

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«Санкт-Петербургский политехнический университет Петра Великого»
Институт компьютерных наук и кибербезопасности
Высшая школа технологий искусственного интеллекта

КУРСОВАЯ РАБОТА
УПРАВЛЕНИЕ МЕТАДАННЫМИ РЕЛЯЦИОННОЙ СИСТЕМЫ
УПРАВЛЕНИЯ БАЗЫ ДАННЫХ POSTGRESQL

Выполнил:
студент гр. 5140201/50301

А.С. Тимофеев

Преподаватель:
доцент ВШТИИ ИКНК

С.Г. Попов

Санкт-Петербург
2025

СОДЕРЖАНИЕ

ПОСТАНОВКА ЗАДАЧИ	4
1 СБОР ДАННЫХ	5
1.1 Публичные датасеты	5
2 ПРЕДОБРАБОТКА ДАННЫХ	7
ЗАКЛЮЧЕНИЕ	10
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	11
ПРИЛОЖЕНИЕ А	12

ПОСТАНОВКА ЗАДАЧИ

В рамках работы необходимо реализовать приложение для распознавания возраста по фотографии используя CNN (convolutional neural network).

Входные данные:

- F - изображение лица человека.

Выходные данные:

- $a \in [0; 115]$ - возраст человека предсказанный по фотографии лица.

Ограничения:

- изображения должны содержать одно лицо;
- размер изображения 224 на 224 пикселя;
- предсказание выполняется только для людей;
- в расчет не берутся этнические и половые признаки.

1 СБОР ДАННЫХ

Для решения задачи распознавания возраста человека по фотографии лица был проведён анализ доступных открытых наборов данных (датасетов), содержащих изображения лиц с аннотированным возрастом.

1.1 Публичные датасеты

В рамках работы используются и рассматриваются следующие открытые датасеты:

- UTKFace;
- APPA-Real Age;

UTKFace

UTKFace - один из наиболее распространённых наборов данных для задач оценки возраста и пола. Он содержит более 20 000 изображений лиц людей в возрасте от 0 до 116 лет, снятых в различных условиях освещения, ракурсах и с разным фоном. [1] Каждое изображение имеет аннотацию, закодированную непосредственно в названии файла в формате:

[age]_[gender]_[race]_[date_time].jpg

- `age` — возраст человека (целое число);
- `gender` — пол (0 — мужчина, 1 — женщина);
- `race` — этническая принадлежность (0–4, пять категорий);
- `date_time` — метка времени.

Изображения имеют размер: 200 на 200 пикселей.

APPA-Real Age

APPA-Real Age - содержит 7591 изображение с указанием реального и предполагаемого возраста. Общее количество предполагаемых голосов составляет около 250 000. В среднем на каждое изображение приходится около

38 голосов, что делает средний предполагаемый возраст очень стабильным (0,3 стандартной ошибки от среднего значения). [2]

Каждое изображение имеет порядковый номер, а реальный возраст указан в отдельном csv файле:

- `image name` — порядковый номер изображения;
- `real_age` — реальный возраст;
- `apparent_age` — воспринимаемый возраст;

Изображения имеют различный размер.

В данной работе будем использовать только метки реального возраста.

Распределение возрастов людей на фотографиях приведено на [Рисунке 1](#).

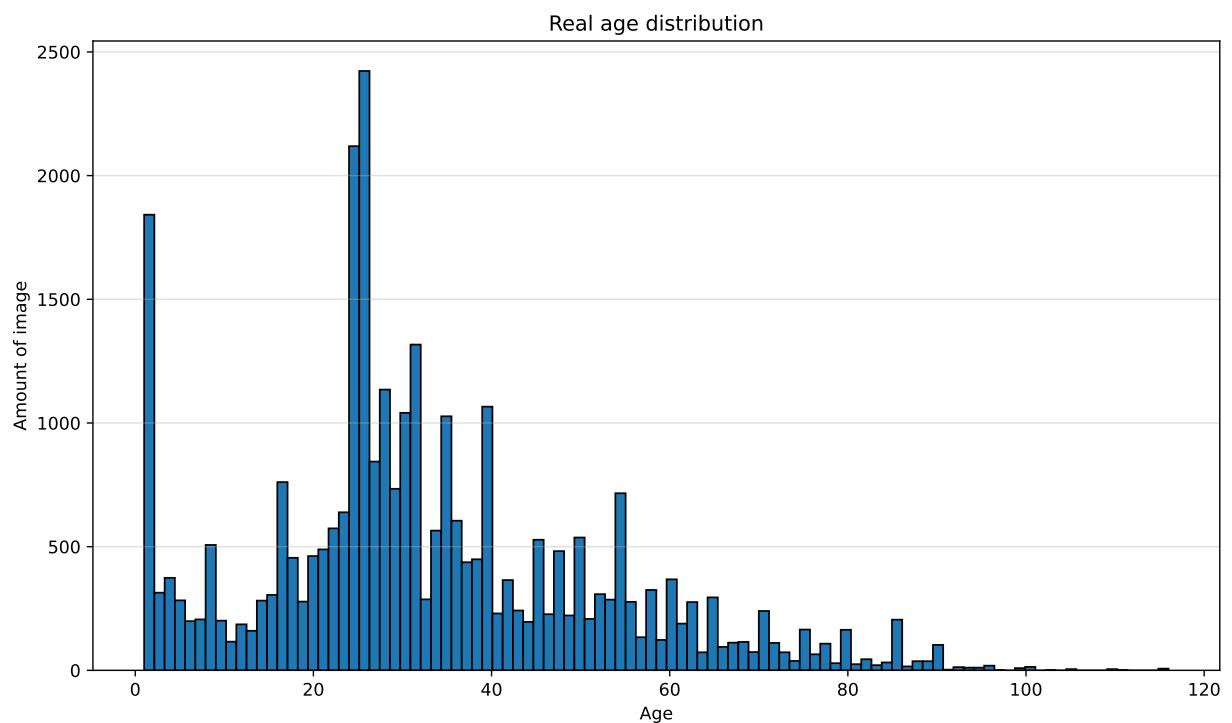


Рис. 1 — Распределения возраста лиц на фотографиях из датасетов APPA, UTKFace

2 ПРЕДОБРАБОТКА ДАННЫХ

Для выбранных датасетов будут выполнены следующие шаги для приведения данных к единому виду:

1. Загрузка данных из открытых источников.
2. Конкатенация датасетов с консистентным именем файла: `image-[number]_[real_age].jpg`
3. Создание csv-файла с метками для изображений.
4. Приведение изображений к единому размеру.
5. Разбиение на тестовую, валидационную и тестовые подвыборки.

Загрузка данных из открытых источников

Для загрузки датасетов был реализован скрипт на языке bash. Исходный код скрипта приведен в [Приложении А](#).

При успешном выполнении скрипта датасет из изображений представлен в предварительном формате:

```
1 ...
2 datasets/
3     UTKFace/          # Изображения метки( внутри названия файла)
4     appa/
5         *.csv          # .csv файлы с метками
6         test/           # Изображения связаны с меткой через название
7             файла
8         train/
9         val/
```

Листинг 2.1 — Иерархия файлов проекта

Конкатенация датасетов

Для конкатенации датасетов реализован скрипт на языке Python [3].

Скрипт собирает изображения из скачанных датасетов в единую директорию. Имена файлов имеют консистентные названия:

`image_[number]_[real_age].jpg`.

Создание файла с метками

Для создания csv-файла с метками объектов реализован скрипт на языке Python [4].

В результате csv-файл имеет следующий вид:

```
1 ,    image_name,          age
2 0,   image_23162_25.jpg,  25
3 1,   image_17662_78.jpg,  78
4 2,   image_06600_32.jpg,  32
```

Листинг 2.2 — csv-файл с метками объектов

Нормализация размеров изображений

Для нормализации размеров изображений реализован скрипт на языке Python [5]

Скрипт наивно (обычное сжатие или растягивание) изменяет размер изображения до заданного значения. В данной работе все изображения нормализованы к размеру 224 на 224 пикселя.

На Рисунке 2 приведен пример наивной нормализации изображения до заданного значений.

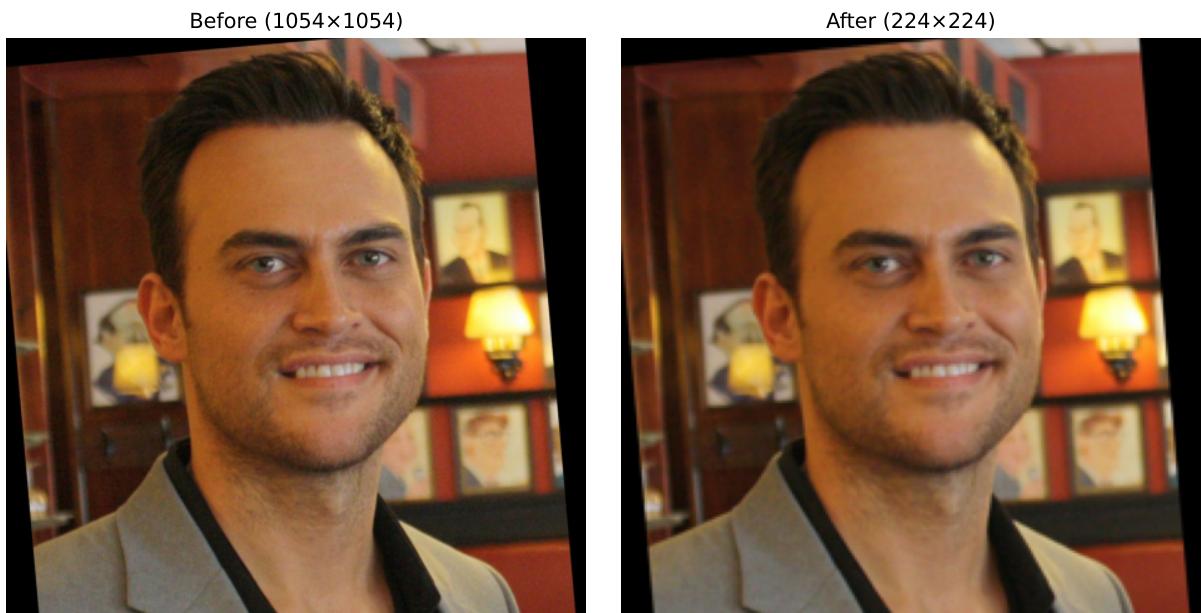


Рис. 2 — Пример сжатия изображения из датасета APPA

Разделение датасета на подвыборки

Для разделения полученного датасета на `test`, `train`, `validation`-подвыборки был реализован скрипт на языке Python [6]

Скрипт делит выборку целиком на K квантилей и равномерно забирает данные для подвыборок в соотношении:

- `train`: 70%;
- `validation`: 20%;
- `test`: 10%;

В результате сбора и предварительной обработки данных был получен датасет размером 31299. Каждое изображение приведено к единому размеру 224 на 224 пикселя. Создан csv-файл с метками для каждой фотографии лица.

ЗАКЛЮЧЕНИЕ

В результате курсовой работы разработано приложение для управления работы с метаданными реляционной системы управления базой данных PostgreSQL и реализован генератор подмножества **SELECT** запросов для заданных баз данных, а также хранение истории сгенерированных запросов.

Для решения данной задачи были выполнены следующие подзадачи:

1. Описание заданного окружения.
2. Создание хранилища метаданных:
 - проектирование схемы базы данных метаданных;
 - составление запросов к `information schema`;
 - реализация сохранения метаданных по DSN-строке подключения к СУБД;
3. Реализация графического интерфейса.
4. Разработка генератора подмножества **SELECT**.
5. Организация хранения истории запросов и возможности их повторного выполнения.

Следует отметить, что одним из недостатков рассматриваемой реализации является хранение истории запросов в базе данных метаданных, что частично нарушает разделения абстракций. Данный подход был выбран для упрощения реализации на этапе разработки.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. *Zhang Z., Song Y., Qi H.* Age Progression/Regression by Conditional Adversarial Autoencoder // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). — 2017. — P. 4352–4360.
2. *Agustsson E., Timofte R., Escalera S., [et al.]*. Apparent and Real Age Estimation in Still Images with Deep Residual Regressors on the APPA-REAL Database // Proceedings of the 12th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG). — 2017. — P. 715–722.
3. *timofeevAS*. `cnn-age-recognize`. — 2025. — GitHub repository. https://github.com/timofeevAS/cnn-age-recognize/blob/main/src/dataset_preprocessing.py.
4. *timofeevAS*. `cnn-age-recognize`. — 2025. — GitHub repository. https://github.com/timofeevAS/cnn-age-recognize/blob/main/src/dataset_csv.py.
5. *timofeevAS*. `cnn-age-recognize`. — 2025. — GitHub repository. https://github.com/timofeevAS/cnn-age-recognize/blob/main/src/dataset_resize_image.py.
6. *timofeevAS*. `cnn-age-recognize`. — 2025. — GitHub repository. https://github.com/timofeevAS/cnn-age-recognize/blob/main/src/dataset_splitter.py.

ПРИЛОЖЕНИЕ А

```
1 #!/bin/bash
2 set -e
3
4 SCRIPT_DIR=$(cd "$(dirname "$0")" && pwd)
5 DATA_DIR="${SCRIPT_DIR}/datasets"
6 UTKFACE_DIR="${DATA_DIR}/UTKFace"
7 ZIP_PATH="${SCRIPT_DIR}/utkface-new.zip"
8 TMP_DIR="${DATA_DIR}/tmp"
9
10 echo "Installing UTKFace dataset..."
11
12 # Dont download archive if exists.
13 if [ ! -f "$ZIP_PATH" ]; then
14     echo "Downloading UTKFace..."
15     curl -L -o "$ZIP_PATH" "https://www.kaggle.com/api/v1/
16         datasets/download/jangedoo/utkface-new"
17 fi
18
19 mkdir -p "$UTKFACE_DIR" "$TMP_DIR"
20 echo "Extracting only utkface_aligned_cropped/UTKFace..."
21 unzip -q -o "$ZIP_PATH" "utkface_aligned_cropped/UTKFace/*.
22     jpg" -d "$TMP_DIR"
23
24 if [ -d "$TMP_DIR/utkface_aligned_cropped/UTKFace" ]; then
25     rsync -a "$TMP_DIR/utkface_aligned_cropped/UTKFace/" "$UTKFACE_DIR/"
26     echo "Moved $(ls "$UTKFACE_DIR" | wc -l) files to
27         $UTKFACE_DIR"
28 else
29     echo "Expected folder not found inside archive"
30     exit 1
31 fi
32
33 echo "Delete temporary folder..."
34 rm -rf "$TMP_DIR"
35
36 echo "Installing UTKFaces successful!"
```

```

35 echo "Installing APPA dataset..."
36
37 ZIP_PATH="${SCRIPT_DIR}/appa-real-release.zip"
38 if [ ! -f "$ZIP_PATH" ]; then
39 echo "Downloading APPA dataset..."
40 curl -L -o "$ZIP_PATH" "https://data.chalearnlap.cvc.uab.cat/
        AppaRealAge/appa-real-release.zip"
41 fi
42
43 APPA_DIR="${DATA_DIR}/appa"
44
45 echo "Extracting appa dataset appa-real-release/..."
46 unzip -q -o "$ZIP_PATH" \
47 "appa-real-release/gt_avg_train.csv" \
48 "appa-real-release/gt_test.csv" \
49 "appa-real-release/gt_valid.csv" \
50 "appa-real-release/train/*" \
51 "appa-real-release/test/*" \
52 "appa-real-release/valid/*" \
53 -d $APPA_DIR
54
55 if [ -d "$APPA_DIR/appa-real-release" ]; then
56 rsync -a "$APPA_DIR/appa-real-release/" "$APPA_DIR"
57 echo "Moved $(ls \"$APPA_DIR/appa-real-release/\") files to $APPA_DIR"
58 else
59 echo "Expected folder not found inside archive"
60 exit 1
61 fi
62
63 echo "Delete temporary folder..."
64 rm -rf "$APPA_DIR/appa-real-release"
65
66 echo "Filtering appa dataset..."
67 echo "Left only necessary pictures in storage... ($APPA_DIR
        /train)"
68 find "$APPA_DIR/train" -type f ! -name "*_face.jpg" -
        delete
69 echo "Left only necessary pictures in storage... ($APPA_DIR
        /test)"
70 find "$APPA_DIR/test" -type f ! -name "*_face.jpg" -
        delete

```

```
71 echo "只剩下必要的图片在存储... ($APPA_DIR  
/valid)"  
72 find "$APPA_DIR/valid" -type f ! -name "*jpg_face.jpg" -  
    delete  
73  
74 echo "安装APPFace成功!"
```

Листинг 2.3 — Bash-скрипт загрузки и распаковки датасета