Анализ

# Определение проблемы

Большая загруженность специалистов, просматривающих снимки пациентов, приводит к потере бдительности, вследствие чего результат может быть не точным.

# Выявление целевой аудитории

Услуги МРТ головного мозга в России предоставляются: государственными клиниками, больницами, поликлиниками, и коммерческими многопрофильными или специализированными клиниками. Частные учреждения открыты ко внедрению новых технологий и поэтому мы и выделяем их как целевую аудиторию.

# Анализ аналогов

На рынке России существует ближайший аналог: Цельс - сервис для анализа компьютерных томограмм на предмет наличия признаков кровоизлияний в мозг.

Наше решение позволит с большой точностью обнаружить опухоль и уведомить доктора.

К преимуществам нашего проекта можно отнести:

* простоту внедрения в процесс;
* скорость выдачи результата;
* перспективу внедрения обработки снимков и других мед. услуг.

Недостатками же являются:

* начальная узко-направленность проекта.

# Определение платформы и стека для продукта

Платформой для обучения нейронной сети стал Google Colab т.к. он проще для старта. Альтернативой является Yandex.Cloud. Google Collab – бесплатен и имеет более гибкий набор GPU.

Датасет был выбран на платформе Kaggle, т.к. его можно на прямую с сайта подключить к сервису Jupiter notebook Google Colab, он имеет размеченные снимки и является хорошим выбором для начального уровня в разработке нейронных сетей.

# Формулирование требований к MVP

MVP должен иметь вид Web-приложения принимающего на вход снимок МРТ. Далее снимок обрабатывается нейросетью, результат обработки отображается в Web-приложении.

Планирование

# Постановка задач

Первым шагом стал анализ всей информации и формулировка требований к MVP. Исходя из требований мы разделили процесс разработки на следующие этапы: проектирование, реализация и контроль.

После чего мы распределили роли и определились со сроками выполнения.

Проектирование позволило определить какие технологии и архитектуры будут использоваться в проекте, расширило сценарии использования системы, и начать работу над макетом Web-приложения.

Этап реализации включил в себя разработку нейронной сети и Web-приложения, их тестирование и внедрение.

Проектирование

# Архитектура системы (компоненты, модули системы)

## Back-end - программирование функционала Web-приложения

В качестве хостинга мы выбрали хостинг beget.

На сервере будет находиться модель и на него же будут загружаться изображения, для последующей обработки.

Далее перед нами стал выбор между связи между клиентом и сервером:

*Серверные приложения*

В этом случае HTTP-запросы отправляются напрямую на сервер приложения, а сервер отвечает HTML-страницей.

*Связь с использованием AJAX*

Другой тип архитектуры использует для связи AJAX (Asynchronous JavaScript and XML). Это означает, что JavaScript, загруженный в браузере, отправляет HTTP-запрос (XHR, XML HTTP Request) изнутри страницы и (так сложилось исторически) получает XML-, JSON-ответ.

*Клиентские (одностраничные) приложения*

AJAX позволяет вам загружать данные без обновления страницы.После сборки такие приложения отправляются в браузер, и любой последующий рендеринг выполняется на стороне клиента (в браузере).

*Универсальные/изоморфные приложения*

Некоторые библиотеки и фреймворки позволяют вам исполнять приложения как на сервере, так и в клиенте.

В этом случае для связи фронтенда с бэкэндом приложение использует и AJAX, и обрабатываемый на сервере HTML.

Задача клиента — уточнить, какие данные ему нужны (базы данных графов), или извлечь все различные фрагменты данных, которые ему нужны (REST API).

## Neuro

### О нейросетях

Искусственная нейронная сеть (ИНС) — вычислительная нелинейная модель, в основе которой лежит нейронная структура мозга.

### Виды нейронных сетей

Классификация по кол-ву слоев:

1. Однослойная структура нейронной сети. Представляет собой структуру взаимодействия нейронов, в которой сигналы со входного слоя сразу направляются на выходной слой, который, собственно говоря, не только преобразует сигнал, но и сразу же выдаёт ответ. Входные нейроны являются объединенными с основным слоем с помощью синапсов с разными весами, обеспечивающими качество связей.

2. Многослойная нейронная сеть со скрытыми промежуточными слоями. В процессе обработки данных каждый промежуточный слой — это промежуточный этап, на котором осуществляется обработка и распределение информации.

Классификация по направлению распределения информации по синапсам между нейронами:

1. Нейросети прямого распространения (однонаправленные). В этой структуре сигнал перемещается строго по направлению от входного слоя к выходному.

2. Рекуррентные нейронные сети (с обратными связями). Здесь сигнал двигается и в прямом, и в обратном направлении. В итоге результат выхода способен возвращаться на вход. Выход нейрона определяется весовыми характеристиками и входными сигналами, плюс дополняется предыдущими выходами, снова вернувшимися на вход.

3. Сверточные нейросети.

4. Радиально-базисные функции.

По методы обучения:

— с учителем;

— без учителя;

— с подкреплением.

### Архитектуры

1. Перцептрон

Модель перцептрона (Perceptron) также известна как однослойная нейронная сеть состоит всего из двух слоев:

* Входной слой (Input Layer)
* Выходной слой (Output Layer)

Перцептрон принимает входные данные и вычисляет веса каждого нейрона. После этого линейная комбинация весов передается функции активации (сигмоидальная функция) для классификации.

Перцептрон является первой моделью нейронных сетей. Добавление скрытых слоев сделает из модели многослойный перцептрон.

2. Сеть прямого распространения (FF)

Сеть прямого распространения (Feed Forward) — это искусственная нейронная сеть, в которой нейроны никогда не образуют цикла. В этой нейронной сети все нейроны расположены в слоях, где входной слой принимает исходные данные, а выходной слой генерирует результат в заданном виде. Помимо входного и выходного слоев, есть еще скрытые слои — это слои, которые не имеют связи с внешним миром. В нейронной сети прямого распространения каждый нейрон одного слоя связан с каждым нейроном на следующем слое. Слои с такими нейронами называются полносвязными (fully-conected, dense).

Увеличение числа скрытых слоев делает из нее глубокую нейронную сеть прямого распространения (Deep Feed-forward, DFF). Для обновления значений весов используется алгоритм обратного распространения ошибки (backpropagation).

Применение DFF и FF:

* Сжатие данных
* Распознавание образов
* Компьютерное зрение
* Распознавание речи

3. Рекуррентные нейронные сети (RNN)

В рекуррентных нейронных сетях (Recurrent Neural Network, RNN) каждый из нейронов в скрытых слоях получает на вход данные с определенной задержкой во времени. Также рекуррентная нейронная сеть обладает состоянием, приобретенное при обработки предыдущих элементов последовательности. RNN имеют внутренние циклы (петли), поэтому решение выносится при учете самих данных, а также текущего состояния сети.

Проблема этой нейронной сети — низкая скорость обучения. А также она не хранит давнюю информацию, т.е. не работает с учетом долгосрочной перспективы.

4. Долгая краткосрочная память (LSTM)

При большом объеме данных RNN становятся просто непригодными, поскольку запоминают скорее последнюю информацию и забывают о информации, полученной давным-давно. Эта проблема схожа с затухающими градиентами в сетях прямого распространения. Поэтому на замену обычным рекуррентным сетям приходит сеть с долгой краткосрочной памятью (Long Short-Term Memory, LSTM).

Нейронные сети LSTM обладают памятью, т.е. текущая информация сохраняется для последующего использования в будущем.

Применение LSTM:

* Распознавание речи
* Оптическое распознавание символов
* Генерация текстов
* Прогнозирование временных рядов
* Машинный перевод
* Вопросо-ответная система, чат-боты и NLP

5. Управляемы рекуррентные нейроны (GRU)

Управляемый рекуррентный блок (Gated Recurrent Unit, GRU) — это разновидность LSTM. Сеть GRU имеет два вентиля (сброса и обновлений), в то время как у LSTM их три (входной, выходной и вентиль забывания).

GRU использует меньше параметров обучения и, следовательно, использует меньше вычислительных ресурсов, выполняется и обучается быстрее, чем LSTM. Если входная последовательность большая или точность очень важна, то обычно используется LSTM, тогда как для меньшего потребления памяти и более быстрой работы используется GRU.

6. Сверточные нейронные сети (CNN)

Сверточные нейронные сети (Convolutional Neural Network, CNN) — это нейронные сети, которые показали высокую точность в классификации и кластеризации изображений, а также в распознавании объектов, хотя применяется практически везде. CNN состоят из двух видов слоев: слои свертки и пулинга. Слой пулинг необходим для уменьшения размерности.

Применение CNN:

* Распознавание образов
* Компьютерное зрение
* Видеоанализ
* NLP
* Распознавание рукописных символов

7. Сеть радиальных базисных функций (RBFN)

Сеть радиальных базисных функций (radial basis function network, RBFN) обычно используются для задач аппроксимации. Эта сеть обладает высокой скоростью обучения. Архитектура такая же как и у сети прямого распространения, но основное различие состоит в том, что RBFN использует радиально-базовую функцию в качестве функции активации.

RBFN определяет, насколько далеко сгенерированный результат радиально-базовой функции находится от целевого значения.

Применение RBFN:

* Аппроксимация функций
* Прогнозирование временных рядов
* Классификация
* В системах автоматического управления

Для нашей задачи подходит сверточная нейронная сеть.

### Принципы работы нейронной сети

Любая архитектура ИНС состоит из искусственных нейронов — элементов обработки, имеющих структуру 3 связанных друг с другом слоев: входным, состоящим из одного или более слоев скрытым и выходным.

Входной слой состоит из входных нейронов, которые передают информацию в скрытый слов. Скрытый слой в свою очередь передает информацию в выходной. Каждый нейрон имеет входы с весами — синапсами, функцию активации, определяющую выходную информацию при заданной входной, и один выход. Синапсы — регулируемые параметры, конвертирующие нейронную сеть в параметризованную систему.

Обучение нейронной сети в основном означает калибровку всех “весов” путем повторения двух ключевых шагов: прямого распространения и обратного распространения.

В прямом распространении мы применяем набор весов к входным данным и вычисляем выход. Нейронные сети повторяют как прямое, так и обратное распространение до тех пор, пока веса не будут откалиброваны для точного прогнозирования выхода.

Чтобы получить конечное значение, мы применяем функцию активации к суммам скрытых слоев. Целью функции активации является преобразование входного сигнала в выходной сигнал и необходимы для нейронных сетей для моделирования сложных нелинейных паттернов, которые могут пропустить более простые модели.

Функция активации

Взвешенная сумма со входов — активационный сигнал — проходит через функцию активации для вывода данных из нейрона. Есть несколько видов функции активации: линейная, ступенчатая, сигмоидная, тангенциальная, выпрямительная (Rectified linear unit, ReLu).

Функции активации вроде сигмоидной или тангенциальной являются нелинейными, но приводят к проблемам с затуханием или увеличением градиентов. Однако гораздо более простой вариант — использовать выпрямленную линейную функцию активации (rectified linear unit, ReLU),

Преимущества использования ReLU:

* ее производная равна либо единице, либо нулю, и поэтому не может произойти разрастания или затухания градиентов,
* вычисление сигмоиды и гиперболического тангенса требует выполнения ресурсоемких операций, в то время как ReLU может быть реализован с помощью простого порогового преобразования матрицы активаций в нуле;
* отсекает ненужные детали в канале при отрицательном выходе.

Из недостатков можно отметить, что ReLU не всегда достаточно надежна и в процессе обучения может выходить из строя («умирать»). Например, большой градиент, проходящий через ReLU, может привести к такому обновлению весов, что данный нейрон никогда больше не активируется. Если это произойдет, то, начиная с данного момента, градиент, проходящий через этот нейрон, всегда будет равен нулю. Соответственно, данный нейрон будет необратимо выведен из строя.

### Принцип работы операции свертки

Операцию свертки можно представить следующим алгоритмом:

1. Скользящее окно, называемое фильтром, с размером (n,n) двигается по входному признаку. Количество движений определяется заданным количеством фильтров.
2. Каждый полученный шаблон имеет форму (n,n,d), где d — глубина входного признака.
3. Каждый шаблон умножается на своё ядро свертки, в результате, формируется выходная карта признаков. Полученная выходная карта признаков имеет форму (h,w,N), где h и w — длина и ширина, полученные в результате отсечения, а N — количество фильтров.

Количество фильтров — гиперпараметр, поэтому выбирается самостоятельно. А ядра свёртки являются обучаемыми параметрами.

Принцип действия операция свёртки

1. На первом этапе сформируются шаблоны с обозначенным размером (х,х,х), где третье число — глубина изображения.
2. Полученные шаблоны умножаются на ядра свертки. Каждый преобразованный в результате умножения шаблон формирует вектор с длиной равной количеству фильтров.
3. Все преобразованные шаблоны объединяются в выходную карту признаков.

Уменьшения размера после операции свертки можно избежать путём эффекта дополнения (padding). Он заключается в добавлении строк и столбцов так, что центр скользящего окна можно поместить в каждую клетку.

Слой Pooling

В сверточных нейронных сетях применяется ещё один слой, называемый слоем Pooling. Суть этого слоя заключается в уменьшении размерности карты признаков.

Pooling имеет две разновидности: max-pooling и average-pooling. В большинстве случаев применяется max-pooling. Операция Pooling схожа с операцией свертки:

1. Скользящее окно, обычно это окно (2,2), двигается по карте признаков.
2. Из выбранного шаблона выбирается максимальное (max-pooling) или среднее (average-pooling) значение.
3. Формируется уменьшенная в размере карта признаков.

Причины уменьшать размерность с помощью Pooling:

* Для поддержания иерархичности.
* Уменьшение размерности приводит к уменьшению количества обучаемых коэффициентов, а следовательно и вычислительных ресурсов.

Свойства сверточных нейронных сетей

У CNN есть несколько свойств:

* Полученные представления являются инвариантными по отношению к переносу. На изображении объект может находиться в любом доступном месте, а сверточная сеть не запоминает его положения, CNN лишь знает о его представлениях.
* Модель CNN является пространственно-иерархической. На первых слоях изучаются локальные шаблоны, а последующие изучают шаблоны, полученные из первых слоев.

## Front-end - программирование внешнего отображения и интерактивных элементов Web-приложения.

Web-приложение будет разрабатывается своими силами на HTML и CSS.

# Разработка сценариев использования системы

В виде веб- или десктопного приложения, с размещением на мощностях медицинских учреждений.

# Разработка макета-Web-приложения

Разработка макета происходила в бесплатном онлайн-сервисе Figma, который представляет собой графический редактор для совместного проектирования Web-приложениев. Совместная работа над макетом, стала важным требованием при выборе графического редактора. К тому же мы уже использовали его для проектов предыдущего семестра и знакомы с интерфейсом.

Шапка — самая верхняя часть Web-приложения. На ней располагается логотип команды и меню. Шапка закреплена и перемещается вместе с перемещением пользователя по странице. Меню содержит ссылки для перехода к интересующим разделам Web-приложения.

В самом начале Web-приложения мы решили расположить краткую информацию о проекте и принципе работы. Знакомство пользователя с проектом позволит ему выстроить полную картину процесса обработки нейронной сетью снимка МРТ.

Раздел “Инструкция” содержит необходимую информацию по загрузке изображений и дальнейшие действия.

Раздел “Загрузка изображений”. Для загрузки изображения требуется авторизизация, кнопками “Войти” или “Зарегистрироваться”. Кнопки загрузить изображение и отправить позволяют осуществить процедуру передачи снимка на обработку нейронной сети.

Раздел “Результаты” предоставляет данные уже обработанных снимков. Информация выводится таблицей со следующими столбцами: время загрузки изображения, имя пациента(или ID), результат (есть опухоль или нет).

При возникновении в процессе работы с web-приложением неполадок или вопросов, раздел “Контакты” поможет связаться с нашей командой.

Рядом располагается информация об участниках проекта.

Наше Web-приложение содержит всю необходимую информацию расположенную в определенном порядке. Схожесть с конкурентами, заключается в порядке расположения информации.

Мы показывали наше Web-приложение знакомым с разных направлений ИРИТ-РТФ. Ими было замечено следующее: лаконичность и понятность интерфейса, не перегруженный дизайн, наличие лишних блоков.

Было решено использовать шрифт, который наиболее выделял бы текст и был доступен для чтения — шрифт Inter. Лаконичный и информативный дизайн Web-приложения позволяет пользователю быстро разобраться в интерфейсе и получить результаты. Интерфейс устроен так, что человек любого возраста сможет разобраться в нём. Чтобы не нагружать Web-приложение, было решение использовать минимальное количество картинок и компонентов.

Анализ конкурентов показал, что большинство, как и мы, стремится обеспечить доступность и понятность для пользователя и такой стиль похож на Web-приложения больниц и понятен людям.

Реализация

# Разработка нейронной сети

Использованные библиотеки:

keras - построение модели, обучение модеди, преобразование изображение;

Matplotlib - визуализация результатов;

Numpy - для работы с массивами.

Разработку нейронной сети мы начали с преобразований изображений в другой размер и нормализации. Разделения данных на тренировочные, валидационные и тренировочные. Также мы проделали аргументацию изображений, то есть мы изменили яркость изображений, контрастность и т.д. Делалась это с помощью встроенных функций keras.

Мы экспериментировали с количеством слоев и гиперпараметрами. Достигли 83 % на тестовых данных. Большего процента простым перебором мы достигнуть не смогли.

В дальнейшем была использована функция keras tuner для автоматического поиска гиперпараметров. После этого был достигнут результат 91 % на валидационных данных и 88 % на тестовых данных.

# Тестирование нейронной сети





# Обучение нейронной сети

Мы обучили сверточную нейросеть с учителем, подавая парами входные данные и ответ. Используя при этом оптимизационный алгоритм обучения Adam.

В ходе разработки нейронной сети мы разбили датасет на три группы тренировочную, валидационную и тестовую.

Мы изменили размер изображений датасета, а также перевели изображения в массивы, произвели нормализацию для более корректной работы нейронной сети.

Также мы проделали аргументацию изображений, то есть мы изменили яркость изображений, контрастность и т.д. Делалась это с помощью встроенных функций keras.

Мы решили использовать сверхточные нейронные сети, подавая на вход нейронной сети измененные нами изображения.

В ходе разработки мы проводили эксперименты с количеством слоев и гиперпараметров. С помощью простого перебора параметров мы достигли 83%. Далее мы использовали функционал keras для подбора гиперпараметров, что дало нам 91 % на валидационных данных и 88 % на тестовых данных.

Далее мы сохранили модель нейронной сети в файл с расширением "\*.h5"

# Разработка Web-приложения

Front-end

Нарисованный макет Web-приложения и окна загрузки изображения перенесён на Web-приложение. Верстка Web-приложения позволила расположить все элементы в соответствии с макетом.

Back-end

Web-приложение размещается на виртуальном хостинге beget.

В качестве веб-фреймворка мы взяли django, в силу его простоты и удобства работы с нейронными сетями.

Мы реализовали функционал регистрации и аутентификации пользователей, используя встроенный функционал Django.

Также была добавлена форма для загрузки изображения и дальнейшего преобразования для работы нейронной сети.

Файл с моделью нейронной сети был загружен в отдельную директорию из которой в процессе загрузки изображения Django будет загружать нейронную сеть и выдавать результат распознавания пользователю.

# Внедрение модели нейронной сети в Web-приложение

В ходе обучения нейронной сети на платформе google Collab, мы сохранили модель нейронной сети в виде файла с расширением "\*.h5". Далее модель была загружена в отдельную директорию серверной части Web-приложения. В момент отправки пользователем изображения, модель нейронной сети загружается, изображение подвергается ряду изменений, которые требуются для корректной работы нейронной сети, изменение размера, перевод фотографии в массив и т.д.

После чего модель нейронной сети делает распознавание на основе этого изображение и результат отправляется пользователю.

# Проверка работоспособности Web-приложения

Тестовые загрузки изображений с опухолями вывели положительный результат.

Мониторинг и управление

# Текущий контроль и корректировка задач

Важной частью любого проекта является контроль за ходом выполнения работ. Благодаря результатам такого контроля, наша команда смогла скорректировать поставленные задачи и направить процесс разработки в нужно русло.

Завершение проекта

К защите проекта нами был написан отчет о проделанной работе и составлены две презентации, одна из них была для предзащиты.

Завершая проект, мы провели встречу команды для оценки результатов и извлечения полезных уроков из успехов и неудач, полученных в ходе работы над проектом.