результаты по использованию модели YOLO в задаче обнаружения клеток. Точность классификации и обнаружения наилучшей выбранной модели составила 91% на 21 типе клеток крови и костного мозга. Модель была

выбрана путем сравнительного анализа различных использованных моделей в контексте аугментации и трансферного обучения.

Для работы с системой было реализовано клиент-серверное приложение и графический интерфейс пользователя. Пользователь может использовать данную систему при помощи веб-браузера.

Достоинства разработанного решения следующие:

1. универсальность классификаторов, то есть наличие возможности расширения количества классов и использование разработанной архитектуры для других типов клеток
2. масштабируемость системы и возможность запуска на любой платформе благодаря Docker
3. большое количество классов (21 класс) без потери точности (91%)
4. возможность отследить признаки, на которые “обращает внимание” нейронная сеть, за счет методов интерпретационного обучения

Недостатки и перспектива разработанного решения:

1. необходимость наличия производительного GPU для новых тренировок сверточных нейронных сетей
2. медленный вывод методов интерпретационного обучения (~2-3 минуты для одного изображения)
3. относительно низкая точность классификации. Сейчас это 91%, то есть примерно каждое 10-ое изображение может быть ошибочно классифицировано, что недопустимо в клинической практике. Данная проблема может быть решена за счет использования более нейронных сетей с большим количеством параметров, а также за счет оптимизации гиперпараметров нейронных сетей, однако все это требует наличия высокопроизводительного оборудования: GPU, CPU и RAM

СПИСОК ЛИТЕРАТУРЫ

- [1] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning second edition*. MIT Press, 2018.
- [2] Mehmed Kantardzic, *Data Mining. Concepts, Models, Methods, and Algorithms*. IEEE, 2020.
- [3] U. Kose, O. Deperlioglu, and D. Jude Hemanth, *Deep Learning for Biomedical Applications*. 2021. [Online]. Available: <https://www.routledge.com/>
- [4] Николенко Сергей, *Николенко Глубокое обучение*. 2018.
- [5] A. Glassner, *Deep Learning*. 2020.
- [6] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT Press, 2012.
- [7] J. Berner, P. Grohs, G. Kutyniok, and P. Petersen, “The Modern Mathematics of Deep Learning,” May 2021, [Online]. Available: <http://arxiv.org/abs/2105.04026>
- [8] N. K. Manaswi, *Deep Learning with Applications Using Python*. Apress, 2018. doi: 10.1007/978-1-4842-3516-4.
- [9] R. Szeliski, “Computer Vision: Algorithms and Applications,” 2010. [Online]. Available: <http://szeliski.org/Book/>.
- [10] C. Matek, S. Krappe, C. Münzenmayer, T. Haferlach, and C. Marr, “Highly accurate differentiation of bone marrow cell morphologies using deep neural networks on a large image data set,” *Blood*, vol. 138, no. 20, pp. 1917–1927, Nov. 2021, doi: 10.1182/blood.2020010568.
- [11] J. Liu *et al.*, “A deep learning method and device for bone marrow imaging cell detection,” *Annals of Translational Medicine*, vol. 10, no. 4, pp. 208–208, Feb. 2022, doi: 10.21037/atm-22-486.
- [12] V. Acharya and P. Kumar, “Identification and red blood cell automated counting from blood smear images using computer-aided system,” *Medical and Biological*

Engineering and Computing, vol. 56, no. 3, pp. 483–489, Mar. 2018, doi: 10.1007/s11517-017-1708-9.

- [13] E. J. Mascha and T. R. Vetter, “Significance, errors, power, and sample size: The blocking and tackling of statistics,” *Anesthesia and Analgesia*, vol. 126, no. 2, pp. 691–698, 2018, doi: 10.1213/ANE.0000000000002741.
- [14] D. J. Biau, S. Kernéis, and R. Porcher, “Statistics in brief: The importance of sample size in the planning and interpretation of medical research,” *Clinical Orthopaedics and Related Research*, vol. 466, no. 9. Springer New York, pp. 2282–2288, 2008. doi: 10.1007/s11999-008-0346-9.
- [15] A. Güneş, H. Kalkan, and E. Durmuş, “Optimizing the color-to-grayscale conversion for image classification,” *Signal, Image and Video Processing*, vol. 10, no. 5, pp. 853–860, Jul. 2016, doi: 10.1007/s11760-015-0828-7.
- [16] J. Rodellar, S. Alférez, A. Acevedo, A. Molina, and A. Merino, “Image processing and machine learning in the morphological analysis of blood cells,” *International Journal of Laboratory Hematology*, vol. 40, pp. 46–53, May 2018, doi: 10.1111/ijlh.12818.
- [17] R. B. Hegde, K. Prasad, H. Hebbar, and B. M. K. Singh, “Feature extraction using traditional image processing and convolutional neural network methods to classify white blood cells: a study,” *Australasian Physical and Engineering Sciences in Medicine*, vol. 42, no. 2, pp. 627–638, Jun. 2019, doi: 10.1007/s13246-019-00742-9.
- [18] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” p. 12, 2012, [Online]. Available: <http://code.google.com/p/cuda-convnet/>

- [19] M. M. Alam and M. T. Islam, “Machine learning approach of automatic identification and counting of blood cells,” *Healthcare Technology Letters*, vol. 6, no. 4, pp. 103–108, Jul. 2019, doi: 10.1049/htl.2018.5098.
- [20] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Dec. 2016, vol. 2016-December, pp. 779–788. doi: 10.1109/CVPR.2016.91.
- [21] C. M. Schürch, C. Caraccio, and M. A. Nolte, “Diversity, localization, and (patho)physiology of mature lymphocyte populations in the bone marrow,” *Blood*, vol. 137, no. 22, pp. 3015–3026, Jun. 2021, doi: 10.1182/blood.2020007592.
- [22] M. Abadi *et al.*, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems,” Mar. 2016.
- [23] F. Seide and A. Agarwal, “CNTK,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug. 2016, pp. 2135–2135. doi: 10.1145/2939672.2945397.
- [24] Y. Jia *et al.*, “Caffe: Convolutional Architecture for Fast Feature Embedding,” Jun. 2014.
- [25] The Theano Development Team *et al.*, “Theano: A Python framework for fast computation of mathematical expressions,” May 2016.
- [26] B. Farnham, S. Tokyo, B. Boston, F. Sebastopol, and T. Beijing, “Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems,” 2021.
- [27] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2009, pp. 248–255. doi: 10.1109/CVPR.2009.5206848.

- [28] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” Dec. 2015.
- [29] A. Howard *et al.*, “Searching for MobileNetV3,” May 2019.
- [30] M. Tan and Q. v. Le, “EfficientNetV2: Smaller Models and Faster Training,” Apr. 2021.
- [31] Z. Dai, H. Liu, Q. v. Le, and M. Tan, “CoAtNet: Marrying Convolution and Attention for All Data Sizes,” Jun. 2021.
- [32] C. Molnar, *Interpretable Machine Learning Interpretable Machine Learning A Guide for Making Black Box Models Explainable*. 2021. [Online]. Available: <https://christophm.github.io/>
- [33] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” Nov. 2013.
- [34] H. S. Oluwatosin, “Client-Server Model,” *IOSR Journal of Computer Engineering*, vol. 16, no. 1, pp. 57–71, 2014, doi: 10.9790/0661-16195771.
- [35] L. Perez and J. Wang, “The Effectiveness of Data Augmentation in Image Classification using Deep Learning,” Dec. 2017.
- [36] M. A. Lones, “How to avoid machine learning pitfalls: a guide for academic researchers,” Aug. 2021, [Online]. Available: <http://arxiv.org/abs/2108.02497>