

ПРОЕКТ

Ищенко Георгия Дмитриевича

GalleryAI: Нейросетевой классификатор изображений с возможностью персонального обучения

Заказчик и руководитель

Ткаченко М. С.

Консультант

Завриев. Н. К.

Москва
2023

Оглавление

Введение	2
Постановка задачи	3
Актуальность	3
Целевая Аудитория	4
Обзор Аналогов	4
Решения	6
Технологии	6
Архитектура системы	6
Machine Learning модель	7
Back-end	7
Front-end	8
Android	9
Модель компьютерного зрения	11
Модель машинного обучения	11
Программная реализация	12
Результат	16
Пример работы программы	16
Точность ML модели	21
Заключение	26
Список Литературы	27

Введение

С каждым годом производители телефонов, компьютеров и другой микроэлектроники наращивают объем встроенной памяти. Связи с этим, большинство пользователей не задумываются о полезности той информации, которая хранится на их гаджетах. Одним из видов такой информации являются изображения, которых становится очень много по мере использования девайса. Поэтому, большое число пользователей часто встречаются с проблемой поиска и фильтрации нужных им фотографий среди других в директории.

На рынке существует достаточно много программ, которые позволяют сортировать фотографии по уже имеющимся критериям. Чаще всего нейронные сети в подобных программах определяют, что изображено на картинке и, основываясь на этом, фильтруют фотографии. Но иногда подобные критерии не являются тем, что нужно пользователю, так как тот хочет, чтобы его фотографии фильтровались по критериям, которые он придумал сам. Например, пользователь — флорист и сильно увлекается цветами, и на его телефоне сохранилось огромное число фотографий с цветами разных видов. У него появляется потребность из всего его огромного альбома с цветами выбрать только белые розы, исключая из списка тюльпаны, ромашки и хризантемы.

Таким образом, цель нашей платформы — помочь обычным пользователям в быстрой фильтрации фотографий в галерее по собственным критериям (далее мы их будем называть тегами), путем самостоятельного обучения пользователя нейронной сети и последующего добавления фотографий для фильтрации.

Постановка задачи

Предоставить пользователям следующие возможности:

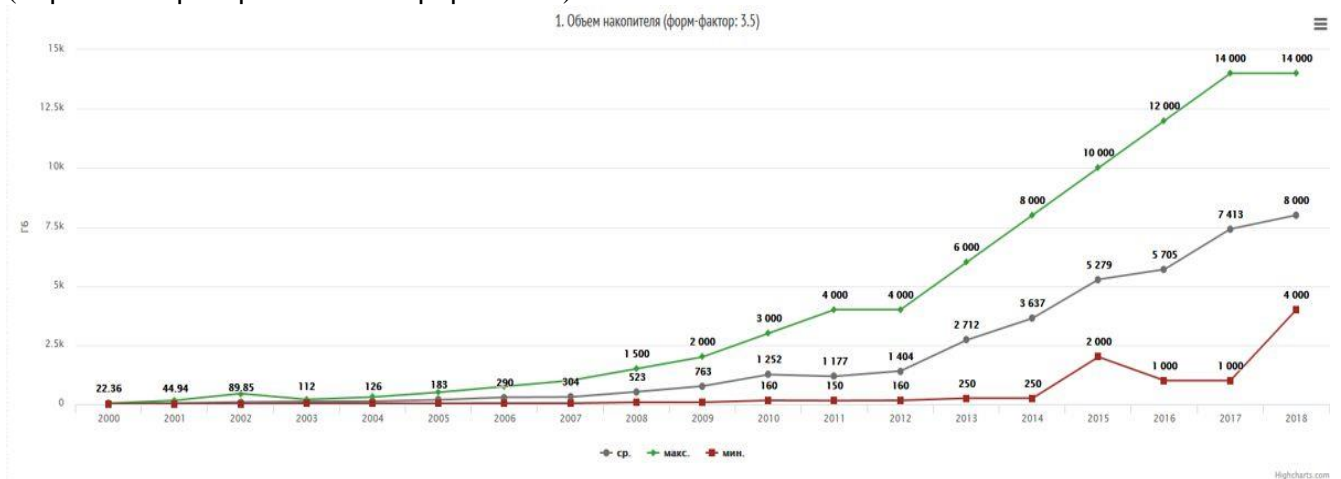
возможность сортировки большого количества фотографий

пользователь вручную размечает небольшое количество фотографии, обучая систему

система делает автоматическую классификацию новых фотографий

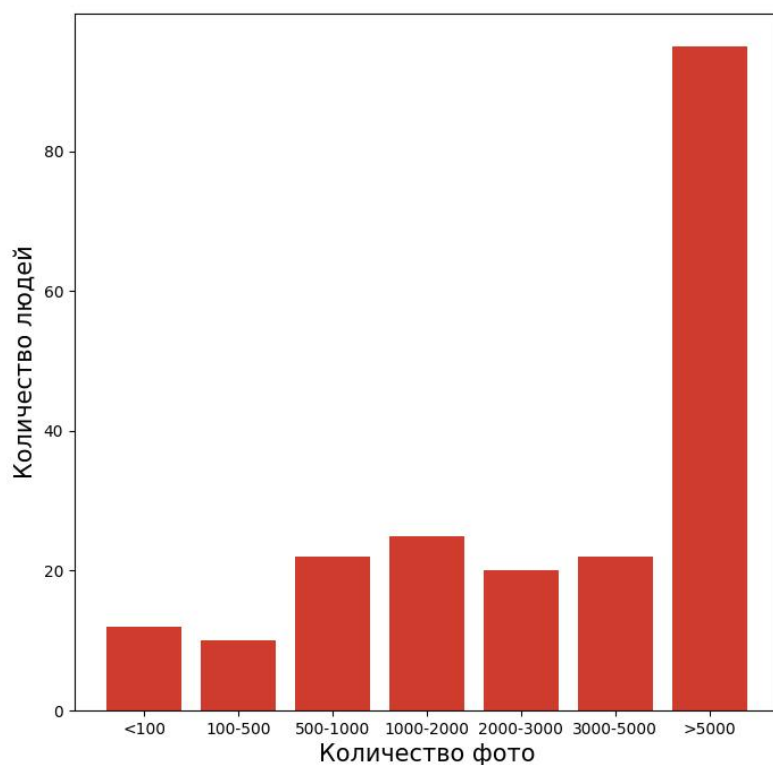
Актуальность

К сожалению, не удалось найти динамику роста количества хранимых людьми изображений. Однако удалось найти связанную с ней динамику роста объемов накопителей. (<https://www.pc4xp.ru/art/detail.php?ID=24>)



Из чего можно сделать вывод, что количество хранимых людьми фотографий лишь растет.

Среди 206 учеников лицея был проведен опрос, и была создана следующая статистика:



Целевая Аудитория

Система создана для людей, взаимодействующих с большим числом фотографий. GalleryAI подходит большинству людей, которые хранят на телефоне тысячу или даже более фотографий. В особенности людям, связавшим свою профессиональную деятельность с графикой (Фотографы, дизайнеры, художники и т. д.).

Обзор Аналогов

Существует множество решений с функционалом, схожим с функционалом нашего проектом, однако некоторые функции нашего проекта не дублируются ни в одном из изученных нами аналогов.

Отличные от данных аналоги не были представлены, т.к. их функционал направлен на другие действия и “пересекается” с функционалом нашего проекта в очень редких местах.

Говоря более конкретно, я выбирал такие аналоги, в функционал которых входит категоризации фотографий.

Оказалось, что возможность классификации фотографий по персонализированным категориям обладает только наш проект.

	GalleryAI	Google Photos	Gemini Photos	iOS internal
Обучение собственных моделей МЛ для автоматического теггирования	+	-	-	-
Android support	+	+	-	-
Web support	+	+	-	-
iOS support	-	+	+	+
Объединение фотографий по темам	+	+-	+	+

Gemini Photos - Приложение доступное только на операционной системе IOS. наиболее близкое по функционалу приложение. т. к. оно может объединять некоторые похожие фотографии в группы. Приложение умеет находить “неудачные” фото (размытые, абсолютные дубликаты существующих, не несущие никакой информации и т. д.)

iOS internal - Встроенный в операционную систему iOS аналог нашего приложения. Может сортировать фотографии по ограниченному списку тегов.

Google Photos - Решение с удобным сайтом, присутствуют приложения на iOS и Android. Присутствует возможность удалять дубликаты и находить “неудачные” фото. Возможность объединять фотографии по тегам почти полностью отсутствует.

Решения

Технологии

1. Используется технология сверточной нейронной сети (нейронная сеть, в которой есть хотя бы один сверточный слой)

При помощи технологий сверточной нейронной сети наиболее оптимально решать задачи классификации изображений. Человеку для классификации объекта необходимо выделить некоторые его базовые признаки: для определения кошки нужно осознать наличие у нее шерсти, лап, хвоста и так далее. Для определения самолета необходимо найти крылья. Подобная технология, однако теперь применяемая не человеком, а компьютером. Для начала выявляются характеристики базового уровня. Затем строится более абстрактная концепция, которую уже можно назвать признаком интересующего нас класса. По признаку класса гораздо легче строить предположение о принадлежности или непринадлежности изображения к интересующему нас классу.

2. Используется технология трансферного обучения нейронных сетей

Технология позволяет существенно сократить объем данных, необходимых для обучения. Подобное достигается за счёт того, что до момента начала обучения нейронная сеть уже умеет складывать базовые характеристики в признаки более высокого уровня. Заранее создается и обучается нейронная сеть. Эта сеть обучается таким образом, что после обучения она способна распознавать свойства многих данных на изображении. Эта модель берётся за основу и замораживается (перестает обучаться), т. к. в силу большого количества тренировочных данных этой сети она верно выполняет свою работу. Эта сеть называется базовой. Используется базовая сеть следующим образом: добавляем к выходному слою базовой нейронной сети ещё несколько слоёв и обучаем только их. Таким образом обучается лишь несколько слоёв, для чего вполне может хватить даже достаточно маленького количества тренировочных данных.

Архитектура системы

Проект состоит из пяти больших частей:

1. Machine Learning модель
2. Back-end
3. Front-end
4. Android приложение
5. Модель компьютерного зрения

Machine Learning модель

Функции модели:

1. Функция обучения
Модель получает отсортированные по категориям (тегам) изображения и учится выявлять характерные черты каждой категории.
2. Функция предсказания
Модель получает одну или несколько Фотографий и предсказывает принадлежность каждой фотографии к какой-либо категории
3. Функция сохранения
Модель сохраняется на сервере для использования в функции предсказания в дальнейшем
4. Функция сохранения и очищения фотографий, предназначенных для тренировки модели.
Для успешного работы функции обучения необходимо сохранять в специальное хранилище данных актуальные фотографии и удалять устаревшие

Back-end

Back-end представляет собой систему взаимодействия сервера, обладающего базой данных с пользовательской информацией, и Android приложения путем создания API.

Функции Back-end'a:

1. Полная система аутентификации пользователя, которая включает в себя:
 - a. Регистрацию
 - b. Авторизацию
 - c. Выход
 - d. Восстановление пароля
 - e. Изменение пароля
 - f. Отправка подтверждений всех действий, уведомлений и индивидуальных ссылок на подтверждение почты, восстановление и изменение пароля на email пользователя
2. Формирование базы данных, включающую в себя следующие модели с указанными полями:
 - a. Пользователь:
 - i. id

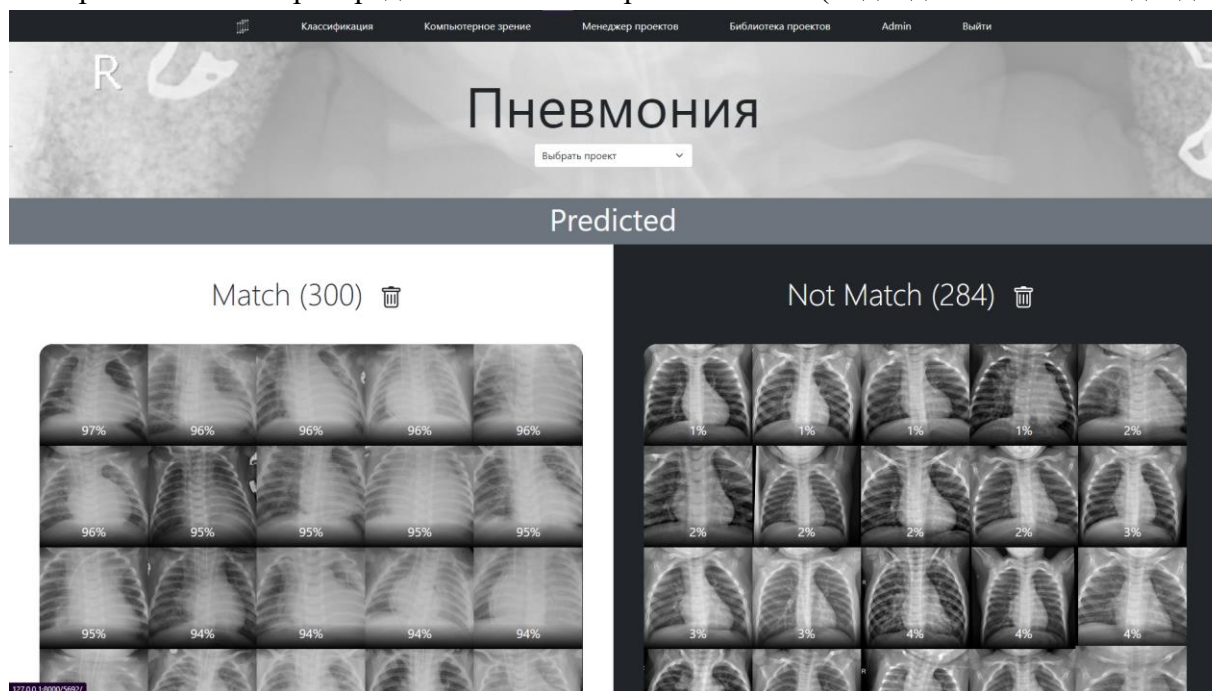
- ii. email
 - iii. имя пользователя
 - b. Теги пользователя:
 - i. id
 - ii. название тэга
 - iii. пользователь, которому принадлежит
 - c. Модель фотографии:
 - i. id
 - ii. изображение
 - iii. пользователь, которому принадлежит
 - iv. статус (подходит/не подходит)
 - v. тэг, которому принадлежит
 - vi. дата загрузки
 - vii. проставлен ли статус ИИ
3. API, обладающее следующим списком функций:
- a. Авторизация пользователя.
 - b. Выход пользователя из системы.
 - c. Получение списка фотографий пользователя с краткой информацией (id, ссылка на изображение, статус).
 - d. Получение подробного представления фотографии со всеми ее полями.
 - e. Удаление фотографии.
 - f. Получение списка тегов (название и список ссылок на картинки).

Front-end

Front-end представляет собой сайт, обладающий следующим функционалом:

1. Интерфейс для всех действий авторизации, описанных выше

2. Отображение распределения картинок (подходит/не подходит)

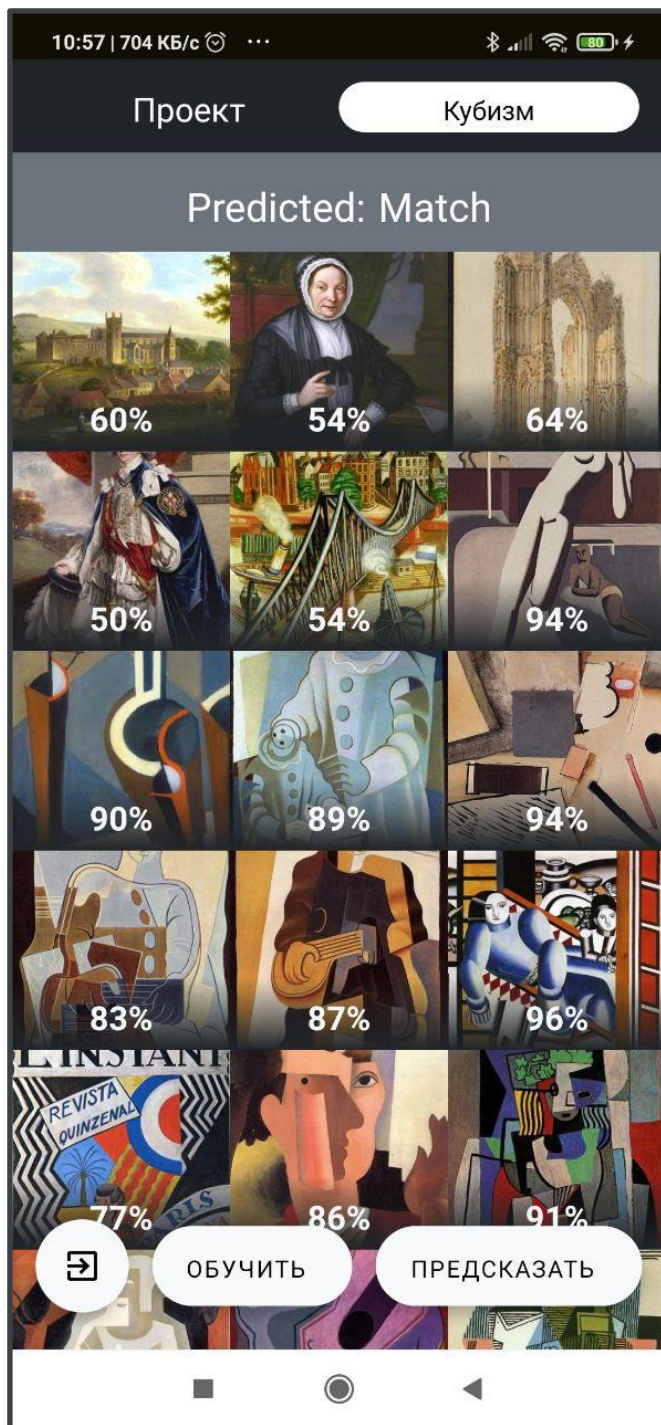


3. Просмотр, изменение статуса, удаление фотографии
4. Загрузка фотографий.

Android

Функционал Android-приложения:

1. Экран авторизации
 - a. Поле email'a
 - b. Поле пароля
 - c. Кнопка "Войти"
 - d. Отправка POST запроса с email'ом и паролем с целью получения токена
2. Главный экран
 - a. Отображение фотографий по статусам в виде списка из трех фотографий в ряду
 - b. Кнопка "Выйти из аккаунта"
 - c. Кнопка "Снять фото"
 - d. Кнопка "Сформировать датасет"
 - e. GET запрос для получения тэгов



3. Экран просмотра фотографии:

- Просмотр фото
- Удалить фото
- Поделиться фото
- GET запрос по определенной фотографии

4. Экран формирования датасета для тэга (пользователю поочередно предлагается выбрать не менее 10 фотографий разных статусов для обучения нейронной сети):

- a. Сообщение с просьбой выбрать фотографии, которые подходят/не подходят
- b. Стандартный Android-интерфейс с множественным выбором фотографий

Модель компьютерного зрения

В качестве базовой модели для машинного обучения была выбрана YOLO V3.

Функции модели:

1. Нахождение объектов на фотографии перед сохранением объекта фотографии.
2. Сохранение координат и класса найденного объекта в отдельную, связанную с объектом фотографии таблицу

Модель машинного обучения

Базовая Модель

Модель создана на базе модели MobileNet V2. MobileNet V2 обучена на наборе данных, состоящем из 1,4 млн изображений и 1000 классов ImageNet. Подробнее: <https://www.image-net.org/>

Модель и Процесс обучения

Базовая Модель используется как экстрактор признаков. Ниже изложена последовательность Создания модели. Тензором в данном контексте называется многомерный массив. Обозначение “тензор (a, b, c, d, e...)” означает многомерный массив размером a, b, c, d, e...

Модель:

1. Входной параметр Базовой модели - RGB изображение 160 x 160 пикселей (преобразованное в тензор (160, 160, 3)). Последний слой базовой модели - тензор (5, 5, 1280).
2. Тензор (5, 5, 1280) преобразуется в одномерный тензор (1280), также называемый вектор. Это происходит при помощи слоя GlobalAveragePooling2D. Подробнее: https://www.tensorflow.org/api_docs/python/tf/keras/layers/GlobalAveragePooling2D
3. Для преобразования выходного значения тензора (1280) в единое предсказание, представляющее из себя тензор (1), используется слой Dense. Подробнее: https://www.tensorflow.org/api_docs/python/tf/keras/layers/Dense
4. Базовая модель и два верхних слоя соединяются в единую модель.

Процесс Обучения:

1. Базовая модель “Замораживается” т. е. предотвращается обновление весов базовой модели в процессе обучения

2. Модель обучается в течение 10 эпох. Для этого используется функция библиотеки tensorflow fit. Подробнее: https://www.tensorflow.org/api_docs/python/tf/keras/Model#fit
3. Базовая модель “Размораживается” т. е. все слои модели могут обновляться в процессе обучения
4. Все слои базовой модели, кроме верхних 100 вновь “Замораживаются”
5. Модель обучается в течение 10 эпох. (Суммарно получается 20 эпох)

Используемая функция потерь

Используется функция “Binary Cross Entropy”. Считается эта функция следующим образом:

$$logloss = -\frac{1}{N} \sum_{i=1}^N \sum_{j=1}^L y_{ij} \log(p_{ij})$$

N - число элементов в выборке. L - число классов. p_{ij} - вероятность принадлежности i-го объекта к классу номер j. Если i-ый объект принадлежит классу номер j, $y_{ij} = 1$. В любом ином случае $y_{ij} = 0$.

Разбиение на данные для обучения и тестирования

В рабочей версии проекта все категоризированные пользователем фотографии используются для обучения модели. Для измерения точности модели были найдены различные наборы фотографий (Подробнее в разделе “Точность и Результаты”) Для обучения модели используются 20 фотографий каждой из 2 категорий фотографий (целевая категория и категория, состоящая из случайных фотографий) и 100 фото каждой из 2 категорий для оценки точности работы модели.

Программная реализация

1. Для написания проекта были выбраны языки

- a. Python3.9 (Back-end, Front-end, Нейронная сеть)

Язык Python был выбран благодаря простому синтаксису и огромному количеству библиотек, написанных под него. По Python существует наибольшее количество учебников и иных методических материалов. При возникновении трудностей в процессе разработке на python с большим шансом может получиться найти ответ на вопрос в интернете (из-за огромного сообщества программистов на python).

- b. Java (Android приложение)

Язык, на котором написано большинство Android приложений, поэтому существует огромное количество материала для изучения, который может быть использоваться для достижения поставленной задачи

2. Для Написания кода были выбраны среды разработки

a. Pycharm (Для написания конкурса на Python)

одна из самых популярных сред разработки для Python, обладающая удобными инструментами для работы с библиотеками, системами контроля версий, и эффективного написания кода

b. Android Studio (для реализации Android приложения)

одна из первых и самых популярных сред для разработки Android приложений. Имеет огромное количество удобных подручных средств, чтобы быстро разобраться в написании разметки и кода, даже для новичка.

3. Программа работает на redis-server при помощи библиотеки rq workers, позволяющей добавлять в очередь задания для выполнения и принимать задания из этой очереди

4. Используются библиотеки:

a. Python:

i. Tensorflow (Обучение и предсказания нейронной сети)

ii. Numpy (Для удобной работы с массивами и библиотекой TensorFlow)

В связке библиотек Tensorflow + Numpy удобно работать с различными данными (Например, легко представить фотографии в виде многомерных массивов). В библиотеке Tensorflow есть функции для работы с фотографиями, для их перевода в многомерный массив, для объединения их в массив большей мерности и создания набора данных. Для этой библиотеки написан учебник, подробно описывающий все классы, методы классов, наборы аргументов этих методов.

iii. OpenCV – Для поиска объектов на изображении в связке с YOLO V3

iv. Rabbit-MQ сервер (для работы Celery. через нее параметры для Train и Prediction передаются в модель и эти процессы запускаются)

v. Celery

Celery — один из способов организовать синхронное выполнение нескольких процессов. Например одновременное “обучение” нескольких моделей разных пользователей. При использовании RQ: Workers легко отслеживать исполняемые процессы, назначать число потоков (число работающих workers), уничтожать конфликтующие процессы.

vi. Urllib (Для скачивания фотографий)

Использовался в старой версии проекта в устаревшем механизме обмена данными между частями проекта

vii. Matplotlib (Для построения графиков)

Наиболее популярная и легкая в освоении библиотека для визуализации данных. С помощью этой библиотеки были построены графики зависимости Precision и Recall от порогового значения, поверхность F1-меры, статистика количества фото на телефоне среди учеников лицея, градиент F1-меры.

viii. Json (Для обмена сложных структур данных между задачами)

json отлично сериализуется и десериализуется в строки, поэтому он отлично подошел для обмена сложными структурами данных между задачами бэкенда и ml-модели

ix. Django-bootstrap4 (Для удобной django-шаблонизации в объектах bootstrap)

эта библиотека позволяет сократить практически до нуля написания html-разметок для объектов (например, формы), которые имеют представление в django-коде и сократить время написания фронтенда.

b. Android:

i. Picasso (загрузка изображений с сайта по ссылке)

чтобы отображать фотографии на телефоне, которые были загружены с сайта и не присутствуют во внутренней памяти телефона, нужно подгружать картинки с сервера. Android лишен большого выбора в библиотеках, решающих эту задачу. Поэтому Picasso был выбрана как наиболее популярное и стабильное решение.

ii. OkHTTP (HTTP запросы)

для обращения к API сайта и получения необходимых данных нужно осуществлять GET и POST HTTP запросы. OkHTTP наиболее современная библиотека, позволяющая асинхронно отправлять запросы на сайт.

iii. Gson (сериализация и десериализация данных JSON)

API сайта возвращает строку, которую можно превратить в JSON массив, в котором содержится объекты фотографий. Для того, чтобы превратить его в объекты Java-классов нужно извлечь значения полей из JSON массива. Gson позволяет это сделать быстро и без лишнего кода.

iv. Material design

В современной Android-разработке появляется все больший тренд на Material design, включающий в себя множество красивых элементов. Для создания приятного для глаза интерфейса был выбраны встроенные в Android Studio элементы Material design'a

5. Backend-фреймворки

- a. Django (Самый популярный фреймворк для быстрого и эффективного написания сайта)

Сегодня существует огромное количество современных инструментов, позволяющих создать сайт за короткое время. С помощью Django можно быстро организовать модели базы данных и проработать логику сайта. Кроме того, в Django есть встроенная админка, позволяющая отлаживать работу сайта.

- b. Django rest framework (Сопутствующая библиотека для написания API)

Для обмена данными между django-сервером и мобильным приложением нужно написать API. DRF позволяет за несколько строчек сериализовать данные Django-модели в JSON объект.

6. Frontend

- a. HTML

Стандартный язык разметки

- b. CSS

Стандартный язык стилей

- c. Bootstrap (Для красивого и быстрого оформления сайта без глубоких знаний в js и css)

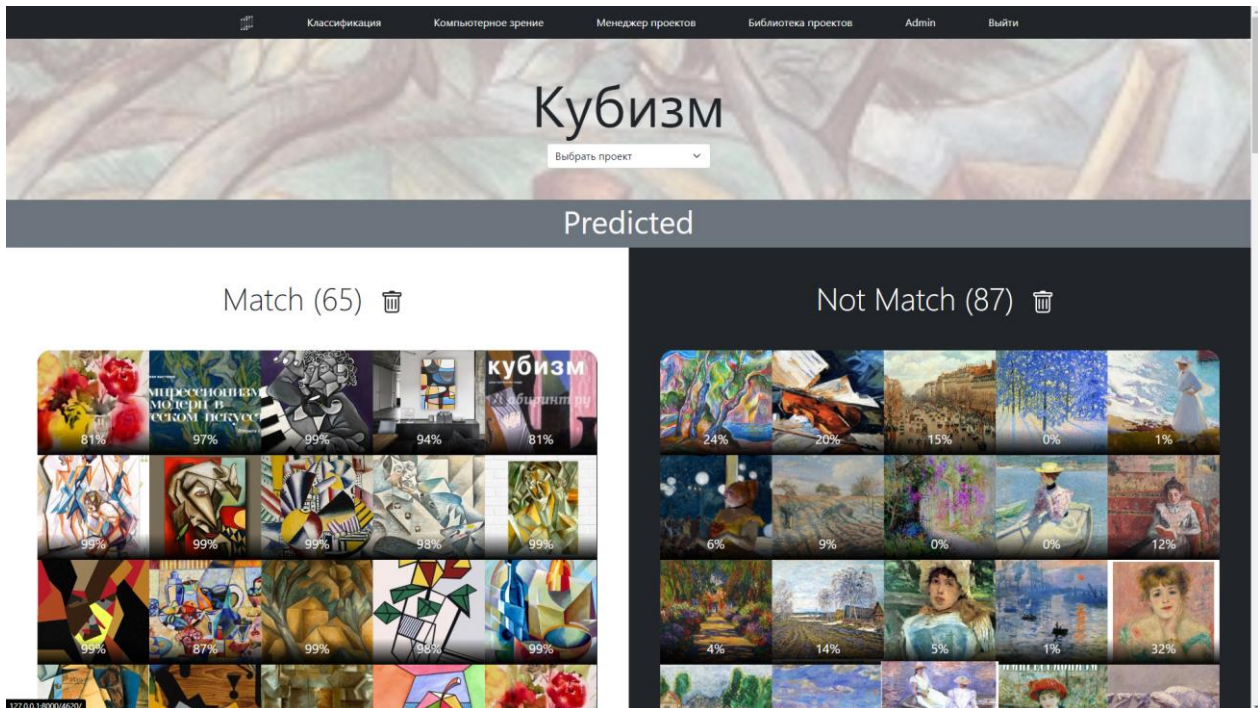
Bootstrap имеет отличную сетку для расположения HTML объектов, в которую входит 12 колонок. Bootstrap помог расположить сетку изображений. Также я использовал встроенные HTML элементы, с прописанными CSS, подходящие нам

Результат

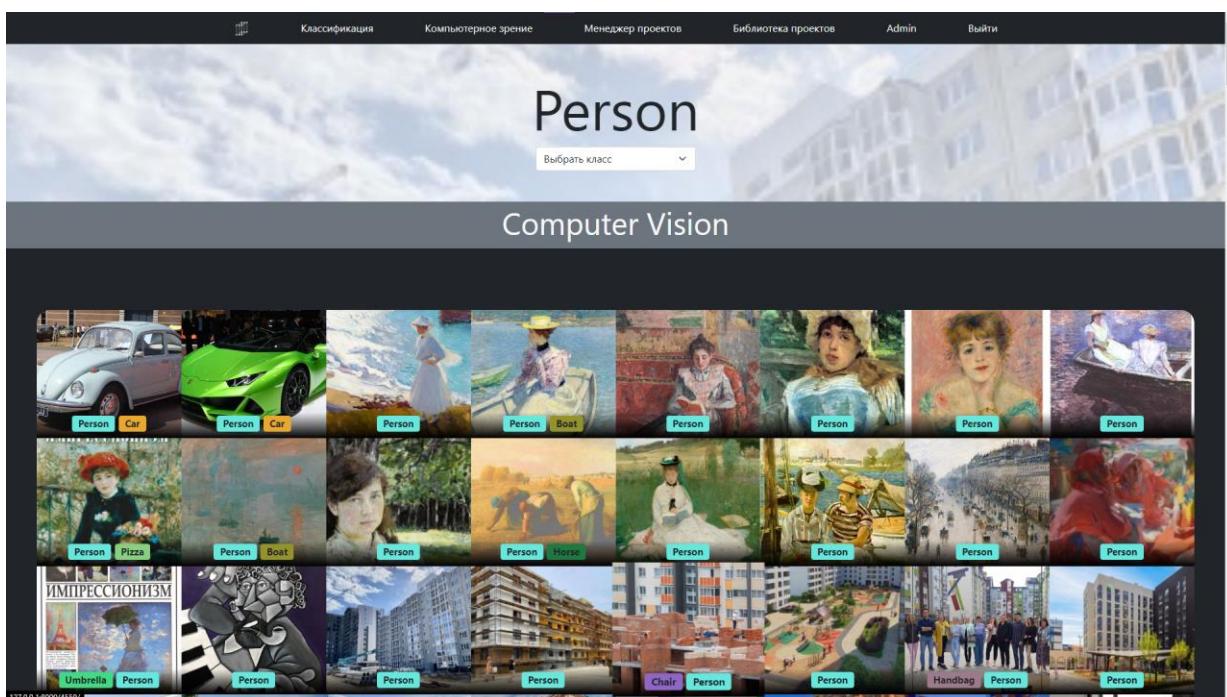
Пример работы программы

1. Web

а. Выборка по объектам классификации



б. Выборка по объектам компьютерного зрения



с. Менеджер проектов

КлассификацияКомпьютерное зрениеМенеджер проектовБиблиотека проектовAdminВыйти

Менеджер проектов

Загрузить фотографии

Классификация

Добавить проект

Название проекта

Порог

Кнопки настройки

Trained

WV Juke

45

Сохранить

Обучить

Удалить

Trained

Глюкометр

96

Сохранить

Обучить

Удалить

Trained

Документы

64

Сохранить

Обучить

Удалить

Trained

Кубизм

50

Сохранить

Обучить

Удалить

Trained

Соцреализм

50

Сохранить

Обучить

Удалить

Trained

Сталинский ампир

70

Сохранить

Обучить

Удалить

Trained

Человек в маске

80

Сохранить

Обучить

Удалить

Компьютерное зрение

parking meter

Просмотреть

sheep

Просмотреть

zebra

Просмотреть

frisbee

Просмотреть

skis

Просмотреть

baseball bat

Просмотреть

tennis racket

Просмотреть

fork

Просмотреть

sandwich

Просмотреть

broccoli

Просмотреть

carrot

Просмотреть

hot dog

Просмотреть

toaster

Просмотреть

d. Обучение

The screenshot shows a web application with a dark header containing navigation links: Галерея, Проекты, Обучить, Предсказать, admin@gmail.com, and Выйти. The main content area has a title 'Обучение в проекте' and a descriptive paragraph: 'Выберите проект и загрузите фотографии, которые вам подходят и не подходят (не менее 20 для каждого статуса) для сортировки по этому проекту. Чтобы предсказания нейросети в дальнейшем имели высокую точность, постарайтесь выбрать разные по цвету, качеству и композиции изображения.' Below this is a white form with the following elements: a dropdown menu 'Выберите проект для обучения' with 'Документы' selected; two sections for selecting photos, each with a 'Выбрать файлы' button and a 'Файл не выбран' status; and a 'Сохранить' button at the bottom.

е. Предсказание

The screenshot shows a web application with a dark header containing navigation links: Классификация, Компьютерное зрение, Менеджер проектов, Библиотека проектов, Admin, and Выйти. The main content area has a title 'Загрузка фото' and a subtitle 'Загрузите фотографии для их классификации и определения объектов на них.' Below this is a white form with the following elements: a 'Load photos' section with a 'Выбрать файлы' button and a 'Файл не выбран' status; a 'Выберите тэги для предсказания' section with a dropdown menu showing 'WV Juke', 'Глюкометр', 'Документы', and 'Кубизм'; a 'Предсказать' button; and a 'Select (1292)' section displaying a list of items with small image thumbnails and text labels like '4639 | ADMIN | 1969_VOLKSWAGEN_BEETLE_DUTCH_REG'.

f. Экран регистрации

Регистрация

Имя пользователя

Email

Пароль

Пароль (еще раз)

Уже есть аккаунт?
[Войти](#)

g. Библиотека проектов

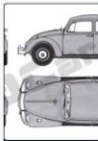
Классификация Компьютерное зрение Менеджер проектов Библиотека проектов Admin Выйти

Библиотека проектов


Search Search

Classification projects


Match




VW Juke
Обновлено:
9 февраля 2023 г. 20:29
Size: 68
[Download](#)




Глюкометр
Обновлено:
12 января 2023 г. 21:32
Size: 33
[Download](#)




Документы
Обновлено:
1 февраля 2023 г. 21:32
Size: 94
[Download](#)



Кубизм
Обновлено:
4 февраля 2023 г. 14:31
Size: 111
[Download](#)




Соцреализм
Обновлено:
2 февраля 2023 г. 18:33




Сталинский ампи
Обновлено:
2 февраля 2023 г. 19:38


Not Match




VW Juke
Обновлено:
9 февраля 2023 г. 20:29
Size: 68
[Download](#)




Глюкометр
Обновлено:
12 января 2023 г. 21:32
Size: 33
[Download](#)




Документы
Обновлено:
1 февраля 2023 г. 21:32
Size: 94
[Download](#)



Кубизм
Обновлено:
4 февраля 2023 г. 14:31
Size: 111
[Download](#)

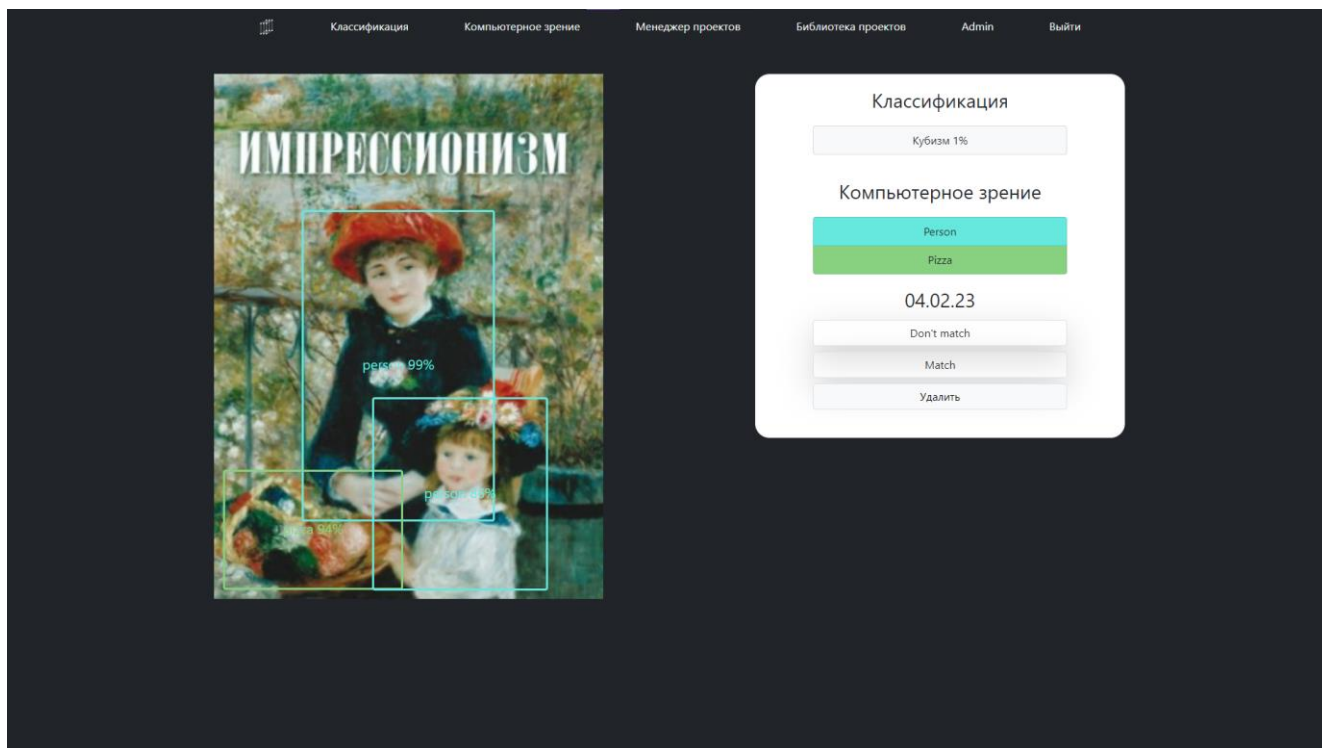


Соцреализм
Обновлено:
2 февраля 2023 г. 18:33

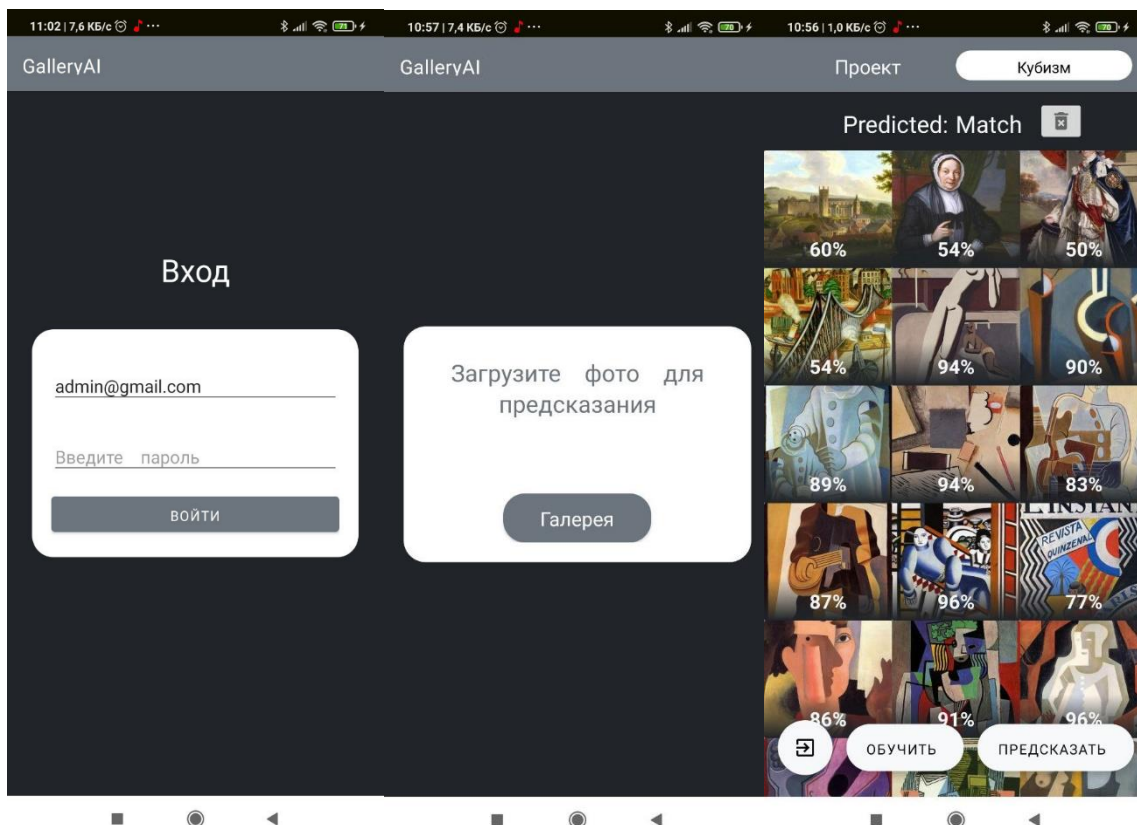


Сталинский ампи
Обновлено:
2 февраля 2023 г. 19:38

h. Просмотр подробной информации об изображении



2. Android



Точность ML модели

Precision (точность) и recall (полнота) - наиболее популярные и полезные метрики при оценке работы нейросетевых классификаторов. В нашем случае precision и recall используются для оценки нейронной сети, классифицирующей объекты по 2 классам. Важным понятием в задачах оценки точности является матрица ошибок (confusion matrix). В нашем случае она выглядит так:

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

(Здесь \hat{y} - предсказание классификатора, y - истинный класс объекта)

$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$ - наиболее очевидная матрица, однако почти никогда не используемая. Использовать *Accuracy* можно лишь в ситуации, в которой при оценке классификатора использовалось равное количество элементов с $y = 1$ и с $y = 0$. В противном случае значение *Accuracy* может быть достаточно высоким даже в ситуации, когда классификатор полностью лишён предсказательной силы.

Precision и *Recall* считаются следующим образом:

$$Precision = \frac{TP}{TP + FP}; Recall = \frac{TP}{TP + FN}$$

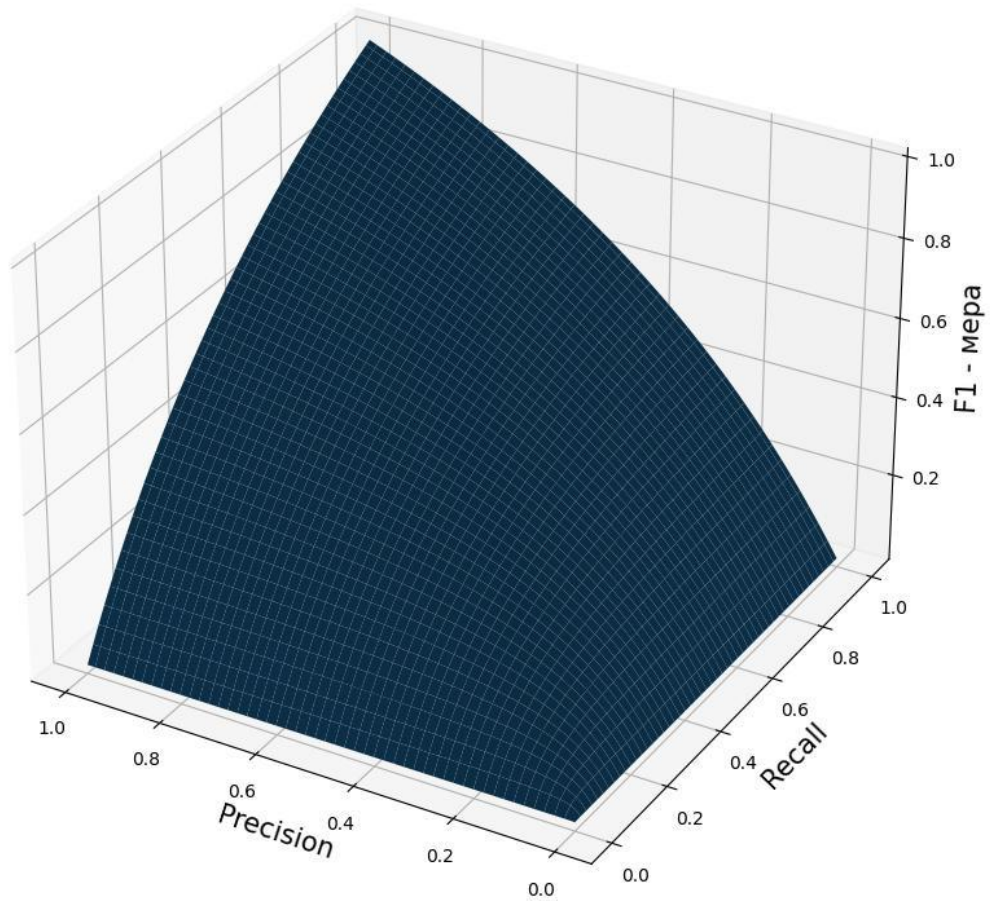
Precision и *Recall* можно интерпретировать следующим образом: *Precision* - доля объектов, названных классификатором положительными ($\hat{y} = 1$), и при этом действительно являющимися положительными. *Recall* - доля верно классифицированных объектов положительного класса ($y = 1$).

Затем *Precision* и *Recall* необходимо объединить в единый критерий качества. Для этого существует специальная метрика, называемая F_β - мера.

$$F_\beta = (1 + \beta^2) * \frac{Precision * Recall}{(\beta^2 * Precision) + Recall}$$

β символизирует то, насколько один параметр важнее другого. В нашем случае $\beta = 1$, т. к. *Precision* и *Recall* в нашем проекте важны одинаково. Следовательно мы будем использовать F_1 - меру

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$



Пусть декартова система координат $(Precision, Recall, F_1)$ образована векторами i, j, k . Тогда:

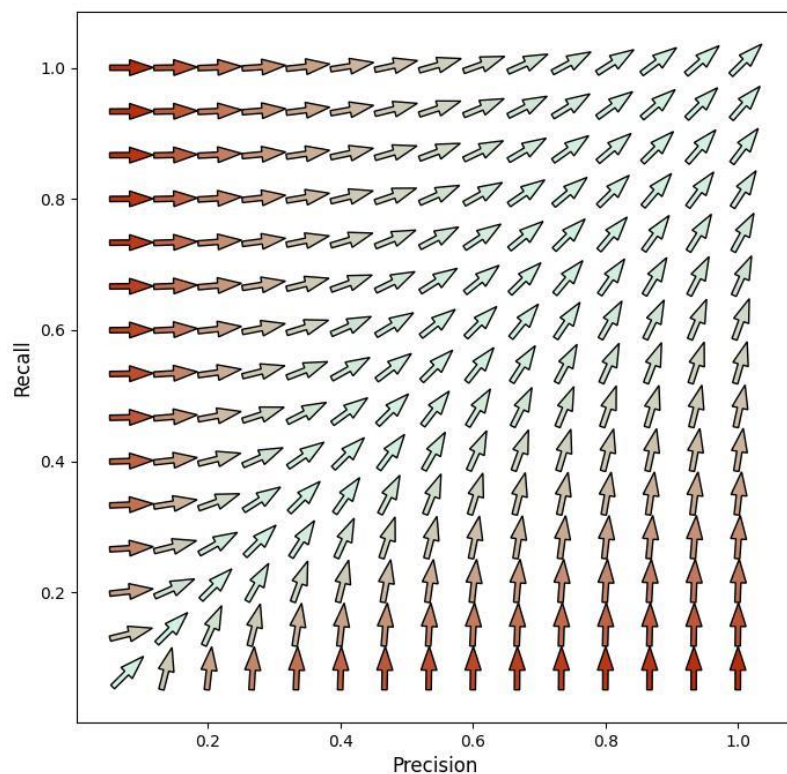
$$\frac{\delta F_1}{\delta Precision} = \frac{2 * Recall^2}{(Precision + Recall)^2}$$

$$\frac{\delta F_1}{\delta Recall} = \frac{2 * Precision^2}{(Precision + Recall)^2}$$

\Rightarrow

$$grad(F_1) = \left(\frac{2 * Recall^2}{(Precision + Recall)^2} \right) * i + \left(\frac{2 * Precision^2}{(Precision + Recall)^2} \right) * j$$

$grad(F_1)$:



Цвет тут служит для отображения длины каждого из векторов. Чем темнее цвет - тем длиннее вектор

Легко видеть, что кроме точки (1, 1) график стремится изменяться к отрезку, расположенному диагонально от точки с координатами (0, 0) до точки с координатами (1, 1). Это означает, что нам необходимо найти такое пороговое значение, что:

$$Precision = Recall$$

(Результат работы модели - вероятность принадлежности к первому или второму классу, однако для окончательной классификации необходимо определить пороговое значение вероятности.)

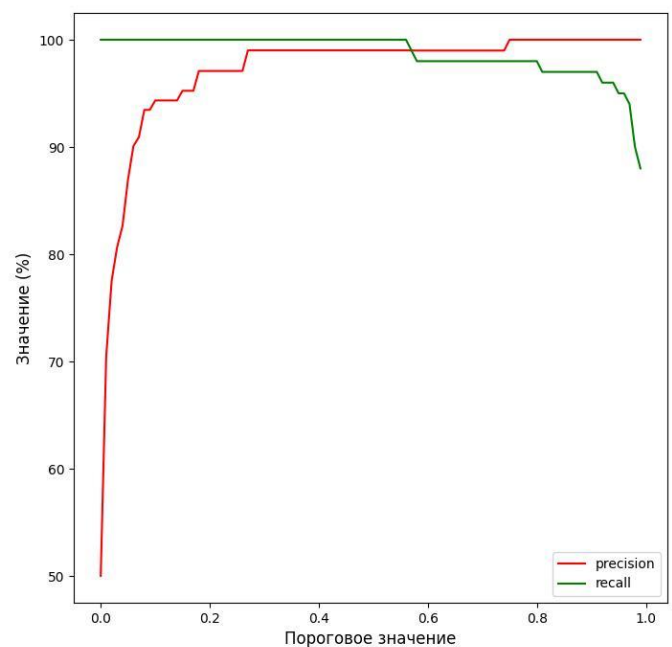
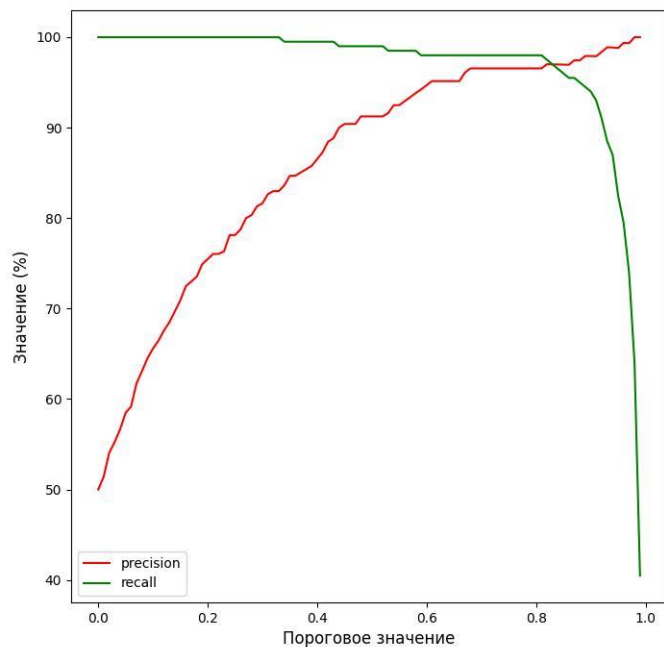
Ниже представлены зависимости значений Precision и Recall от различных пороговых значений.

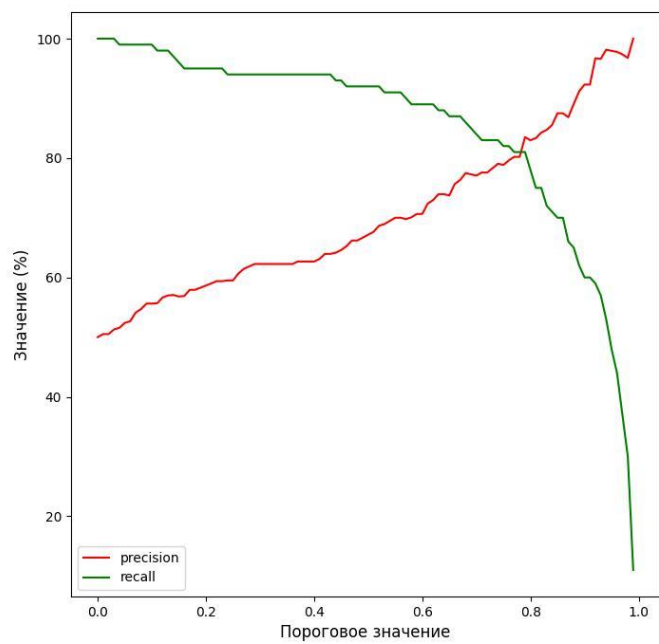
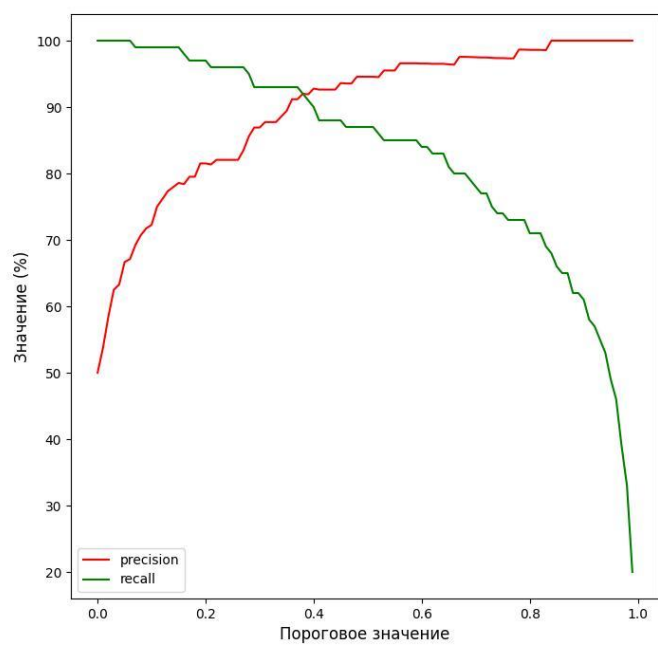
1. Классификация кошек и котов
2. Классификация конспектов
3. Классификация размытых фотографий
4. Классификация кубизма

EER и пороговое значение

$$EER (Equal Error Rate) = 100\% - Precision \text{ (или Recall)}$$

Название категории	Equal Error Rate	Пороговое Значение
Классификация кошек и котов	3.01%	0.83
Классификация конспектов	1.13%	0.57
Классификация размытых фотографий	7.96%	0.38
Классификация кубизма	19.82%	0.77





Заключение

В дальнейшем планируется добавление следующих функций:

1. Распознавание лиц, категоризация по людям.

На данный момент нейронная сеть способна отличать одного конкретного человека от всех остальных, если разметить 20 фотографий этого человека. Однако в этой ситуации персональное обучение демонстрирует себя не с лучшей стороны. У аналогичных проектов (например, во встроенной библиотеке фотографий apple) возможность классификации фотографий появляется начиная с единственного примера искомого человека. Для реализации этой возможности необходимо изучить, при помощи каких технологий она работает и добавить её в проект как дополнительную функцию, отдельно от технологии персонального обучения.

2. Улучшение технологии обучения нейронной сети:

- а. Увеличение точности
- б. Уменьшение требуемого количества фотографий для обучения (Путём внедрения технологии One-shot learning)

Технология One-shot learning несколько отлична от технологии transfer learning. При использовании технологии One-shot learning достаточно представить нейросети один пример положительного и один пример отрицательного класса. Однако закономерности, основываясь на которых объект причисляется к тому или иному классу, могут быть выведены неправильно и точность работы такой модели будет ожидаемо низкой.

3. Поддержка устройств на iOS

Разработка на iOS, как и на некоторых других операционных системах устройств компании Apple, требует изучения языка программирования Swift. В силу отсутствия у нас опыта взаимодействия с этим языком, эта задача кажется наиболее трудной. Также, весь рынок приложений на iOS контролируется компанией Apple, которая может лишить программиста возможности публиковать свои продукты в магазине приложений iOS.

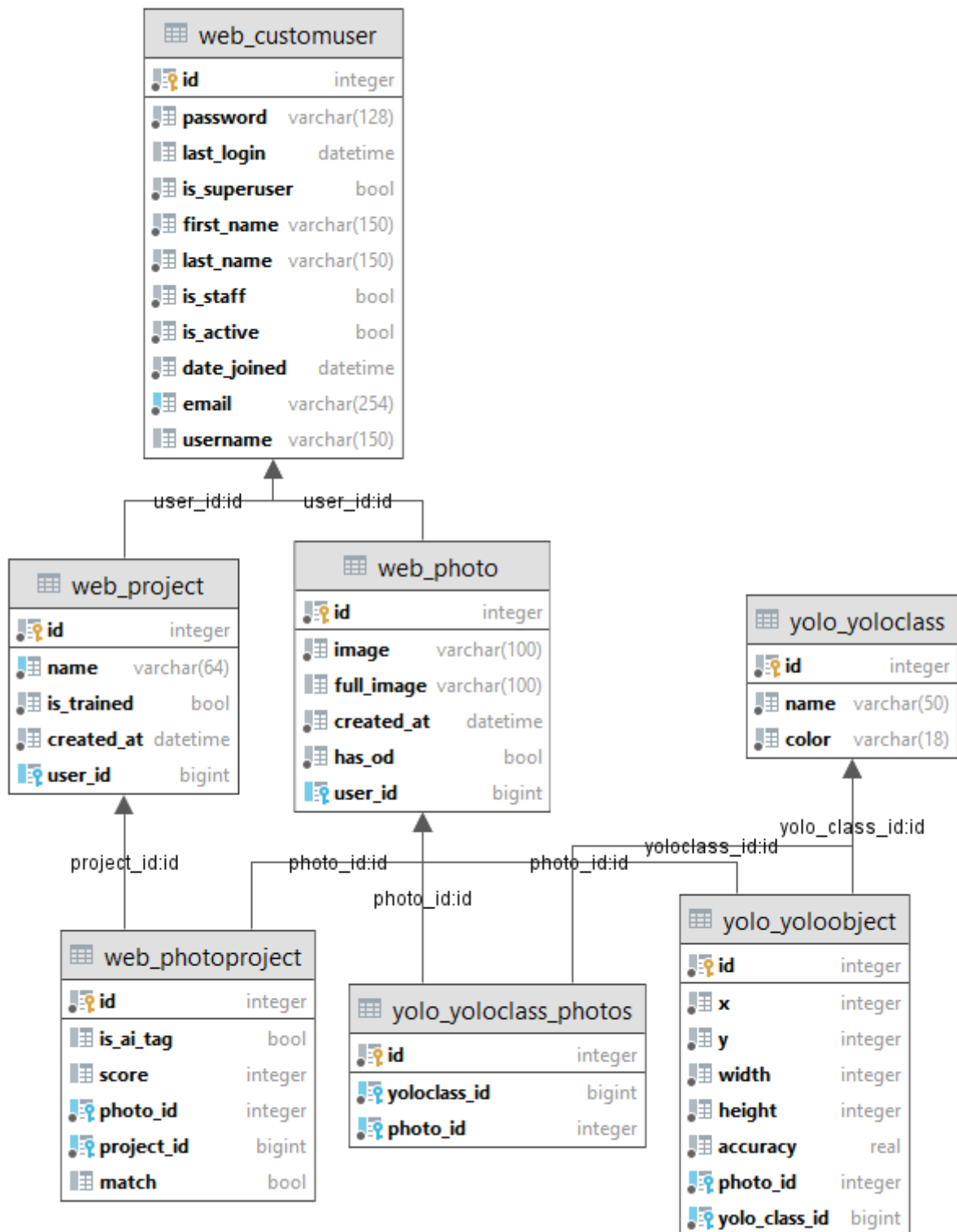
Мы считаем, что тема не утратила своей актуальности и нами был избран верный вариант её реализации. Стоит отметить, что наш проект приобретает дополнительную ценность если взять во внимание то, что все аналоги созданы вне России и управляются, точно так же, извне. Таким образом на отечественном рынке нет аналогов нашего проекта.

Список Литературы

1. Tensorflow [Электронный ресурс] : Электронный справочник по библиотеке tensorflow / Google — Электрон. справ. — Режим доступа: https://www.tensorflow.org/api_docs (Дата обращения - 07.06.2022)
2. RQ: Workers [Электронный ресурс] : Электронный справочник по библиотеке RQ: Workers / Vincent Driessen — Электрон. справ. — Режим доступа: <https://python-rq.org/docs/workers/> (Дата обращения - 07.06.2022)
3. Manali Shaha, Meenakshi Pawar. Transfer Learning for Image Classification / Manali Shaha, Meenakshi Pawar. // 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA 2018) (Коимбатур, 2-4 июля 2018 г.).
4. Peter Zhang. Neural Networks for Classification: A Survey // IEEE Transactions on Systems Man and Cybernetics Part C. — 2000. — № 3. — С. 451 - 462.
5. Django documentation [Электронный ресурс] : Электронный справочник по фреймворку Django/ Django Software Foundation — Электрон. справ. — Режим доступа: <https://docs.djangoproject.com/en/4.0/> (Дата обращения - 15.06.2022)
6. Documentation for app developers [Электронный ресурс] : Электронный справочник для программистов на Android/ Google — Электрон. справ. — Режим доступа: <https://developer.android.com/docs> (Дата обращения - 15.06.2022)
7. Django REST framework [Электронный ресурс] : Электронный справочник по API Django / MkDocs — Электрон. справ. — Режим доступа: <https://www.django-rest-framework.org/> (Дата обращения - 15.06.2022)
8. Mark, Richards Software Architecture Patterns / Richards Mark. — 1-е изд. — Севастопол : O'Reilly Media, 2015. — 54 с.

Приложения

Схема БД



Powered by yFiles