МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

Кафедра №43 «Компьютерных технологий и программной инженерии»

ОТЧЁТ ПО ПРАКТИКЕ
ЗАЩИЩЁН С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

Ст. преп.		М. Д. Поляк
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЁТ ПО ПРАКТИКЕ

вид практики производственная

тип практики по получению профессиональных умений и опыта профессиональной деятельности

на тему индивидуального задания Разработка нейросетевой модели определения возраста человека по фотографии лица

выполнен Цыбиным Дмитрием Андреевичем
фамилия, имя, отчество обучающегося в творительном падеже

по направлению подготовки 09.03.04 (код) Программная инженерия (наименование направления)

наименование направления

направленности 02 (код) Проектирование программных систем (наименование направленности)

наименование направленности

Обучающийся группы № 4232 (номер) _____ (подпись, дата) Д. А. Цыбин (инициалы, фамилия)

Санкт–Петербург 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном цифровом пространстве компьютерное зрение стало одной из важнейших технологией. Благодаря этой технологии обнаруживаются дефекты на производстве, контролируется регулировка светофоров и многие другие задачи, при этом перечень таких задач постоянно расширяется. Востребованной практической задачей в этой области является автоматическое определение возраста человека по изображению. Эта технология может применяться в широком спектре областей: от персонализированного маркетинга и адаптивного контента до систем безопасности и биометрической верификации. Например, в розничной торговле возрастная идентификация позволяет предлагать релевантные товары, а в банковской сфере – автоматически проверять соответствие возрастных ограничений при оказании услуг.

Точное определение возраста остается сложной задачей. На визуальное восприятие возраста влияют множество факторов: этническая принадлежность, генетические особенности, образ жизни, освещённость и мимика в момент съемки. Однако современные нейросетевые модели могут идентифицировать человека при разном освещении и разных ракурсах, хотя, разумеется, различные помехи негативно влияют на итоговый результат. Более того существуют методы полностью запутать нейросеть: специальный макияж, очки и т.п.

Описание предметной области

Для использования моделей определения возраста, необходимо, в первую очередь, выделить лицо человека на изображении. Для обнаружения лиц существует несколько различных инструментов:

- Каскад Хаара – использует набор примитивов Хаара (чёрно-белые прямоугольные фигуры) для обнаружения объектов, в более поздних вариациях могут использоваться наклонные признаки Хаара. Суть каскада в отсеивании областей, где искомый предмет отсутствует. Имеет хорошую эффективность в идеальных условиях (отсутствие помех, высокое качество изображения и положение лица – анфас), но при отклонении от таких условий, эффективность заметно снижается и корректное обнаружения лица практически невозможно.
- HOG (Histogram of Oriented Gradients) – Гистограмма направленных градиентов. Изображение делится на небольшие ячейки, и для каждой ячейки строится гистограмма направлений градиентов. Эти гистограммы затем объединяются в блоки и нормализуются, комбинация этих гистограмм и является дескриптором.
- Модели на основе глубокого обучения (например MTCNN) – используют глубокие нейронные сети для обнаружения лиц. Высокая точность даже в сложных условиях, включая изменения в освещении, позе и качестве изображения. Требует больше вычислительных ресурсов.

Точность определения, в том числе в сложных условиях, намного важнее скорости работы, поскольку создаваемая система не предполагает использование на большом количестве экземпляров и работы в реальном времени, а жесткие требования к качеству фотографии и положению лица сильно ограничивают возможности использования и снижают удобство пользования. Поэтому для определения лица была выбрана модель MTCNN, поскольку, несмотря на низкую скорость работы, на порядок меньше скорости метода Виолы–Джонса (0.6-0.9 и 0.04-0.05 сек соответственно), имеет лучшую

точность из всех рассматриваемых вариантов (F-мера: 0.89 и 0.76 для MTCNN и метода Виолы–Джонса соответственно) [1].

После необходимо вычислить эмбеддинги, с помощью одной из моделей:

- FaceNet – использует глубокую сверточную нейронную сеть (CNN обученная с помощью SGD и AdaGrad[2]) для преобразования изображений лиц в компактные векторные представления (эмбеддинги). Расстояние между эмбеддингами одного и того же лица – минимально, а между эмбеддингами разных лиц — максимально.
- VGG-Face – создана на основе VGG (модель, созданная на основе CNN) для задачи распознавания лиц. Использует каскады из последовательных сверточных слоев, которые работают вместе для извлечения более сложных признаков на разных уровнях абстракции. Строгая последовательность операций, высокие вычислительные затраты и использование памяти.
- Vision Transformer – обрабатывает изображения как последовательность патчей (изображение разбивается на патчи), улучшая производительность, однако требует много данных для обучения. Лучше эффективность на больших данных и интерпретируемость, чем модели, основанные на CNN.

Для вычисления эмбеддингов используется FaceNet.

Получив эмбеддинги, используется обученная модель для предсказания возраста. Тип задачи – регрессия, возможные используемые модели:

- SVR (Support Vector Regression) – алгоритм регрессии, основанный на алгоритме классификации SVM;
- Линейная регрессия - прогнозирование значений на основе линейной комбинации независимых признаков, путём поиска линейной функции, которая наилучшим образом объясняет наблюдаемые данные.
- L1 и L2 регрессии – линейная регрессия с добавлением регуляризации (LASSO и Ridge соответственно);

- Регрессия дерева решений – алгоритм регрессии, основанный на алгоритме классификации дерева решений;

Перечень решаемых задач

Анализ предметной области:

- Изучение современных подходов к определению возраста – рассмотрение различных алгоритмов прогнозирования возраста и их составляющих, в том числе: моделей вычисления эмбеддингов, обнаружения лиц и прогнозирования.

Подготовка данных для обучения:

- Сбор датасета изображений лиц с указанием возраста – поиск или создание датасета с достаточным количеством изображений, подходящих для обучения модели прогнозирования возраста по фото. Такой датасет должен содержать фотографии лиц всех групп возрастов и каждое изображения должно иметь возраст человека, представленного на нём.
- Обработка изображений – приведение всех фотографий к одному виду, если необходимо.
- Получение эмбеддингов изображений – использование модели вычисления эмбеддингов и создание тренировочной и тестовой выборок, где признаками являются вычисленные значения, а целевая переменная – возраст.
- Предобработка выборок – масштабирование.

Разработка модели:

- Создание ансамбля – поиск наилучших моделей, путём сравнения различных регрессионных моделей, и их объединение для повышения точности предсказания.
- Обучение модели – обучение созданной модели на тренировочных данных и сохранение обученной модели.
- Детекция лиц с помощью MTCNN и обработка изображения для предсказания – создание алгоритма предобработки изображения для реализации предсказания (для предсказания обученной модели требуются эмбеддинги, следовательно необходимы произвести аналогичные получению эмбеддингов изображений действия).

- Оптимизация гиперпараметров – нахождение гиперпараметров для моделей ансамбля, имеющих наилучшие метрики качества.

Экспериментальная оценка:

- Расчет метрик качества:
  - MSE (Mean Squared Error) – Средний квадрат отклонения;
  - R2-метрика – метрика, показывающая качество модели, где 0 – угадывание, 1 – идеальная модель;
  - MAE (Mean Absolute Error) – среднее абсолютное отклонение.
- Сравнение с существующими решениями – сравнение метрик созданной модели с метриками аналогов.

Описание входных и выходных данных

Входные данные:

Для обучения модели предсказания использовался датасет UTKFace, содержащий более 23000 изображений лиц людей от 0 до 116 лет, средний возраст – 35.5 лет. Особенностью данного датасета является сильно обрезанные изображения, часто обрезаны не только фон и части тела, отличные от лица, но и волосы и даже части лица. Таким образом лицо всегда занимает более 80% фотографии.

На вход подаётся фотография с лицом человека, однако, учитывая датасет обучения, требуется дополнительно обрезать изображение таким образом, чтобы лицо занимало не менее 80%. Пример изображения из датасета, используемого для обучения, показан на рисунке 1.

Рисунок 1 – пример изображения из датасета UTKFace

Для этого применяется MTCNN, результатом работы которого является предполагаемое местоположение лица на изображении. MTCNN – каскад свёрточных нейронных сетей, состоит из трёх сетей: первая находит множество областей, где, предположительно, находятся лица, вторая и третья отсеивают области, не содержащие лиц, помимо этого третья сеть обнаруживает пять лицевых точек-ориентиров (глаза, нос и уголки рта) рисунок 2.

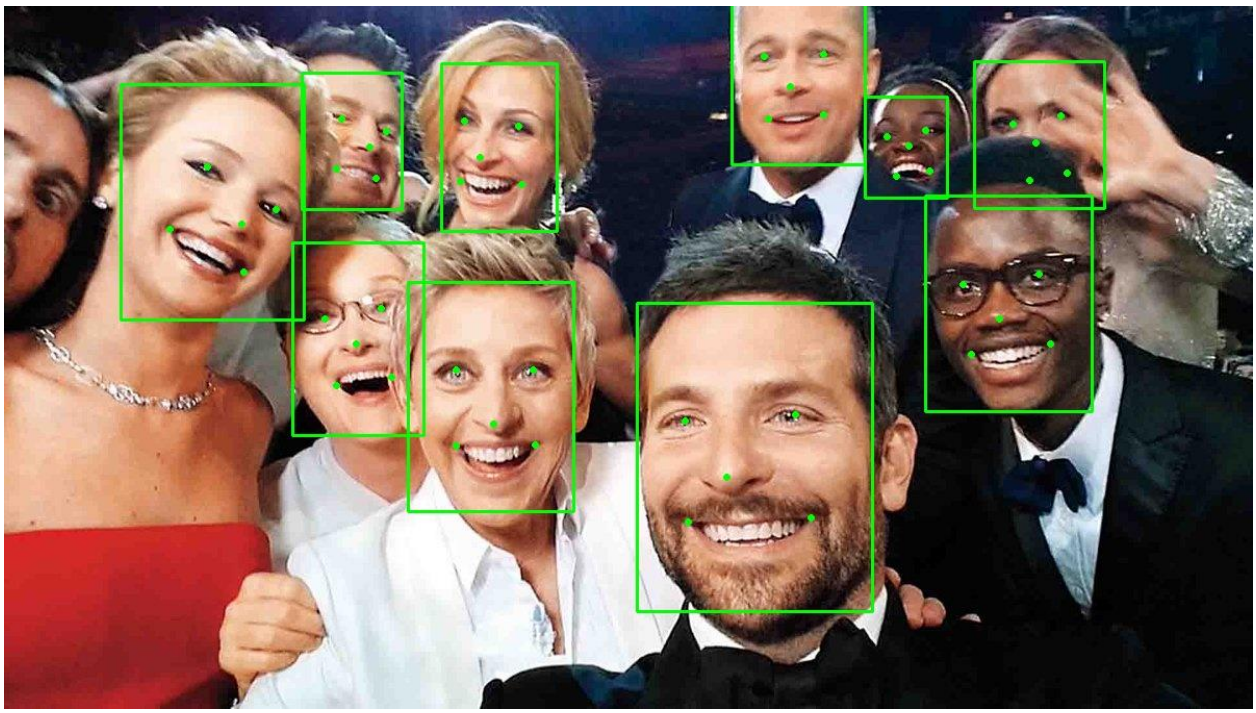

Рисунок 2 – Обнаружение нескольких лиц и расстановка точек-ориентиров

Так как MTCNN обнаруживает все лица на фотографии, для предсказания выбирается лицо с наибольшей уверенностью (пример работы MTCNN показан на рисунке 3).

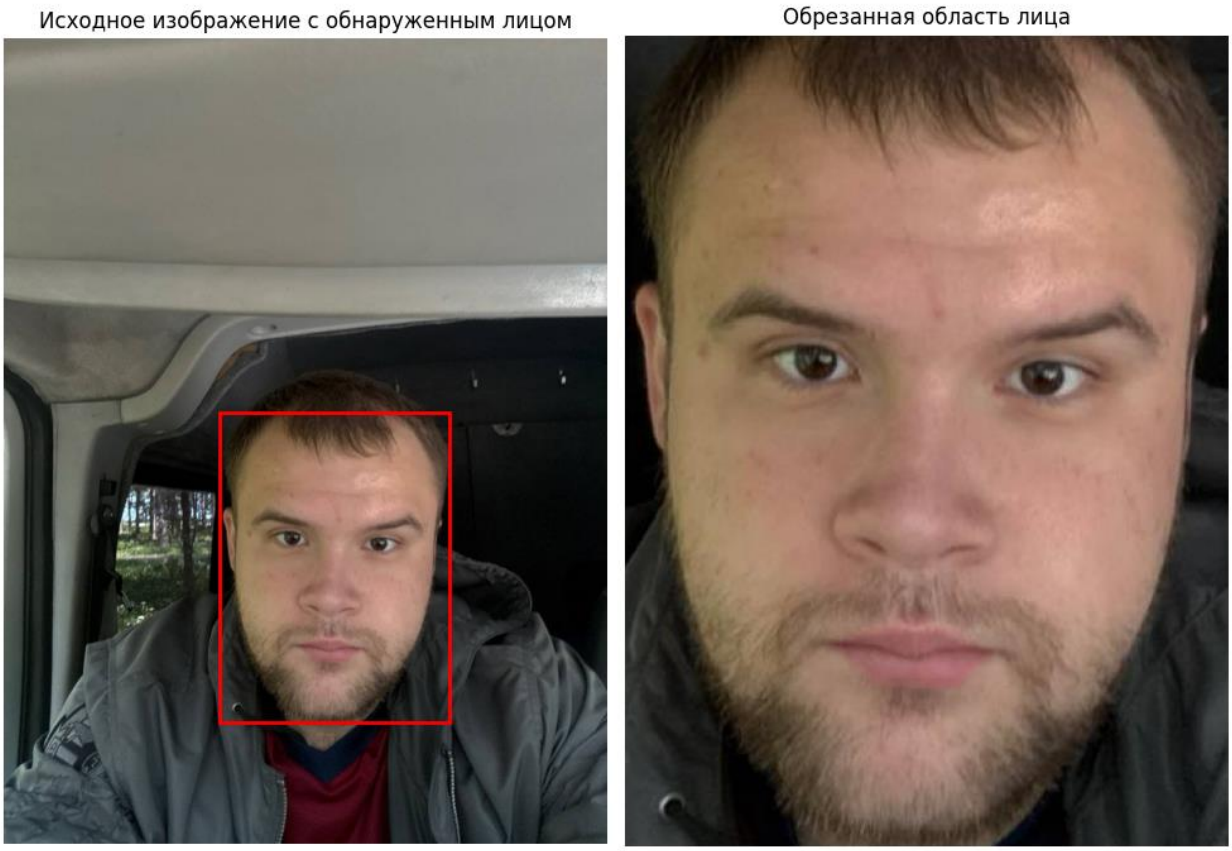

Рисунок 3 – исходное изображение и результат поиска лица

Однако иногда может происходить неудачное определение, пример такого случая показан на рисунке 4 (очевидно, что на данном изображении фокус был сделан на другом человеке, хотя лицо на фотографии было найдено верно).

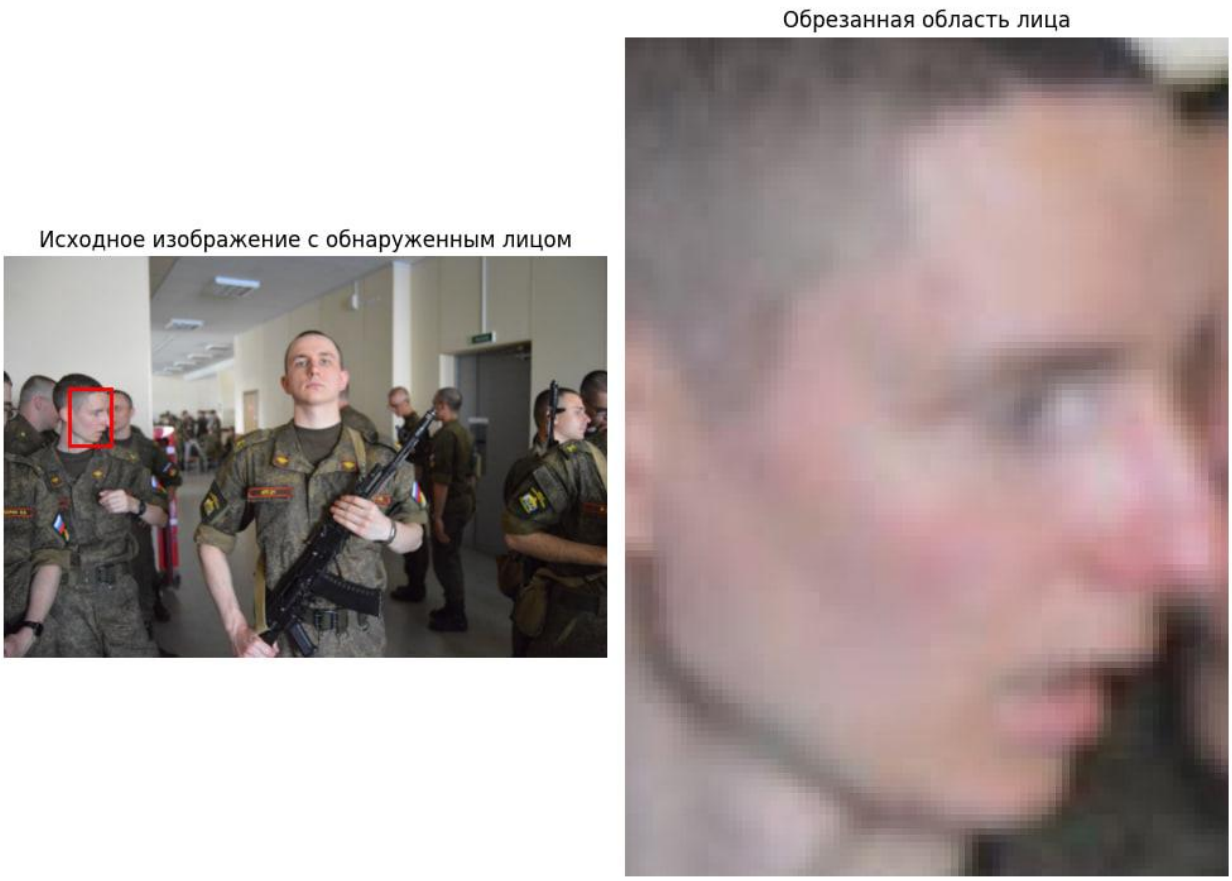

Рисунок 4 – относительно неудачный результат поиска лица

Однако для вычисления эмбеддинга лица необходимо привести изображение к виду: (1, 160, 160, 3), т.е. изображение 160x160 пикселей в трёхканальном RGB.

После приведения изображения к соответствующему виду, с помощью FaceNet вычисляется эмбеддинг. Вычисленный набор значений передаётся в обученную модель, которая делает предсказание возраста человека на фотографии.

Выходные данные:

Предполагаемый возраст человека на фото, предсказанный моделью по полученному эмбеддингу.

Используемые технологии и языки программирования

Для выполнения задания использовались следующие инструменты:

- Google Colab – облачная платформа для разработки, удобство разработки и возможность синхронизированной работы с несколькими устройствами.
- Язык программирования Python
- Датасет UTKFace – содержит набор из более 23000 фотографий лиц от 0 до 116 лет, возраст указан в имени файла, также в имени файла указан пол, однако для задачи определения возраста это не имеет значения.
- Модель FaceNet – для вычисления эмбеддингов изображений лиц. Принимает на вход изображение 160x160 пикселей в формате RGB, возвращает 128-мерный эмбеддинг лица.
- Модель MTCNN – для обнаружения лица на изображении и вырезания фона изображения. Чтобы соответствовать датасету, на котором обучалась модель, требуется приводить изображения к соответствующему виду (лицо должно занимать более 80% изображения).
- SVR – для выполнения задачи предсказания возраста человека по эмбеддингу изображения лица. Используются ансамбль из двух моделей SVR с разными ядрами, причём влияние на итоговый результат RBF-ядра в два раза больше (2 : 1):
  - RBF (Гауссово) ядро;
  - Polynomial (Полиноминальное) ядро.
- Библиотека SKLearn – для обработки входных данных, использования SVR, создания ансамбля и оценки модели.
- Google Drive – для сохранения, загрузки и последующего использования созданной модели, модели FaceNet, вычисленных эмбеддингов и датасета.

Задача предсказания возраста человека по фото – регрессия. Для выполнения этой задачи рассматривались различные алгоритмы регрессии.

Для выбора наилучшего варианта, проводилось сравнение между основными типами регрессионных моделей: линейная (LASSO и Ridge), дерево решений (XGBoost) и регрессия опорных векторов (SVR с ядрами RBF и полиномиальным).

В ходе тестирования были получены результаты, показанные в таблице 1. Методы регрессии, основанные на SVM, показали лучшие результаты. LASSO оказался наихудшим, а Ridge и XGBoost, хоть и заметно лучше, но всё же уступают методу опорных векторов.

Таблица 1 – Метрики методов

Метрики / Методы	MSE	R2
SVR (RBF)	57.23	0.8517
SVR (Poly)	58.33	0.8488
LASSO	110.31	0.7141
Ridge	82.35	0.7866
XGBoost	72.97	0.8027

Учитывая результаты сравнения полученных метрик, итоговый вид модели прогнозирования: ансамбль из двух моделей SVR с ядрами: RBF и полиномиальное, при этом итоговый результат – усредненные значения каждой модели, при этом влияние результата модели с ядром RBF в два раза больше, так как она имеет лучшие показатели точности. Метрики модели, полученной с использованием ансамбля показаны в таблице 2.

Таблица 2 – Метрики созданного ансамбля

Метрики / Методы	MSE	R2	MAE
Ансамбль (SVR (RBF) + SVR (Poly))	53.31	0.8618	5.19

Результаты оценки ансамбля показывают, что полученная модель прогнозирования превосходит любые одиночные методы по обеим метрикам. Улучшения метрики MSE на почти 7% и метрики R2 более чем на 0.01 (при условии, что 1 – показатель идеальной модели, недостижимое значение).

Результат выполнения работы

В результате работы была получена программа предсказания возраста человека по фото. В алгоритм входит обученная модель предсказания возраста, модели определения лица и расчёта эмбеддинга, а также алгоритмы преобразования изображений, необходимые для корректной работы нейросетевых моделей.

Изображение, загруженное пользователем, обрабатывается, обнаруживается лицо, и после обрезки вычисляется эмбеддинг. Далее делается прогноз возраста человека с помощью обученной модели предсказания. Пользователю выводится исходное изображение и предсказанный возраст, округлённый до десятых. Результат предсказания и диаграмма деятельности показаны на рисунках 5 и 6 соответственно.

Рисунок 5 – Результат предсказания

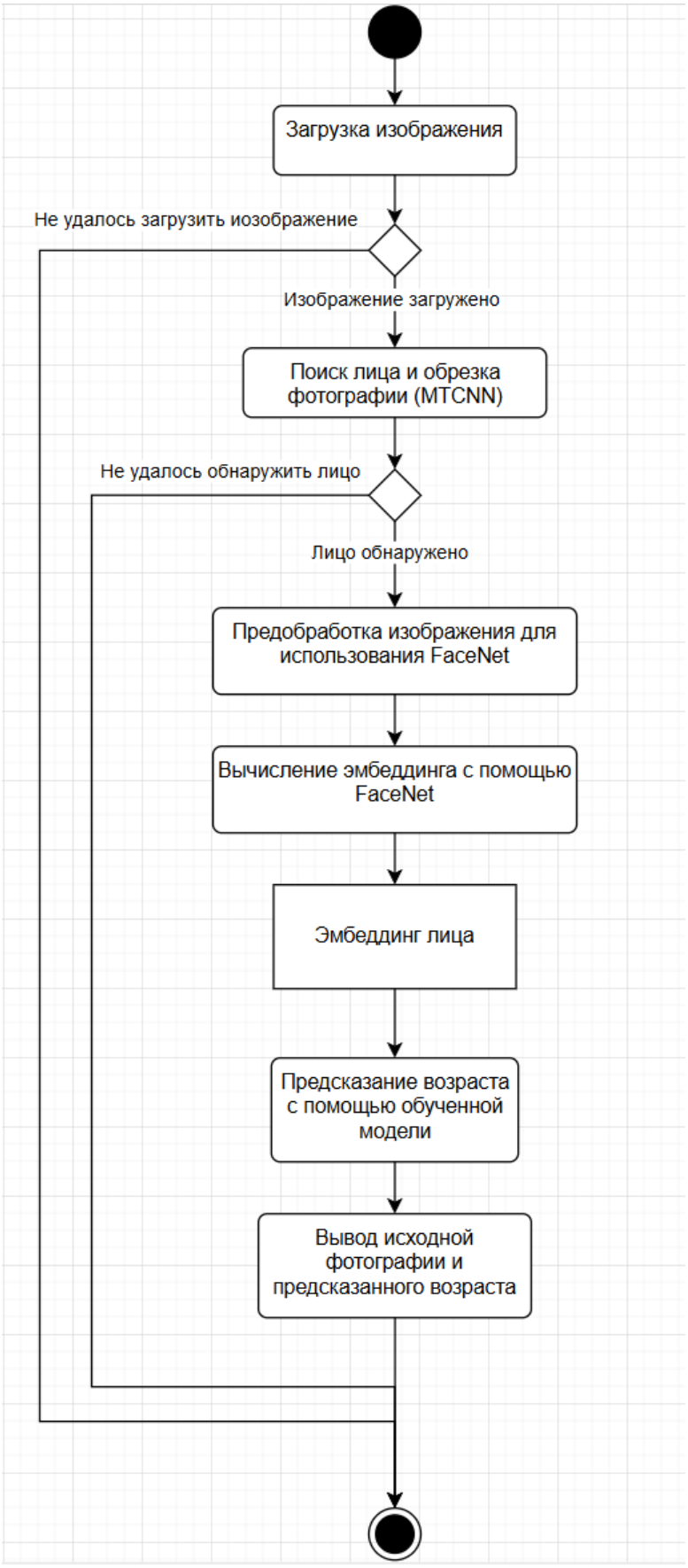

Рисунок 6 – Диаграмма деятельности

Среднее абсолютное отклонение созданной модели – 5.19. Автор MiVOLO[3] заявляет о MAE = 3.47 и о превосходстве своей модели над любыми аналогами, однако данная модель анализирует не только лицо, но и тело человека на фотографии (модели того же разработчика, основанные только на анализе лица имеют абсолютное отклонение равное 4.23). Также еще в двух статьях было указано значение MAE 5.4 и 5.46.

Однако все примеры имеют возраст более двух лет, следовательно относительные хорошие показатели полученной модели могут быть связанны с частичным устареванием сравниваемых аналогов.

Подводя итоги сравнения, можно сказать, что созданная модель имеет приемлемые показатели, но, очевидно, не является эталоном. Точность модели позволяет использовать её для каких-либо нужд.

ЗАКЛЮЧЕНИЕ

В ходе выполнения производственной практики была разработана нейросетевая модель определения возраста человека по фотографии лица. Разработанная модель предсказывает точное значение возраста с округлением до десятых с погрешностью около пяти лет.

Выполнен весь перечень задач: проанализирована предметная область, подготовлены данные для обучения, разработана модель предсказания возраста, а также проведена оценка созданной модели, в том числе в сравнении с аналогами.

Перспективы дальнейшего развития: улучшение ансамбля прогнозирования (добавление новых моделей, оптимизация гипперпараметров), добавление определения возраста для нескольких лиц на одном изображении, создание удобного интерфейса и метода загрузки изображения для предсказания и добавление определения пола человека на фотографии.

Подводя итоги, алгоритм, разработанный в результате прохождения производственной практики, выполняет свою задачу с хорошей точностью и выполняет все необходимые преобразования изображения, так что пользователь не ограничен жесткими критериями загружаемой фотографии.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Egorov A., Idiyatullin A., Zakirov A. Comparison of the Parametrically Optimized Implementation of Viola–Jones Object Detection Method and MTCNN // 2021 IV International Conference on Control in Technical Systems (CTS). Saint Petersburg, 2021. P. 246–248.

2 Schroff F., Kalenichenko D., Philbin J. FaceNet: A Unified Embedding for Face Recognition and Clustering // 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Boston, 2015. P. 815–823.

3 Kuprashevich M., Tolstykh I. MiVOLO: Multi-input Transformer for Age and Gender Estimation [Электронный ресурс] // arXiv. 2023. URL: https://arxiv.org/abs/2310.15108 (дата обращения: 17.07.2025).

ПРИЛОЖЕНИЕ

Приложение А – Листинг :

```
class AgePredictor:
    def __init__(self, embedding_model_path):
        """Инициализация"""
        # Загрузка модели для получения эмбеддингов лиц
        self.embedding_model = tf.saved_model.load(embedding_model_path)
        self.embedder = self.embedding_model.signatures['serving_default']

        # Масштабирование признаков
        self.scaler = StandardScaler()

        # Ансамбль
        self.ensemble = None

        # Отдельные модели
        self.individual_models = {}

        # Метрики оценки
        self.evaluation_metrics = {}
        try:
            # Детектор лиц
            self.face_detector = MTCNN()
        except Exception as e:
            print(f"Ошибка инициализации MTCNN: {str(e)}")
            self.face_detector = None

    def cutting_face(self, img):
        """Обнаружение лица на изображении"""
        if self.face_detector is None:
            return img

        # Обнаружение лиц на изображении
        detections = self.face_detector.detect_faces(img)
        if not detections:
            return None

        # Лицо с наибольшей уверенностью
        best_face = max(detections, key=lambda x: x['confidence'])
        x, y, w, h = best_face['box']

        # Отступ 10%
        margin = 0.1

        # Корректировка отступа
        x = max(0, int(x - margin * w))
        y = max(0, int(y - margin * h))
        w = min(img.shape[1] - x, int(w * (1 + 2 * margin)))
        h = min(img.shape[0] - y, int(h * (1 + 2 * margin)))
```

```
        return img[y:y+h, x:x+w]

    def preprocess_image(self, img_path):
        """Загрузка и предобработка изображения"""
        # Загрузка изображения
        img = cv2.imread(img_path)
        if img is None:
            print("Не удалось загрузить изображение")
            return None

        # Перевод в необходимый формат для работы MTCNN
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

        # Обнаружение и обрезка лица
        face = self.cutting_face(img)
        if face is None:
            print("Лицо не обнаружено")
            return None

        # Изененние размера и перевод в float32 (требования FaceNet)
        face = cv2.resize(face, (160, 160))
        face = face.astype(np.float32)

        # Нормализация
        face = (face - 127.5) / 128.0

        return face

    def predict_age_from_image(self, img_path):
        """Предсказание возраста"""
        # Получение эмбеддинга
        embedding = self.get_face_embedding(img_path)
        if embedding is None:
            return None

        # Масштабирование
        scaled_embedding = self.scaler.transform([embedding])

        # Предсказание возраста
        age = self.ensemble.predict(scaled_embedding)[0]

        # Возврат с округлением до 1 знака
        return round(age, 1)

    def get_face_embedding(self, img_path):
        """Получение эмбеддинга лица"""
        img = self.preprocess_image(img_path)
        if img is None:
            return None
```

```
            # Расширение измерений (требования FaceNet, форма фходных данных: (1, 160,
160, 3))
            img_array = np.expand_dims(img, axis=0)
            img_tensor = tf.convert_to_tensor(img_array, dtype=tf.float32)

            # Получение эмбеддинга
            result = self.embedder(img_tensor)
            embedding = result['Bottleneck_BatchNorm'].numpy()
            return embedding.squeeze(0)

        def train_models(self, X, y):
            """Обучение модели"""
            # Масштабирование данных
            X_scaled = self.scaler.fit_transform(X)

            # Обучение SVR с RBF ядром
            svm_rbf = SVR(kernel='rbf', gamma='auto', epsilon=0.05, C=100)
            svm_rbf.fit(X_scaled, y)
            self.individual_models['svm_rbf'] = svm_rbf

            # Обучение SVR с полиномиальным ядром
            svm_poly = SVR(kernel='poly', epsilon=0.1, degree=2, coef0=0.5, C=100)
            svm_poly.fit(X_scaled, y)
            self.individual_models['svm_poly'] = svm_poly

            # Создание ансамбля
            self.ensemble = VotingRegressor(
                estimators=[
                    ('svm_rbf', svm_rbf),
                    ('svm_poly', svm_poly)
                ],
                weights=[2, 1]  # Вес модели с RBF ядром в 2 раза больше
            )
            self.ensemble.fit(X_scaled, y)

            return self

        def evaluate_model(self, X_test, y_test):
            """Оценка модели"""
            if self.ensemble is None:
                print("Модель не обучена")
                return None

            # Масштабирование тестовых данных
            X_test_scaled = self.scaler.transform(X_test)

            # Предсказание на тестовых данных
            y_pred = self.ensemble.predict(X_test_scaled)

            # Вычисление метрик
            metrics = {
                'MSE': mean_squared_error(y_test, y_pred),
```

```
            'RMSE': np.sqrt(mean_squared_error(y_test, y_pred)),
            'R2': r2_score(y_test, y_pred),
            'MAE': np.mean(np.abs((y_test - y_pred)))
        }

        # Сохранение метрик
        self.evaluation_metrics = {
            'ensemble': metrics
        }

        return self.evaluation_metrics

    def print_metrics(self):
        """Вывод метрик оценки"""
        if not self.evaluation_metrics:
            return

        print("\nМетрики оценки модели:")
        for metric, value in self.evaluation_metrics['ensemble'].items():
            print(f"{metric}: {value:.4f}")

    def save_models(self, path):
        """Сохранение модели"""
        if self.ensemble is None:
            return
        joblib.dump({
            'ensemble': self.ensemble,
            'individual_models': self.individual_models,
            'scaler': self.scaler
        }, path, compress=('gzip', 3))

    @classmethod
    def load_models(cls, embedding_model_path, predictor_path):
        """Загрузка модели"""
        # Создание экземпляра класса
        predictor = cls(embedding_model_path)

        # Загрузка сохраненной модели
        models = joblib.load(predictor_path)
        predictor.ensemble = models['ensemble']
        predictor.individual_models = models['individual_models']
        predictor.scaler = models['scaler']

        return predictor
```