МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«ТЮМЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ НАУК
Кафедра программного обеспечения

РЕКОМЕНДОВАНО К ЗАЩИТЕ
В ГЭК И ПРОВЕРЕНО НА ОБЪЕМ
ЗАИМСТВОВАНИЯ
Заведующий кафедрой
к.т.н.,доцент
М. С. Воробьева
24.06. 2019 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
(бакалаврская работа)

РАЗРАБОТКА СИСТЕМЫ РАСПОЗНАВАНИЯ ДЕМОГРАФИЧЕСКИХ ХАРАКТЕРИСТИК ЧЕЛОВЕКА В ВИДЕОПОТОКЕ С РЕАЛИЗАЦИЕЙ ВЫЧИСЛЕНИЙ НА CPU

02.03.03. Математическое обеспечение и администрирование информационных систем

Выполнил работу
Студент 4 курса
очной формы обучения
Устелемов
Максим
Алексеевич

Руководитель работы
к.т.н., доцент
Воробьева
Марина
Сергеевна

г. Тюмень, 2019

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Демографические характеристики человека играют важную роль во многих сфера жизни общества: сфере рекламы, охраны права, развлечения, обслуживания и др.

Автоматизированное распознавание таких характеристик человека по изображению лица может быть использовано, например:

- В таргетированном маркетинге, где релевантная для аудитории реклама или информация может транслироваться с цифровых рекламных щитов;
- При сборе данных для дальнейшего анализа: например, при разработке маркетингового плана или проведении таргетированных мероприятий;
- Для контентного поиска, где наличие таких характеристик существенно упрощает пространство поиска;
- Для контроля безопасности: автоматическая система распознавания может помочь предотвратить продажу запрещенной продукции несовершеннолетним;
- При взаимодействии компьютер – человек: для подстройки контекста общения в соответствии с полом и возрастом.

Реализация многих из описанных практических возможностей применения автоматического распознавания требует непрерывной обработки видеопотока, при этом вычисления должны занимать минимально возможное количество времени, чтобы предоставить запас для дальнейших этапов обработки.

Широкому применению систем автоматизированного распознавания должна способствовать низкая вычислительная сложность наряду с обеспечением высокой точности работы и возможностью корректной работы на встроенных платформах без использования графических ускорителей.

Цель выпускной квалификационной работы: разработать систему распознавания демографических характеристик человека в видеопотоке с реализацией вычислений на CPU.

Задачи, решение которых необходимо для достижения цели:

1. Изучение подходов к детектированию лица человека и распознаванию демографических характеристик по изображению лица;
2. Проведение сравнительного анализа изученных подходов;
3. Проектирование и разработка системы распознавания демографических характеристик человека в видеопотоке;
4. Проведение тестирования разработанной системы.

ГЛАВА 1. КОМПОНЕНТ ДЕТЕКТИРОВАНИЯ ЛИЦА НА ИЗОБРАЖЕНИИ

Детектирование лица является первым компонентом в построении систем анализа лиц: например, систем распознавания лиц, определения характеристик человека, определения эмоций и др.

Детектирование лица является частным случаем задачи детектирования объектов [1]. Входными данными при решении задачи детектирования объектов является список классов детекции и целевое изображение для поиска объектов. Задача состоит в определении локализации объектов на изображении независимо от масштаба этих объектов, положения на изображении, частичного перекрытия и освещения. Выходными данными для каждого класса является набор координат прогнозируемого положения объектов этого класса в системе координат входного изображения.

Задача детектирования лица представляет собой задачу детектирования объектов двух классов: лица и остаточного фона. Результатом отнесения объекта к классу является процентная вероятность принадлежности, вариация порового значения которой от меньшего к большему будет изменять выходные данные от наиболее полных к наиболее точным.

Реализация алгоритмов детектирования лиц в системах потоковой обработки изображений требует высокой скорости вычислений. Для оценки скорости работы используется метрика *FPS* (*Frames per Second*) или кадров в секунду. Стандартная кадровая частота современных камер находится в диапазоне от 25 до 30 *FPS*. [2]. Высокая скорость работы алгоритма детектирования лица в режиме потоковой обработки определяется максимальным приближением значения метрики скорости работы алгоритма к значению кадровой частоты современных камер.

Низкая точность работы алгоритма детектирования лица при реализации системы, где детекция используется, как начальный этап вычислений может привести к увеличению числа некорректных обработок на

последующих этапах, что приведет к увеличению времени работы и уменьшению точности итогового результата вычислений системы.

В соответствии с разрабатываемой системой к компоненту детектирования лица предъявляется требование обеспечения высокой точности результатов детектирования с низкой временной сложностью для возможности гарантирования высоких значений показателя обработки кадров в секунду при работе на встроенных устройствах.

1.1. Обзор современных подходов к решению задачи детектирования лица на изображении

Современные подходы к решению задачи детектирования лица могут быть представлены в виде двух категорий:

1. Методы, основанные на выделении локальных особенностей (признаков) на изображении;
2. Методы, основанные на использовании сверточных нейронных сетей (*CHC*) с различными архитектурами.

Методы, основанные на выделении локальных признаков на изображении

Haar Cascades Classifier (Haar) – метод, использующий признаки Хаара, организованные в каскад классификаторов [3]. Изображение разбивается на небольшие участки (окна) и последовательно проходит этапы классификатора, затем размер изображения последовательно уменьшается, и каждое изображение вновь проходит этапы классификатора. Успешно прошедшие классификатор окна обрабатываются алгоритмом подавления немаксимумов для объединения перекрывающих друг друга участков [4].

Histogram of Oriented Gradients Classifier (HoG) – метод, использующий каскад классификаторов, где признаками выступают распределения (гистограммы) направлений градиентов [5]. Обработка изображения осуществляется схожим с *Haar Cascades Classifier* методом: последовательный проход по окнам изображения с подавлением немаксимумов на финальном этапе.

Методы, основанные на использовании *CHC* с различными арихтектурами

Multi-Task Cascades Convolution Network (MTCNN) – представляет собой многозадачный каскад *CHC*, выходным результатом обработки которого являются:

- Прогнозируемые координаты положения лица на изображении;
- Список прогнозируемых координат ориентиров (ключевых точек) лица.

Процесс обработки изображения в *MTCNN* состоит из трех этапов:

- Определение предположительных окон локализации лица при помощи неглубокой *CHC*;
- Уточнение окон с использованием более глубокой *CHC*;
- Обработка *CHC* с ещё более глубокой архитектурой для уточнения результата и локализации ориентиров лица [6].

Single Shot Multibox Detector (SSD) – подход, основывающийся на архитектурах сетей, используемых для классификации изображений [7]. Последние слои такой архитектуры дополняются сверточными слоями с последовательно уменьшающимся размером свертки для возможности детекции объектов разных размеров. В *SSD* используются шаблоны заранее определенных размеров, с помощью последовательных проходов которых по картам признаков и локализуется объект на изображении. После каждого прохода расположение и точность определения класса для шаблона корректируется с использованием комбинированной функции ошибки, являющейся суммой функции ошибки по точности и по расположению.

Согласно проведенному сравнительному анализу авторами статьи [7] метод *SSD* существенно превосходит ранее разработанные методы с похожими подходами к решению задачи детектирования. С помощью такого подхода удается добиться хорошей точности детектирования наряду с высокой характеристикой *FPS*.

На рисунке 1 приведено сравнение метода *SSD* с ранее известными подходами такими как: *Faster-RCNN*, *Fast-RCNN*, *YOLO* и др. На

вертикальных осях располагается: метрика *mAP* (*mean Average Precesion*), характеризующая точность метода детекции, и метрика *FPS*, характеризующая временную сложность метода детекции. Приводятся результаты тестирования на тестовой выборке Pascal VOC 2007 с использованием Nvidia Titan X для моделей с размером входного слоя 300x300, обученных на датасете Trainval Sets 2007+2012.

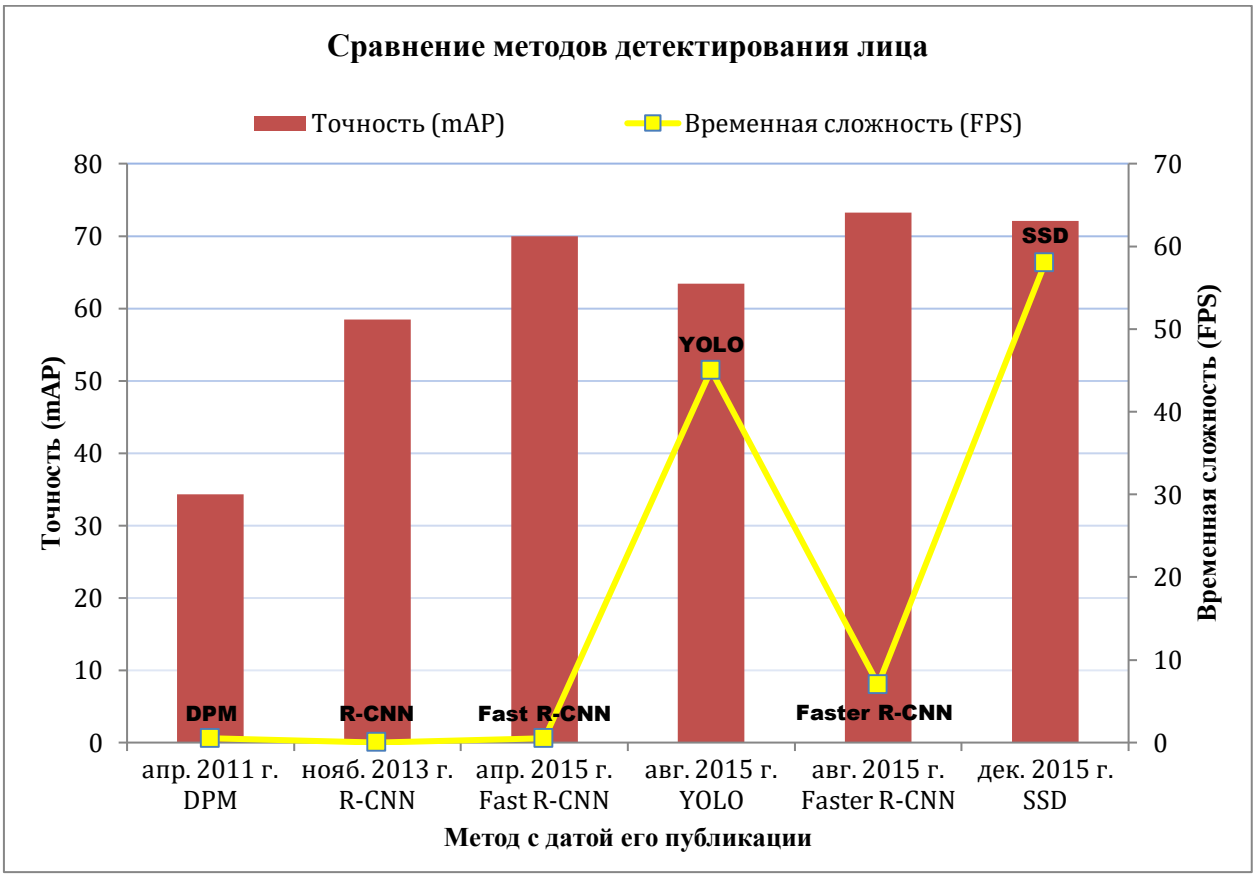


Рисунок 1. Сравнение методов детектирования лица

Faceboxes – подход, основывающийся на *CHC* с двухуровневневой архитектурой: первый уровень – это неглубокая *CHC*, разработанная для быстрого уменьшения пространственного размера входного изображения, что призвано увеличить скорость работы сети; второй уровень – *CHC* с похожим на подход *SSD* использованием шаблонов с корректировкой комбинированной функцией ошибки [8].

1.2. Проведение сравнительного анализа

После формулировки требований, предъявляемых к методу детектирования лиц нужно определить: какой из рассмотренных подходов

наиболее удовлетворяет данным требованиям. Необходимо провести экспериментальное исследование, для чего требуется выполнить следующие этапы:

- Выбрать метрики тестирования, соответствующие сравниваемым моделям и предъявляемым к их работе требованиям;
- Выбрать для проведения тестирования данные, аналогичные тем, которые впоследствии будут использоваться при работе системы;
- Реализовать тестируемые методы для возможности проведения эксперимента;
- Провести сравнительный эксперимент с использованием соответствующего системе оборудования, и на основе анализа полученных в ходе эксперимента результатов, определить модели наиболее подходящие для использования как компоненты разрабатываемой системы.

1.2.1. Метрики тестирования

Основными характеристиками эффективности метода детектирования является точность детекции и временная сложность.

Для оценки точности работы методов детектирования широко используются две метрики: *average IoU* (*avgIoU*) и *mean Average Precesion* (*mAP*) [9].

Intersection over Union (*IoU*) – метрика, характеризующая то, насколько точно прогнозируемые границы объекта совпадают с истинными границами объекта.

Пусть A – множество прогнозируемых принадлежащих объекту пикселей, а B – множество истинно принадлежащих объекту пикселей, тогда:

$$IoU\ (A,B) = \frac{A\ \cap\ B}{A\ \cup\ B} \quad (1)$$

Average IoU (avgIoU) представляет собой усредненное значение показателя по всем прогнозируемым границам.

При реализации тестирования используется определенное пороговое значение *IoU* для отнесения результата детектирования к истинным результатам. Широко распространенные пороговые значения 0.5 и 0.75 [9].

True Positive ($TP(c)$) – истинно позитивный результат детектирования для класса c. Прогнозируемое детектирование было осуществлено для класса c, и объект класса c действительно находится в спрогнозированных границах.

False Positive ($FP(c)$) – ложно позитивный результат детектирования для класса c. Прогнозируемое детектирование было осуществлено для класса c, но объекта класса c нет в спрогнозированных границах.

Average Precesion ($AP(c)$) для класса c вычисляется как:

$$AP(c) = \frac{TP(c)}{TP(c) + FP(c)} \quad (2)$$

Mean Average Precesion (mAP) может быть вычислена как:

$$mAP = \frac{1}{|C|} \sum_{c \in C} \frac{TP(c)}{TP(c) + FP(c)}, \quad (3)$$

где C – множество классов

При оценке методов детектирования лица может рассматриваться только один класс – лицо. При таком подходе метрики *AP* и *mAP* являются эквивалентными.

Для оценки временной сложности методов используется:

- Время обработки одного изображения, выраженное в секундах или миллисекундах;
- Количество обрабатываемых в секунду кадров – *FPS*.

1.2.2. Данные для тестирования

Для тестирования используется распространенный при решении задачи детектирования лиц датасет – WIDER Face, содержащий 32 203 изображений и 393 703 аннотированных лиц с разнообразными вариациями в размерах лиц, положении, освещении, мимики и перекрытия [10]. Такие данные

хорошо подходят для оценки методов детекции. Примеры изображений из базы WIDER Face приведены на рисунке 2.

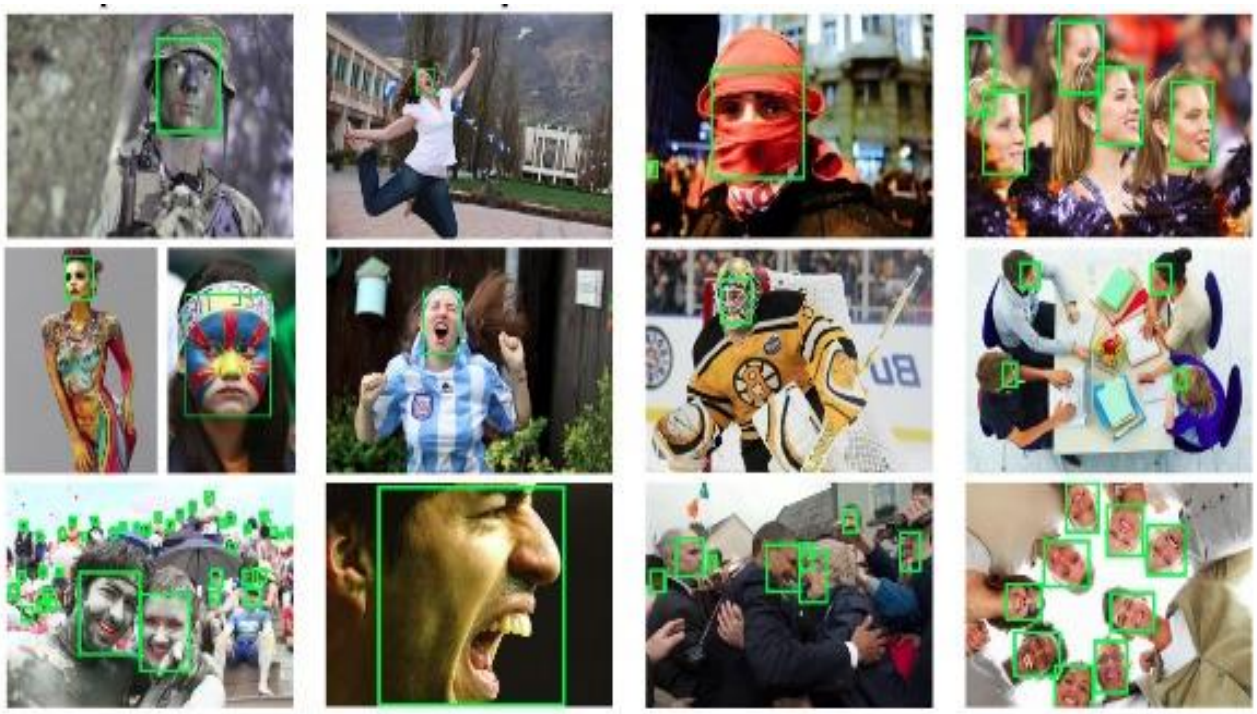


Рисунок 2. Примеры изображений из базы WIDER Face

База изображений разделена разработчиками на три части:

- 40% изображений – тренировочная часть;
- 10% изображений – оценочная часть;
- 50% изображений – тестовая часть.

Для тестирования используется тестовая часть датасета. Детектированные области размерностью меньше чем 15x15 пикселей не представляют информативных данных, поэтому информация о таких детекциях была удалена из используемой части. После чего количество изображений в тестовой части составило 16 106, а количество аннотированных лиц – 98 871.

1.2.3. Реализация методов детектирования

Для проведения вычислительного эксперимента необходима реализация рассматриваемых методов детекции. Реализация подходов *MTCNN*, *Haar* и *HoG* представлена в библиотеках компьютерного зрения

языка *Python*, для методов *SSD* и *Faceboxes* требуется провести обучение *CHC* на соответствующих данных [11].

Метод SSD

Согласно предъявляемым при реализации системы требованиям необходима высокая скорость работы алгоритма детектирования. Авторами интернет-ресурса [12] приводятся результаты тестирования скорости работы различных архитектур *CHC* при реализации метода *SSD*. Наименьшие показатели временной сложности зафиксированы при использовании архитектур *MobileNet* [13] и *MobileNetV2* [14].

Разработчиками данных архитектур введен гиперпараметр $\alpha \in [0; 1]$, называемый в работах – *width multiplayer*. Вариация значений данного параметра от большего к меньшему ведет к уменьшению числа выходных параметров на каждом сверточном слое, что приводит к увеличению скорости вычислений, но уменьшению точности: изменяя значения параметра, можно регулировать взаимоотношение между метриками точности вычислений и их скорости.

Для сети с архитектурой *MobileNetV2* авторы в статье [14] приводят реализацию модификации метода *SSD* – *SSDLite*, заключающуюся в изменении добавочных сверточных слоев *SSD* в соответствии с модификациями, приведенными в *MobileNet* архитектурах.

Рекомендуемые авторами значения параметра α – значения с шагом 0.25: $\alpha \in \{0.25, 0.5, 0.75, 1\}$.

Для проведения тестирования метода *SSD* были выбраны архитектуры, представленные в таблице 1.

Таблица 1. Тестируемые архитектуры с реализацией SSD

Архитектура	Размер входного слоя	Значение параметра α
MobileNet SSD	240x180x3	$\alpha \in \{0.5, 0.75, 1\}$
MobileNetV2 SSD	240x180x3	$\alpha \in \{0.5, 0.75, 1\}$
MobileNetV2 SSDLite	240x180x3	$\alpha = 1$

Размер входного слоя рассматриваемых архитектур был выбран 240x180x3 для соответствия распространенному соотношению сторон кадра – 4:3, используемому современными цифровыми камерами [15]. Глубина 3 используется для обработки трех каналов цвета входного изображения.

Выбранные архитектуры с реализацией *SSD* метода были обучены с использованием конфигурационных параметров, представленных в таблице 2.

Таблица 2. Параметры обучения моделей с реализацией метода SSD

Параметр	Значение
Данные для обучения	Тренировочная часть датасета WIDER Face: 12 880 изображений
Данные для оценки	Оценочная часть датасета WIDER Face: 3 220 изображений
Функция ошибки	Softmax Loss + L1 Loss
Метод обучения	SGD с Momentum 0.9
Количество итераций	180 тыс.
Learning Rate	10^{-3} до 60 тыс. итераций; 10^{-4} до 100 тыс. итераций; 10^{-5} до 140 тыс. итераций; 10^{-6} до 180 тыс. итераций
Batch Size	16
Платформа обучения	Google Collaboratory
Библиотеки для реализации процесса обучения	Tensorflow 1.13 Tensorflow Object Detection API 2.0
GPU	Nvidia Tesla T4, 14 Gb
Время обучения	От 12 до 15 часов в зависимости от параметра α

Входе процесса обучения рассматриваемые модели оценивались на соответствующих данных по метрикам точности детектирования лиц: *mAP* с *IoU* = 0.5 и *mAP* с *IoU* = 0.75. Результаты оценки моделей в конце обучения приведены в таблице 3.

Таблица 3. Сравнение обученных моделей с реализацией метода SSD

Наименование модели	mAP с IoU 0.5	mAP с IoU 0.75
SSD MobileNet α = 1	0.7229	0.4342
SSD MobileNet α = 0.75	0.7113	0.4413
SSD MobileNet α = 0.5	0.6165	0.3251
SSD MobileNetV2 α = 1	0.7267	0.4585
SSD MobileNetV2 α = 0.75	0.6982	0.4272
SSD MobileNetV2 α = 0.5	0.6678	0.4021
SSDLite MobileNetV2 α = 1	0.7402	0.4611

Согласно таблице 3 наилучший результат по двум измерениям *mAP* показывает модель *SSDLite MobileNetV2* $\alpha = 1$. Точность моделей уменьшается с уменьшением параметра α. Дальнейшее проведение сравнения по временной сложности покажет, какая из моделей лучшим образом отвечает требованиям по точности и скорости, обеспечивая наилучший баланс между этими двумя метриками.

Модель Faceboxes

Для проведения эксперимента использовалась обученная *CHC* с архитектурой *Faceboxes* и размером входного слоя 1024x1024x3. Параметры обучения модели представлены в таблице 4.

Также для проведения сравнительного эксперимента были использованы подходы со следующей реализацией:

- *MTCNN* – предобученный каскад *CHC*, доступный для использования в библиотечной реализации [16].
- *Haar* – реализация в библиотеке *OpenCV* [17].

- *HoG* – реализация в библиотеке *dlib* [18].
- *Dlib CNN* – предобученная *CHC*, доступная в библиотеке *dlib* [18].

Таблица 4. Параметры обучения модели *Faceboxes*

Параметр	Значение
Данные для обучения	Тренировочная часть датасета WIDER Face: 12 880 изображений
Данные для оценки	Оценочная часть датасета WIDER Face: 3 220 изображений
Функция ошибки	Softmax Loss + L1 Loss
Метод обучения	SGD с Momentum 0.9
Количество итераций	140 тыс.
Learning Rate	10^{-3} до 80 тыс. итераций; 10^{-4} до 100 тыс. итераций 10^{-5} до 120 тыс. итераций
Batch Size	16

1.2.4. Проведение эксперимента

В соответствии с предъявляемыми к разрабатываемой системе требованиями тестирование моделей детектирования лиц на изображении проводилось на CPU.

Эксперимент проводился для изображений трех размеров: 240x180, 640x480, 1024x768. 240x180 – это размерность входного слоя обученных моделей *SSD*, и для сравнения этих моделей между собой по метрикам скорости и точности достаточно использования данного разрешения. Однако другие модели показывают низкую точность при работе с таким размером изображения, так как были разработаны для обработки изображений более высоких разрешений. Поэтому для получения действительных результатов используется три размера изображений, при этом размеры соответствуют соотношению сторон 4:3 по описанным выше причинам.

Характеристики оборудования для проведения тестирования:

- CPU: Intel(R) Core(TM) i3-4030U CPU @ 1,90GHz;
- Операционная система: Ubuntu, 18.04.

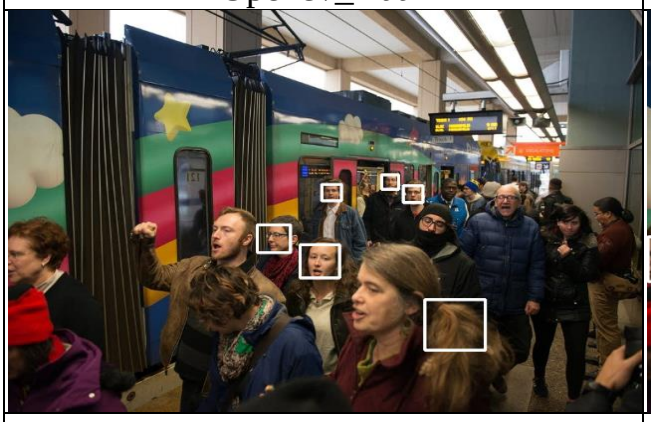
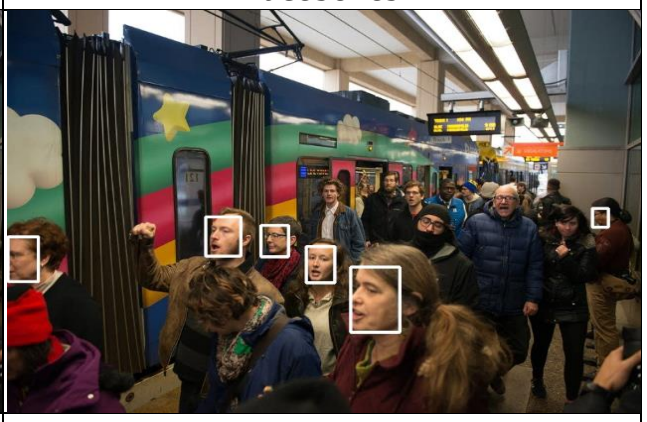
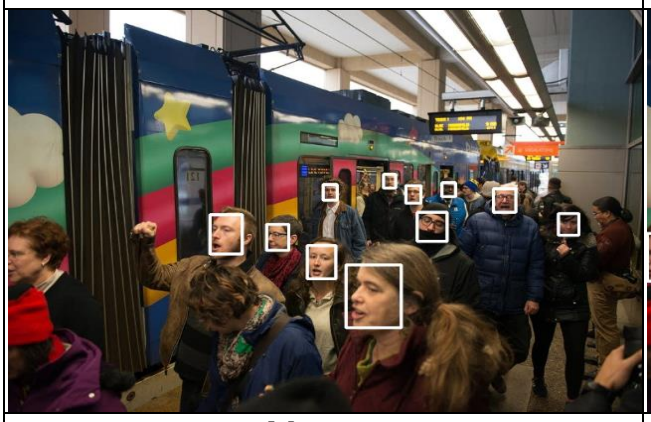
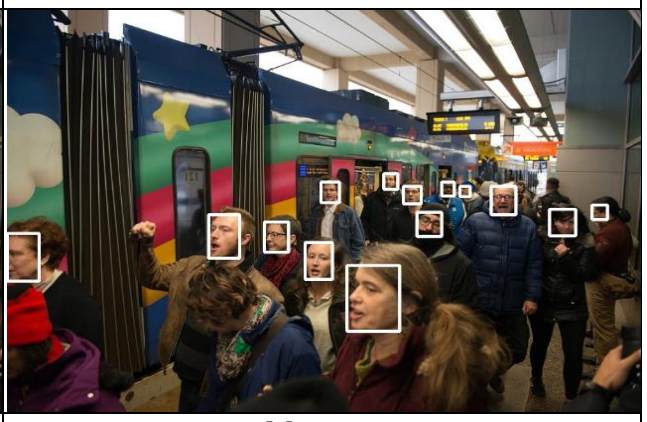
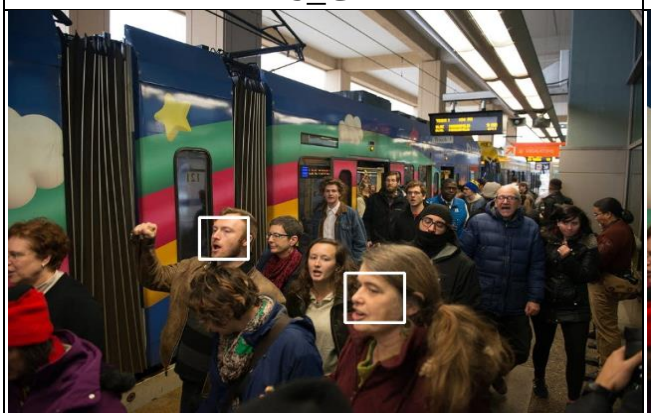
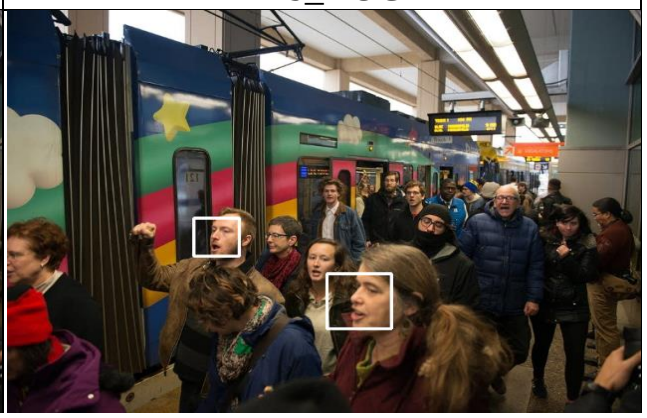
Результаты по временной сложности моделей представляют собой усредненные значения по 1000 запускам.

На рисунке 3 изображен сравнительный график по показателям точности и временной сложности, полученным при проведении тестирования моделей детектирования лиц. На графике приводятся наилучшие показатели работы методов: значения в миллисекундах по временной сложности для разрешения 240x180, значения точности по метрикам *mAP IoU 0.5* и *avgIoU* для разрешения 1024x768.

Рисунок 3. Результаты тестирования моделей детектирования лиц

Результаты проведенного тестирования показывают, что обученные модели *SSD* имеют наилучшие показатели по временной сложности и точности в сравнении с другими подходами. В таблице 5 приведены примеры обработки рассматриваемыми моделями изображения с лицами разного размера и положения.

Таблица 5. Пример обработки изображения моделями детектирования лиц

Opencv_Haar	Faceboxes
	
MTCNN	SSD MobileNet α = 0.75
	
Dlib_CNN	Dlib_HOG
	

Характеристики моделей *SSD* 240x180, имеющих наилучшие значения баланса между двумя метриками приведены в таблице 6.

Таблица 6. Характеристики моделей SSD 240x180

Модель	mAP	avgIoU	FPS	мс
SSD MobileNet α = 0.75	0.72	0.54	25	39
SSD MobileNet α = 0.5	0.67	0.5	37	27
SSD MobileNetV2 α = 0.5	0.7	0.52	31	32

Благодаря высокой точности детектирования лица рассматриваемыми моделями *SSD* и выбору данных для обучения детектирование осуществляется в широком диапазоне углов поворота головы:

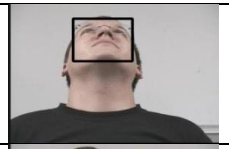
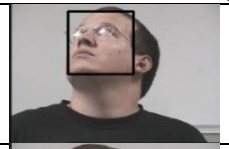
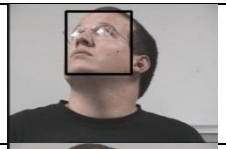
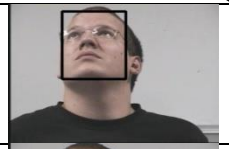
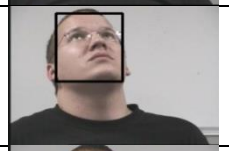
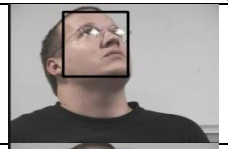
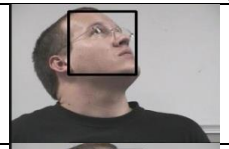
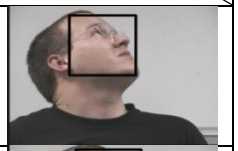
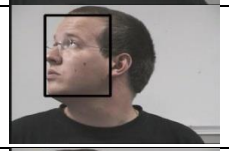
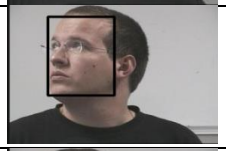
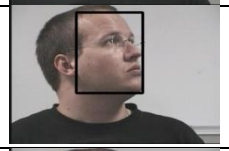
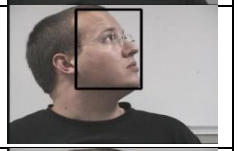
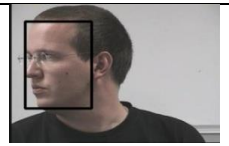
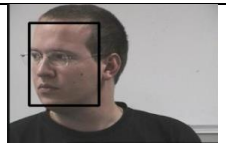
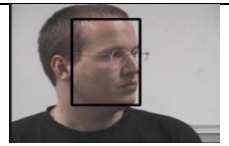
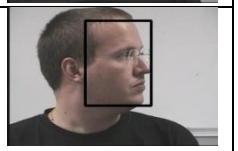
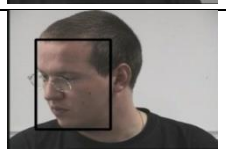
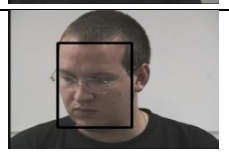
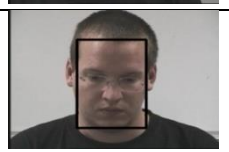
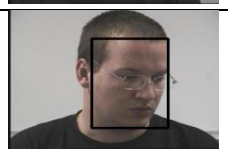
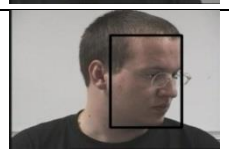
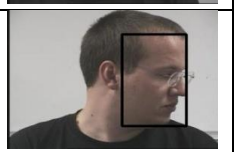
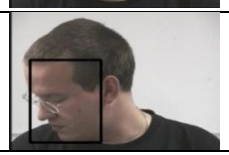
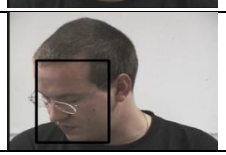
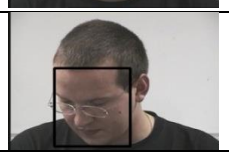
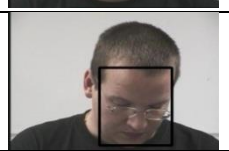
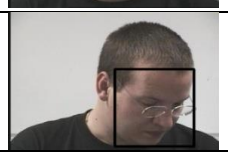
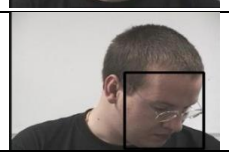
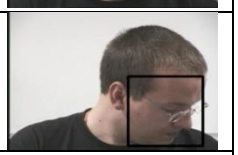
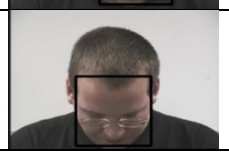
$$\begin{cases} -90^\circ \leq \textit{угол рыскания} \leq +90^\circ \\ -60^\circ \leq \textit{угол тангажа} \leq +60^\circ \end{cases}$$

В таблице 7 приведены результаты детектирования лиц методом *SSD MobileNet* $\alpha = 0.75$ при различных углах поворота головы для изображений из датасета [19].

Рассматриваемые модели на основе метода *SSD* устойчивы к наличию очков (согласно изображениям, представленным в таблице 7). Модели также устойчивы и к различным выражениям лица: на рисунке 4 приведены результаты детектирования модели *SSD MobileNet* $\alpha = 0.75$ для различных выражений лица [20].

Рисунок 4. Результаты детектирования лиц с различными выражениями

Таблица 7. Оценка степени инвариантности метода детектирования лица к повороту головы

V \ H	+90°	+60°	+30°	0°	−30°	−60°	−90°
+90°							
+60°							
+30°							
0°							
−30°							
−60°							
−90°							

H – горизонтальная ось (угол рыскания), V – вертикальная ось (угол тангажа)

ГЛАВА 2. КОМПОНЕНТ РАСПОЗНАВАНИЯ ДЕМОГРАФИЧЕСКИХ ХАРАКТЕРИСТИК ПО ИЗОБРАЖЕНИЮ ЛИЦА

Задача распознавания возраста может быть поставлена как:

- Задача регрессии: при рассмотрении возраста как непрерывную переменную;
- Задача мультиклассовой классификации: возраст определяется как дискретное значение.

Задача распознавания пола формулируется как задача бинарной классификации, где классы представляют собой соответственно мужской и женский пол.

С появлением *CHC* и больших наборов данных с размеченными атрибутами пола и возраста удалось существенно увеличить точность распознавания по сравнению с ранними подходами, основанными на ручном выделении признаков. Однако, большинство современных подходов, основанных на *CHC*, представляют собой глубокие *CHC* с большим числом параметров или ансамбль таких сетей, которые не адаптированы к потоковой обработке кадров, особенно на устройствах с ограниченными вычислительными ресурсами.

При построении системы определения характеристик человека по изображению лица, компоненты распознавания пола и возраста являются следующими за компонентом распознавания лица, результат работы которого и используется в качестве входных данных для методов распознавания характеристик. При потоковой обработке и требованию к высокой полноте работы системы значительная часть вычислительного времени используется на детектирование лица. При необходимости потокового распознавания методы определения характеристик должны обладать низкой вычислительной сложностью для обработки максимального числа кадров из потока, обеспечивая высокую точность работы и запас вычислительного времени для последующих этапов обработки.

К методам определения пола и возраста как к компонентам системы с реализацией вычислений на CPU предъявляется требование обеспечения максимальной точности распознавания наряду с максимально низкой временной сложностью.

2.1. Выбор подхода к решению задачи распознавания

При разработке системы распознавания демографических характеристик пол и возраст человека рассматриваются как независимые атрибуты, и для распознавания каждого из них требуется реализация отдельного модуля.

Необходимо рассмотреть доступные подходы к решению задачи и провести тестирование моделей, для выбора метода, максимально отвечающего поставленным для компонента разрабатываемой системы требованиям.

2.1.1. Распознавание возраста

Для решения задачи определения возраста человека по изображению лица необходимо выбрать архитектуру *CHC*, способную обеспечить хорошую скорость вычислений для выполнения потоковой обработки.

Вариантами, отвечающими поставленным требованиям, являются компактные архитектуры *MobileNet* и *MobileNetV2*, разработанные для обработки на устройствах с ограниченными вычислительными ресурсами. При помощи изменения гиперпараметра α возможно регулирование количества параметров в *CHC* для балансировки точности вычислений и их скорости.

При разработке архитектуры *CHC* для распознавания возраста задача определения возраста рассматривается как задача регрессии. Требуется внести изменения в стандартные архитектуры для их адаптации к решаемой задаче: *MobileNet* и *MobileNetV2* используются как базовая часть архитектуры [21]. Верхние слои архитектур удаляются и дополняются слоями, описание которых приведено в таблице 8.

Таблица 8. Описание верхних слоев моделей распознавания возраста

Слой	Характеристика
Conv2d 1x1	Сверточный слой с ядром 1x1, используемый для сокращения числа фильтров перед полносвязным слоем, что приводит к уменьшению числа параметров сети
Flatten	Слой выравнивания, осуществляющий переход от двумерного представления к одномерному
Dropout с параметром 0.2	Слой регуляризации, предназначенный для предотвращения переобучения. Параметр 0.2 означает, что в процессе обучения 20% случайно выбранных нейронов будут исключены из дальнейших вычислений [22]
Dense с размером 32	Полносвязный слой
Dense с размером 1	Выходной полносвязный слой

Схема архитектуры для моделей распознавания возраста, основанных на подходах *MobileNet* и *MobileNetV2* приведена на рисунке 5.

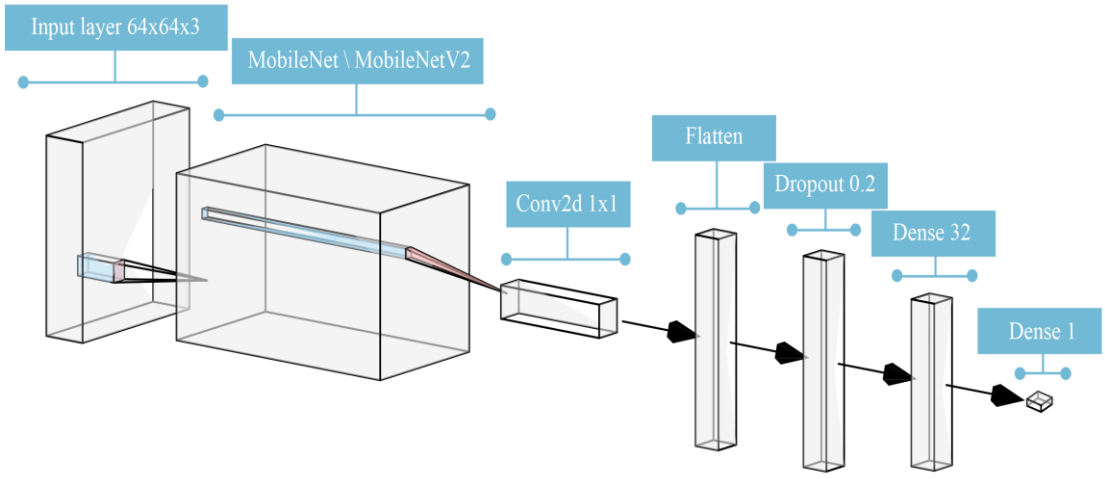


Рисунок 5. Архитектура СНС для распознавания возраста

Для входного слоя была выбрана размерность 64x64x3 по следующим причинам:

- Соотношение 1:1 соответствует соотношению контуров лиц, получаемых на этапе детектирования;
- Малый размер в 64 пикселя способствует уменьшению временной сложности метода;

- Глубина 3 используется для обработки трех каналов цвета входного изображения.

Для проведения сравнительного анализа описанной модели были выбраны базовые архитектуры, представленные в таблице 9.

Таблица 9. Базовые архитектуры для модели распознавания возраста

Базовая архитектура	Входной слой	Значение параметра α
MobileNet	64x64x3	$\alpha \in \{0.25, 0.5, 0.75, 1\}$
MobileNetV2	64x64x3	$\alpha \in \{0.25, 0.5, 0.75, 1\}$

Представленная в работе [23] архитектура *CHC SSR-Net*, разработанная именно для решения задачи распознавания возраста на устройствах с ограниченными вычислительным ресурсами, также была выбрана для проведения тестирования. Размерность входного слоя *SSR-Net*, как и для рассмотренных выше архитектур – 64x64x3.

2.1.2. Распознавание пола

Задача распознавания пола может быть переформулирована из задачи бинарной классификации в задачу регрессии, где классы представляются значениями 0 и 1, а результатом работы *CHC* является действительное число $p \in [0; 1]$, характеризующее доверительную вероятность отнесения к классу.

Для проведения дальнейшего тестирования, как базовые рассматриваются архитектуры *MobileNet* и *MobileNetV2*.

Итоговая схема архитектур распознавания пола, основанных на моделях *MobileNet* и *MobileNetV2* приведена на рисунке 6.

Для проведения сравнительного анализа описанной модели в качестве базовых были выбраны архитектуры, сходные с базовыми архитектурами моделей распознавания возраста (см. таблицу 9).

Также при тестировании рассматривается архитектура *SSR-Net* с размером входного слоя – 64x64x3.

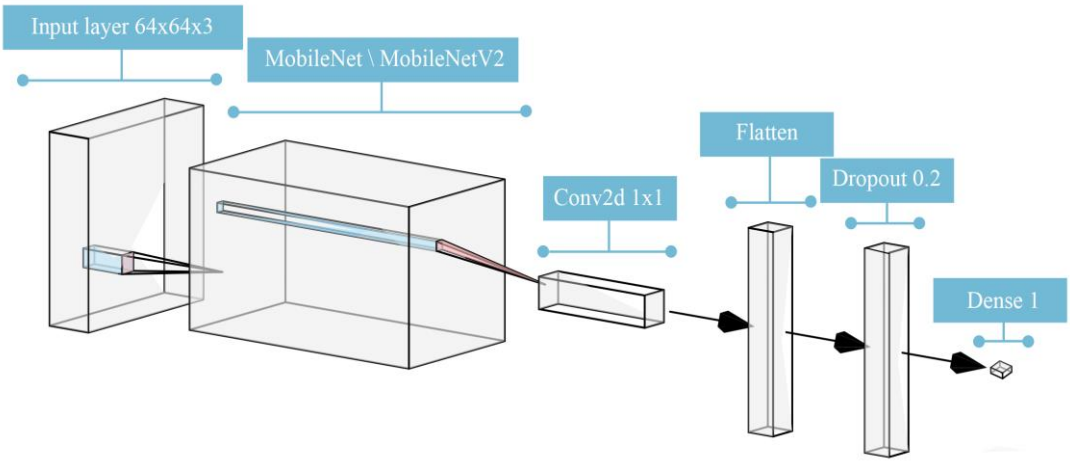


Рисунок 6. Архитектура СНС для распознавания пола

2.2. Данные для обучения

Выбор данных для обучения *CHC* является важным подготовительным этапом перед этапом обучения: требуется подобрать данные, отвечающие требованиям обучаемой модели, обеспечив их корректность и целостность.

2.2.1. Обзор используемых данных

Датасет IMDB-WIKI

IMDB-WIKI – датасет, содержащий 523 051 изображений лиц 20 284 звезд: 460 723 были получены из базы *IMDB*, а оставшиеся 62 238 из базы *Wikipedia*. Изображения содержат аннотацию по полу и возрасту [24].

Датасет был аннотирован в полуавтоматическом режиме и содержит неточности в характеристиках: например, часть изображений не содержит людей или возраст человека на фотографии равен отрицательному значению; такие неточности должны быть устранены перед началом процесса обучения. Поэтому для датасета была применена предобработка:

- Удалены фотографии, аннотированный возраст которых выходит за границы $[0; 100]$ лет;
- Удалены фотографии, не содержащие лица в аннотации;
- Удалены фотографии, не содержащие пола в аннотации;
- Удалены фотографии, содержащие более одного лица на изображении;

- Удалены фотографии со значением face_score (пороговое значение нахождения лица на фотографии) меньшим единицы.

После предобработки *WIKI* часть содержит 37 665 изображений. Распределение возраста и пола в данной части приведено на рисунках 7 и 8 соответственно.

Рисунок 7. Распределение возраста в базе WIKI

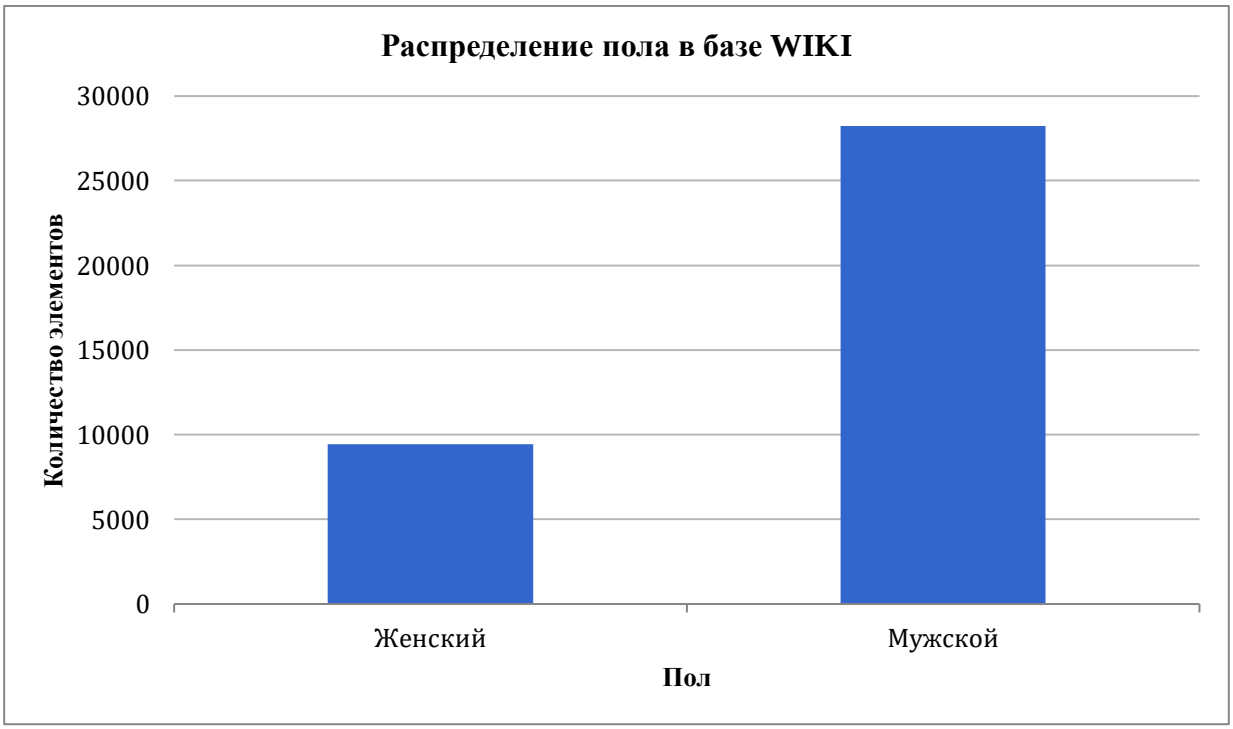


Рисунок 8. Распределение пола в базе WIKI

IMDB часть после предобработки содержит 167 300 элементов. Распределение по возрасту и полу приведено на рисунках 9 и 10 соответственно.

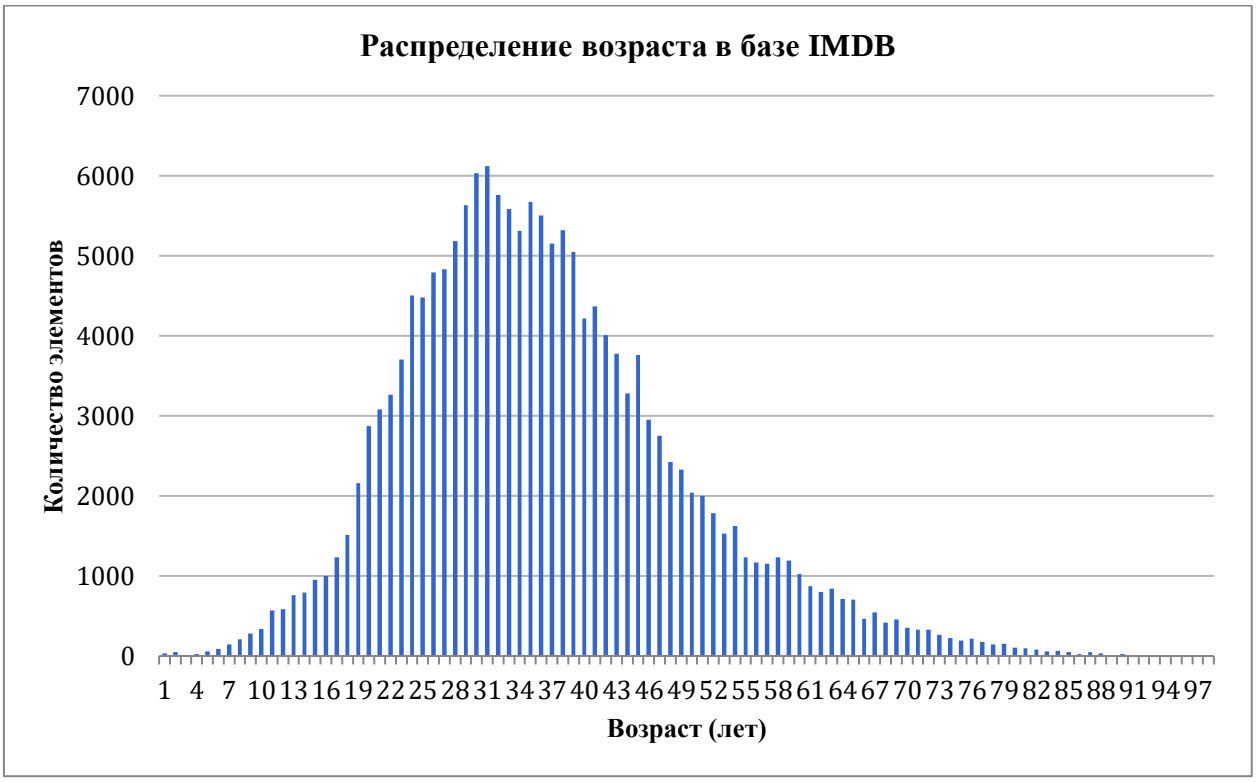


Рисунок 9. Распределение возраста в базе IMDB

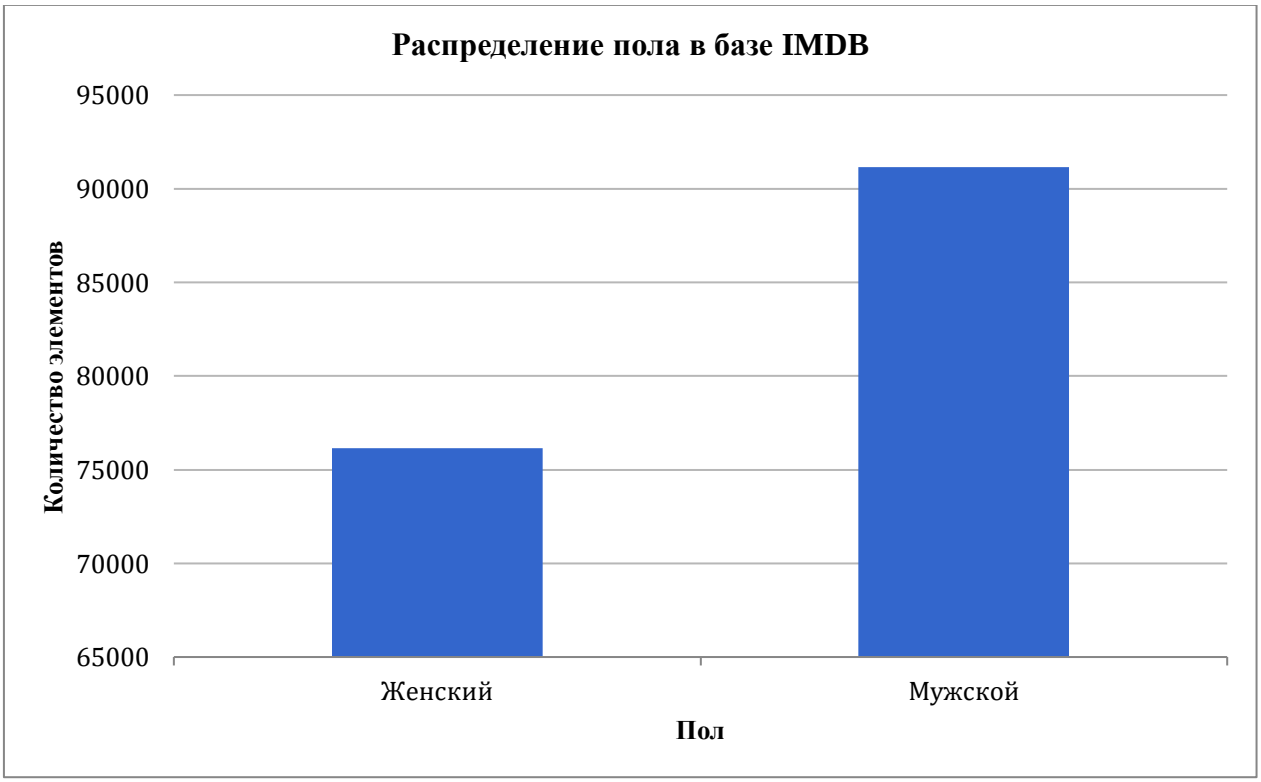


Рисунок 10. Распределение пола в базе IMDB

IMDB-WIKI является самым крупным из представленных в свободном доступе датасетов для возможного использования при решении задачи распознавания пола и возраста. Однако из-за механизма сбора данных он обладает погрешностями в возрасте, поэтому не подходит для проведения сравнительного анализа методов по точности распознавания и в большинстве случаев используется как предтренировочная часть перед обучением на датасете с более точной аннотацией.

Датасет MORPH2

MORPH2 – датасет, в полном виде содержащий 55 134 изображений 13 000 человек в возрасте от 16 до 77 лет с точной аннотацией пола и возраста [25]. Благодаря точной аннотации датасет может быть использован при проведении сравнительного анализа методов распознования возраста.

В открытом доступе представлена часть датасета, содержащая 38 499 изображений. Распределение по возрасту и полу этой части датасета приведено на рисунках 11 и 12 соответственно.

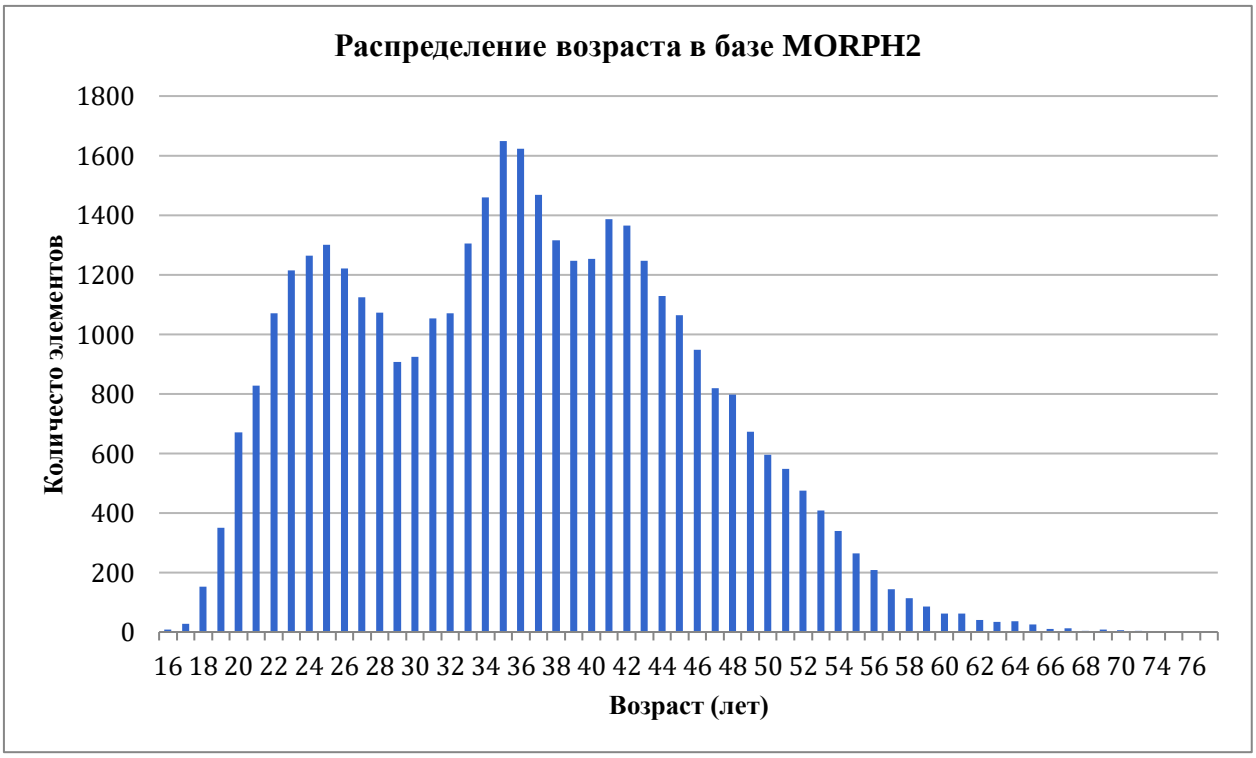


Рисунок 11. Распределение возраста в базе MORPH2

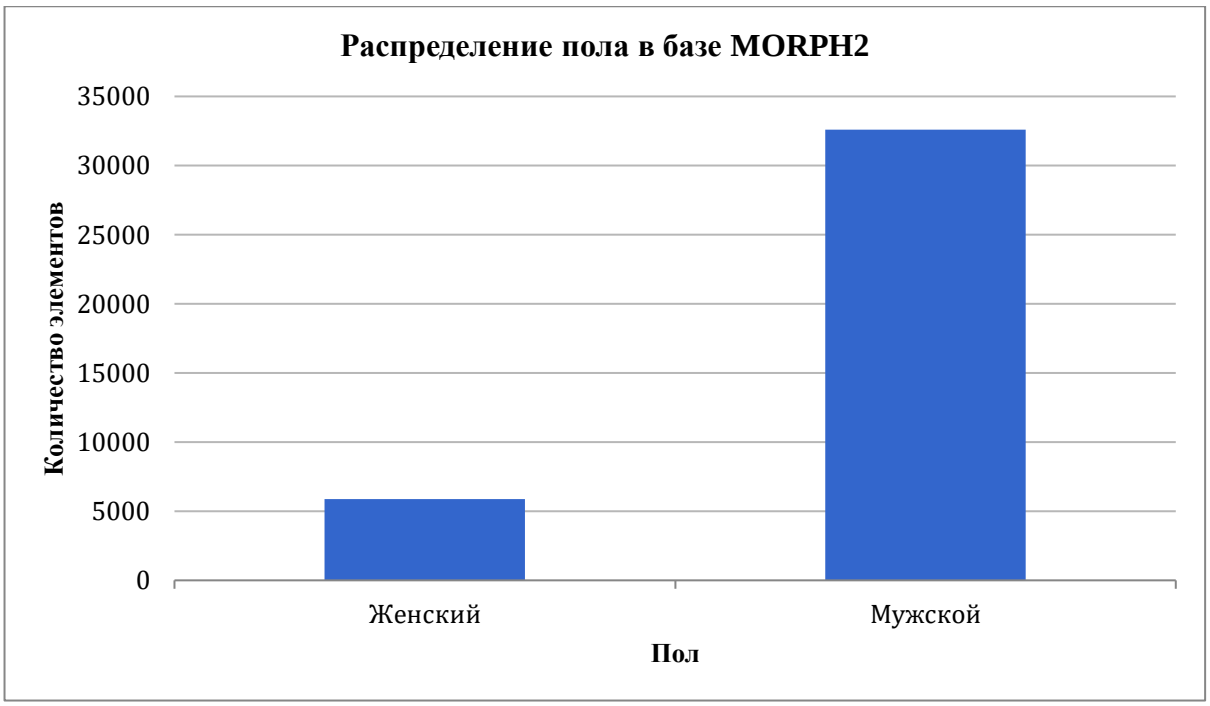


Рисунок 12. Распределение пола в базе MORPH2

2.2.2. Алгоритм выравнивания лиц

Детектированые на изображении лица зачастую не выровнены, а имеют некоторый наклон или поворот головы. Для увеличения точности обучения и дальнейшего использования моделей на других данных необходимо в обоих случаях обработки выравнивать лицо при помощи одного определенного метода так, чтобы ключевые точки лица для каждого изображения находились в одинаковых позициях [26].

К выровненному лицу предъявляются следующие требования:

1. Лицо должно быть центрировано на изображении;
2. Лицо должно быть повернуто так, чтобы глаза располагались на одной горизонтальной оси;
3. Лицо должно иметь приближенно идентичный к другим лицам размер.

Ключевые точки лица могут быть найдены с использованием предтренированного каскада регрессионных деревьев [27] библиотеки dlib [28]. Для выравнивания требуется осуществление аффинного преобразования исходного изображения в соответствии с разработанным методом.

2.3. Проведение сравнительного анализа

2.3.1. Обучение моделей

В связи с большим размером датасета *IMDB-WIKI* (более 7 Гб.) процесс обучения был разделен на три этапа:

- Обучение на наборе данных *IMDB* с параметром *batch size* 128;
- Обучение на наборе данных *WIKI* с параметром *batch size* 64;
- Обучение на наборе данных *MORPH2* с параметром *batch size* 64.

Данные каждого набора случайным образом разбиваются на две части: 80% и 20% изображений, используемые для тренировки и оценки соответственно. Остальные параметры обучения моделей для определения пола и возраста по изображению лица представлены в таблице 10.

Таблица 10. Параметры обучения моделей распознавания пола и возраста

Параметр	Характеристика
Функция ошибки	MAE
Метод обучения	Adam
Число эпох	90
Learning Rate	Начальный: $2 * 10^{-3}$ Множитель на каждой 30 эпохе: 10^{-1}
Платформа обучения	Google Collaboratory
Библиотеки для реализации процесса обучения	Keras 2.0.2
GPU	Nvidia Tesla K80, 12 Gb
Время обучения	В промежутке от 3 до 5 часов в зависимости от модели.

2.3.2. Проведение эксперимента

Характеристики оборудования для проведения тестирования:

- CPU: Intel(R) Core(TM) i3-4030U CPU @ 1,90GHz;
- Операционная система: Ubuntu, 18.04.

Метрики тестирования

Mean Absolute Error (*MAE*) – средняя квадратичная ошибка, используемая при сравнении моделей распознавания возраста [29].

Пусть дано множество изображений лиц $X = \{x_n \mid n = 1..N\}$ и множество истинных значений возраста $Y = \{y_n \mid n = 1..N\}$ для изображений X, и в результате работы метода распознавания возраста получено множество прогнозируемых значений возраста $\hat{Y} = \{\widehat{y_n} \mid n = 1..N\}$. Тогда

$$MAE\ (X) = \frac{1}{N}\sum_{n=1}^{N}|\widehat{y_n} - y_n| \quad (4)$$

Binary Classification Accuracy (BCA) – точность бинарной классификации. Используется при сравнении моделей распознавания пола [29].

Пусть дано множество изображений лиц $X = \{x_n \mid n = 1..N\}$ и множество истинных значений пола $Y = \{y_n \mid n = 1..N\}$, *где* $y_n \in \{0,1\}$. В результате работы метода распознавания пола получено множество прогнозируемых точностей отнесения объекта к классу $\hat{Y} = \{\widehat{y_n} \mid n = 1..N\}$, где $\widehat{y_n} \in [0; 1]$. Тогда

$$BCA\ (X) = \frac{1}{N}\sum_{n=1}^{N}\begin{cases}1, \text{если } y_n = Round(\widehat{y_n}) \\ 0 - \text{иначе}\end{cases}, \quad (5)$$

где Round() – функция округления числа до ближайшего целого.

Временная сложность, как и в случае оценки моделей детектирования лица (см. параграф 1.2.4), рассматривается в *мс*.

Модели распознавания возраста

На рисунке 13 приведены результаты сравнения обученных моделей для распознавания возраста на описанных данных по метрике точности (*MAE*) и временной сложности (*мс*). Показатели по временной сложности приводятся как усредненные для 1000 запусков.

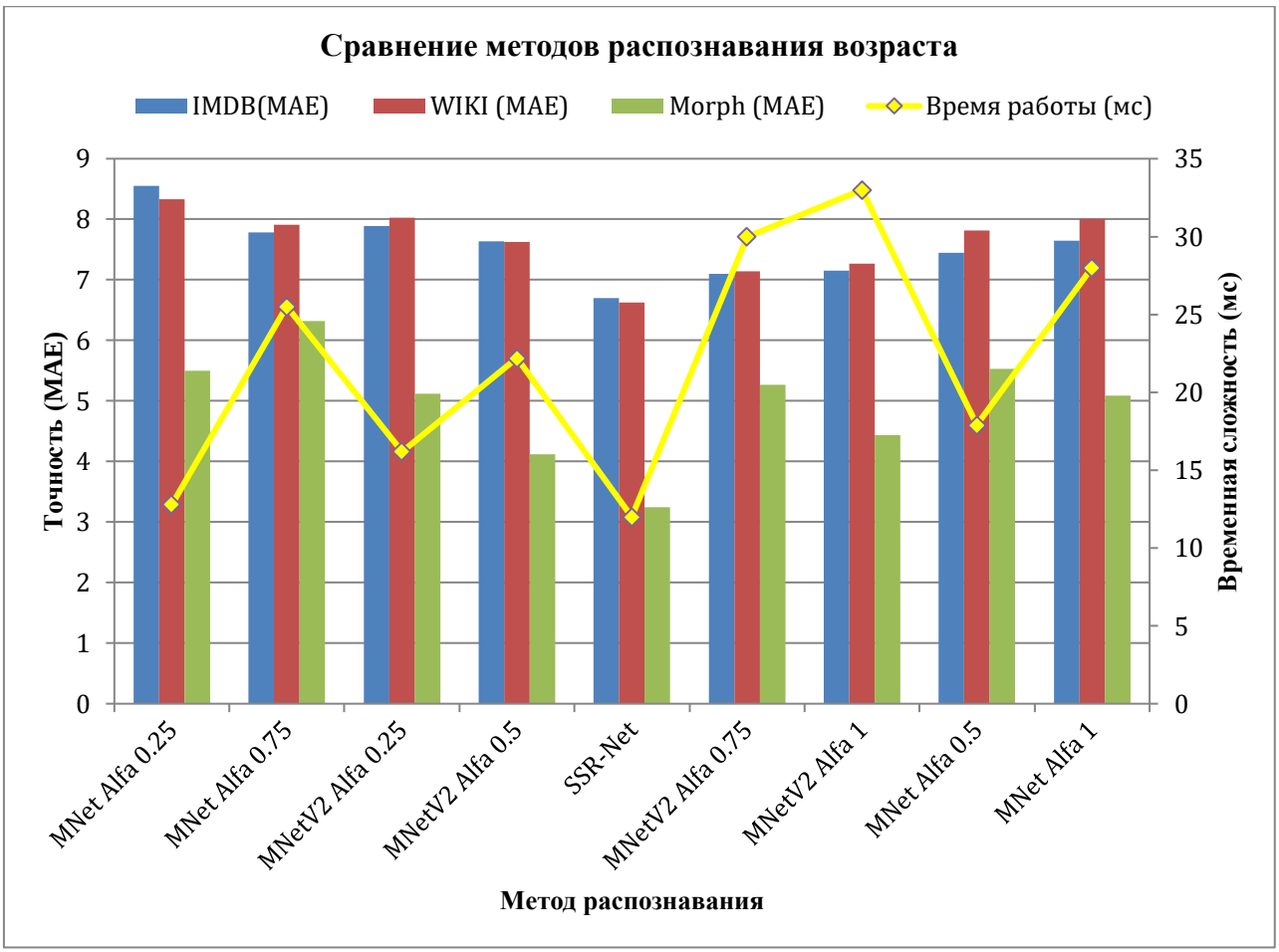


Рисунок 13. Результаты тестирования методов распознавания возраста

Согласно приведенным данным наилучший результат по скорости и точности работы показала архитектура *SSR-Net*:

- *MAE* для датасета *IMDB* – 6.7;
- *MAE* для датасета *WIKI* – 6.63;
- *MAE* для датасета *MORPH2* – 3.24;
- Временная сложность – 12 *мс* или 83 *FPS*.

Разработанный на основе архитектуры *SSR-Net* метод показывает хорошие результаты по точности распознавания в сравнении со значительно более глубокими *CHC*, скорость работы которых на встроенных платформах (на CPU) не достигает даже 1 *FPS*. В таблице 11 приведено сравнение *SSR-Net* метода и наиболее точных подходов, представленных в работах [24, 30,31,32,33]. Сравнение приводится по метрике *MAE* для датасета *MORPH2*. Также приводятся показатели для метода *SSR-Net* с размерностью входного слоя 96x96x3.

Таблица 11. Сравнение метода SSR-Net с другими подходами

Метод	ARN	DEX	Hot	AP	Ranking CNN	SSR-Net	SSR-Net
Размерность входного слоя	224x224x3					64x64x3	96x96x3
MAE MORPH2	3.00	3.25	3.45	2.52	2.96	3.24	3.08
FPS	<1 FPS					83	30

Модели распознавания пола

На Рисунке 14 приведены результаты тестирования методов распознавания пола на представленных данных. Результаты сравнения приводятся по метрике точности – *BCA*, выраженной в процентах, и метрике временной сложности – *мс*.

По представленному графику видно, что наилучшие результаты в сравнении с другими методами показывает архитектура *SSR-Net*:

- *BCA* для датасета *IMDB* – 89.4 %;
- *BCA* для датасета *WIKI* – 93.4 %;
- *BCA* для датасета *MORPH2* – 98 %;
- Временная сложность – 12 *мс* или 83 *FPS*.

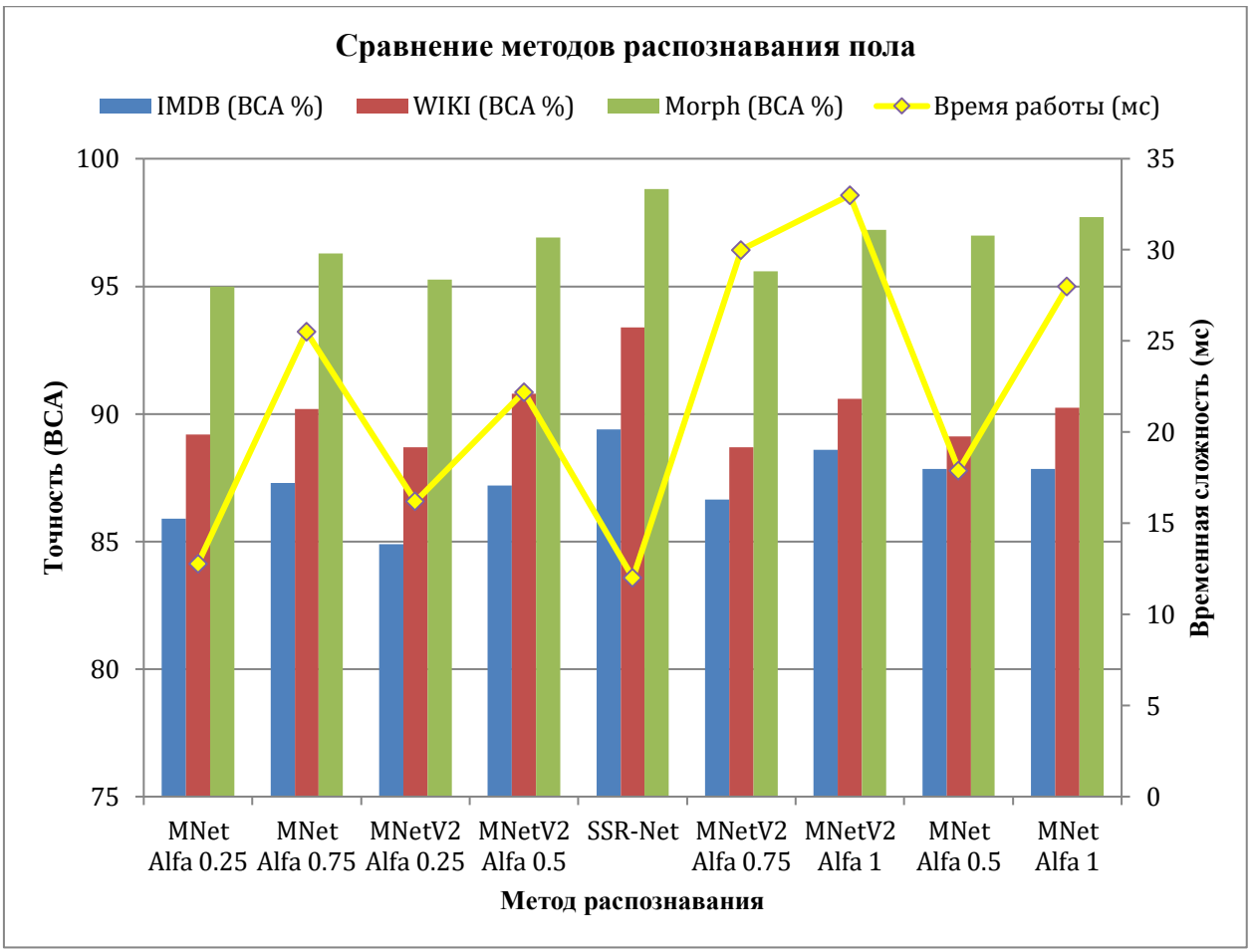


Рисунок 14. Результаты тестирования методов распознавания пола

ГЛАВА 3. РЕАЛИЗАЦИЯ СИСТЕМЫ РАСПОЗНАВАНИЯ ДЕМОГРАФИЧЕСКИХ ХАРАКТЕРИСТИК ЧЕЛОВЕКА В ВИДЕОПОТОКЕ

3.1. Проектирование архитектуры системы

Архитектура системы распознавания демографических характеристик человека в видеопотоке представляет собой несколько блоков: входные данные, распознавание демографических характеристик, выходные данные.

Блок входных данных

При работе системы требуется получение видеопотока для дальнейшего распознавания. Опционально видеопоток может быть получен на вход системы как из другой сторонней системы, так и непосредственно с камеры. При реализации системы для получения потока используется камера, осуществляющая непрерывный захват кадров с кадровой частотой 25 – 30 кадров в секунду.

Блок распознавания демографических характеристик

Обработка потока данных в виде кадров, получаемых от предыдущего блока системы, выделение на полученном кадре прогнозируемых положений лиц, распознавание атрибутов по изображению этих лиц, формирование выходных данных в определенном формате и их передача для дальнейшей обработки следующему блоку системы.

Блок выходных данных

Результат распознавания может быть передан на вход другой системы для дальнейшей обработки, может быть представлен в виде записи в хранилище данных либо выведен на экран визуализации. При реализации системы используется экран для отображения в реальном времени результатов распознавания.

Для реализации системы был выбран многопоточный подход к разработке архитектуры [34]. Полная архитектура системы распознавания демографических характеристик человека в видеопотоке представлена на рисунке 15.

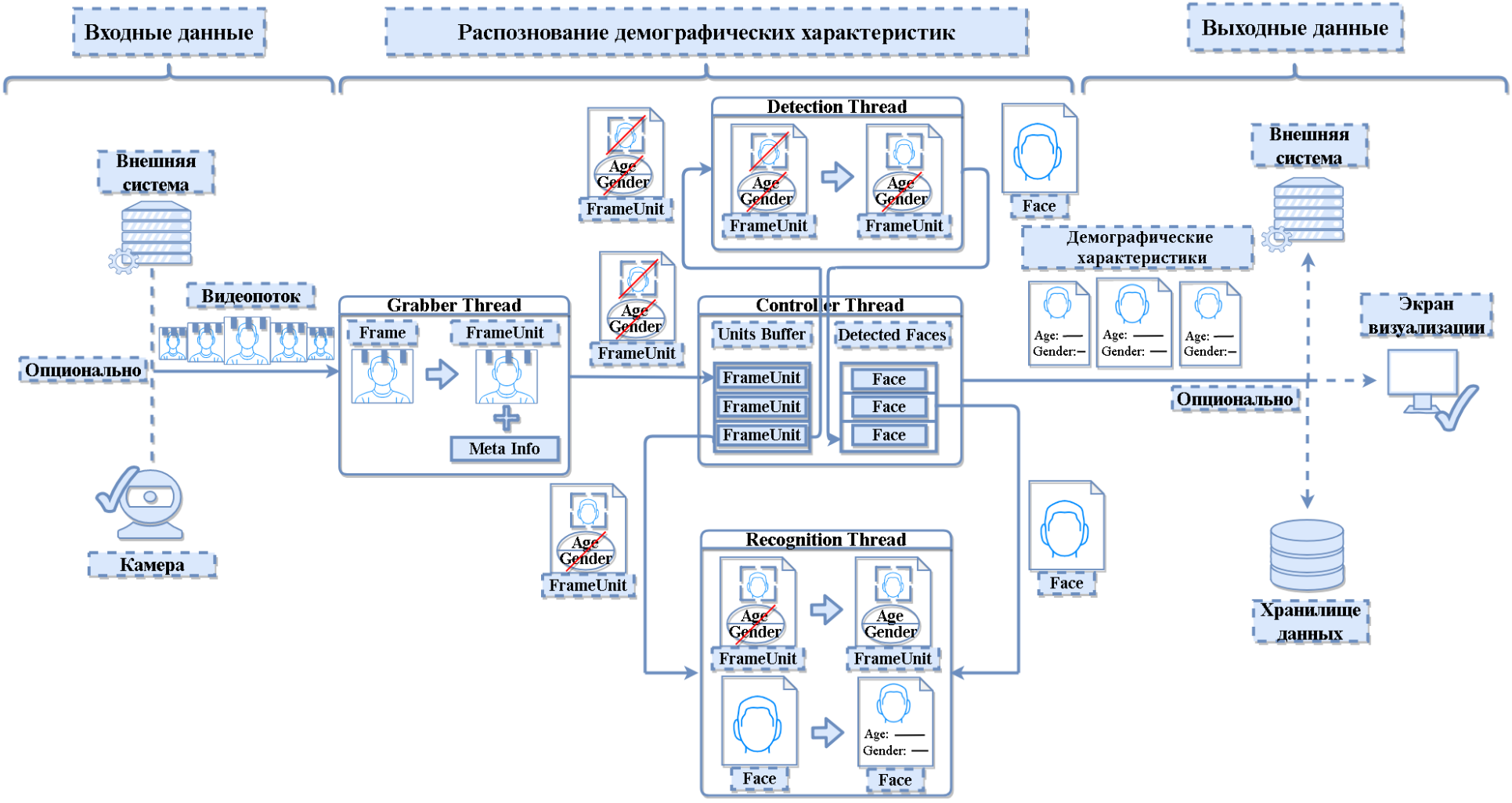


Рисунок 15. Архитектура системы распознавания демографических характеристик человека в видеопотоке

Решение задачи распознавания демографических характеристик человека представляет собой решение трех подзадач:

- Детектирование лица, то есть его локализация на обрабатываемом изображении: входными данными является обрабатываемое изображение, выходными данными — прогнозируемые координаты границ лица в системе входного изображения; представляет собой решение задачи детектирования объектов;
- Распознавание возраста по изображению лица: входными данными является изображение обрабатываемого лица, выходными данными является прогнозируемый возраст объекта; представляет собой решение задачи регрессии;
- Распознавание пола по изображению лица: входными данными является изображение обрабатываемого лица, выходными данными является прогнозируемый пол объекта; представляет собой решение задачи бинарной классификации.

Подзадачи должны выполняться согласовано, корректно распределяя между собой ограниченные вычислительные ресурсы.

Система распознавания демографических характеристик представлена несколькими потоками:

- *Controller Thread* — контролирующий поток системы, являющийся родительским для всех других потоков и осуществляющий подготовку и передачу выходных данных, в частности — взаимодействующий с экраном отрисовки;
- *Grabber Thread* — поток, осуществляющий прием входных данных, в частности — обеспечивающий получение видеокадров с камеры;
- *Detection Thread* — поток, отвечающий за детектирование лиц на полученных кадрах;
- *Recognition Thread* — поток, отвечающий за распознавание атрибутов для полученных лиц.

Процесс работы системы распознавания построен следующим образом:

1. *Grabber Thread*, взаимодействующий с камерой, обрабатывает получаемый поток кадров, оборачивая каждый в объект класса *FrameUnit* (в дальнейшем — кадр), содержащий необходимую метаинформацию, и передавая его в *Controller Thread*;

2. Полученный в *Controller Thread* кадр помещается в *FIFO* буфер — *Units Buffer* и передается для визуализации на экран: визуализация для кадра происходит в момент его получения с аннотацией доступной на данном этапе в структуре *Detected Faces* информацией по детекции лиц и распознавании характеристик. Результаты по данному кадру будут визуализированы в дальнейшем, когда он будет обработан потоками детекции и распознавания;

3. *Detection Thread* в асинхронном режиме получает из буфера кадр, для которого не была произведена детекция лиц. Координаты найденных лиц передаются в *Controller Thread*, где сопоставляются с детектированными на ранее обработанных кадрах лицами, сохраненными в структуре *Detected Faces* для усреднения результата работы дальнейших этапов распознавания.

4. *Recognition Thread* в асинхронном режиме получает из буфера кадр, для которого была произведена детекция лица, но ещё не было произведено распознавание характеристик. Результат распознавания для каждого лица записывается с усреднением по предыдущим результатам распознавания при их наличии.

Построение системы подобным образом позволяет производить визуализацию с высокой кадровой частотой, несмотря на возможные этапы обработки, требующие длительного времени. Так как процесс отрисовки происходит сразу же после получения кадра с использованием ранее полученной информации, то отрисовка осуществляется с кадровой частотой камеры. Контролируя частоту обработок в потоках, работающих в асинхронном режиме, можно регулировать частоту обновления различной информации по кадру для балансировки использования вычислительных ресурсов.

При этом возможно расширение системы для распознавания дополнительной информации с помощью добавления новых асинхронно работающих потоков.

3.2. Техническая архитектура системы

Для реализации системы требуется определить методы для решения выделенных подзадач. Подходами к детектированию и распознаванию, в соответствии с проведенным сравнительным анализом, были выбраны *Сверточные Нейронные Сети* с архитектурами, наилучшим образом отвечающими требованиям разрабатываемой системы:

- Архитектура *SSR-Net* с размерностью входного слоя 64x64x3: используется для распознавания и пола, и возраста (2 отдельные сети для каждой задачи);
- Архитектура *MobileNet* с $\alpha = 0.5$ и размерностью входного слоя 240x180x3: используется для детектирования лиц [35].

Скорость работы выбранных методов и их точность приведена в таблице 12. Скорость работы приводится для оборудования со следующими характеристиками:

- CPU: Intel(R) Core(TM) i3-4030U CPU @ 1,90GHz;
- OS: Ubuntu, 18.04.

Таблица 12. Характеристики используемых методов

Архитектура	Задача	Скорость	Точность
SSD MobileNet 240x180x3 $\alpha = 0.5$	Детектирование лица	27 мс / 37 FPS	0.67 mAP (IoU 0.5) для WIDER Face
SSR-Net 64x64x3	Распознавание возраста	14 мс / 71 FPS	3.24 MAE для MORPH2
SSR-Net 64x64x3	Распознавание пола	14 мс / 71 FPS	98.8 % BCA для MORPH2

Язык разработки: Python 3.6.

Среда разработки: PyCharm 2019 1.2.

Технологии и библиотеки: Dlib 19.17, OpenCV 4.0.0, Tensorflow 13.0.0, Keras 2.0.2.

3.3. Разработка системы

Для реализации системы было разработано 10 классов: 6 основных классов, реализующих разработанную потоковую архитектуру, и 4 вспомогательных класса. На рисунке 16 приводится полная диаграмма классов.

Разработанные вспомогательные классы с их описанием приведены в таблице 13.

Таблица 13. Описание вспомогательных классов системы

Класс	Описание
SSDFaceDetector	Класс для детектирования лица на изображении с использованием архитектуры SSD
DlibFaceAligner	Класс для выравнивания положения лица относительно изображения с использованием библиотеки dlib для детектирования ключевых точек лица (см. параграф 2.2.2.)
SSRNetRecognizer	Класс для распознавания демографических характеристик по изображению лица с использованием архитектуры SSR-Net
DrawUtils	Класс методов для визуализации на изображении полученной в ходе работы системы информации

Разработанные классы потоков: *Detection Thread*, *Recognition Thread*, *Grabber Thread*, *Controller Thread* наследуются от класса *Thread* модуля t*hreading* [36]. Метод *run* класса *Thread* является методом, вызываемым в отдельном потоке после его создания, поэтому в реализованных классах переопределяется метод *run*, в котором и реализуется работа потока.

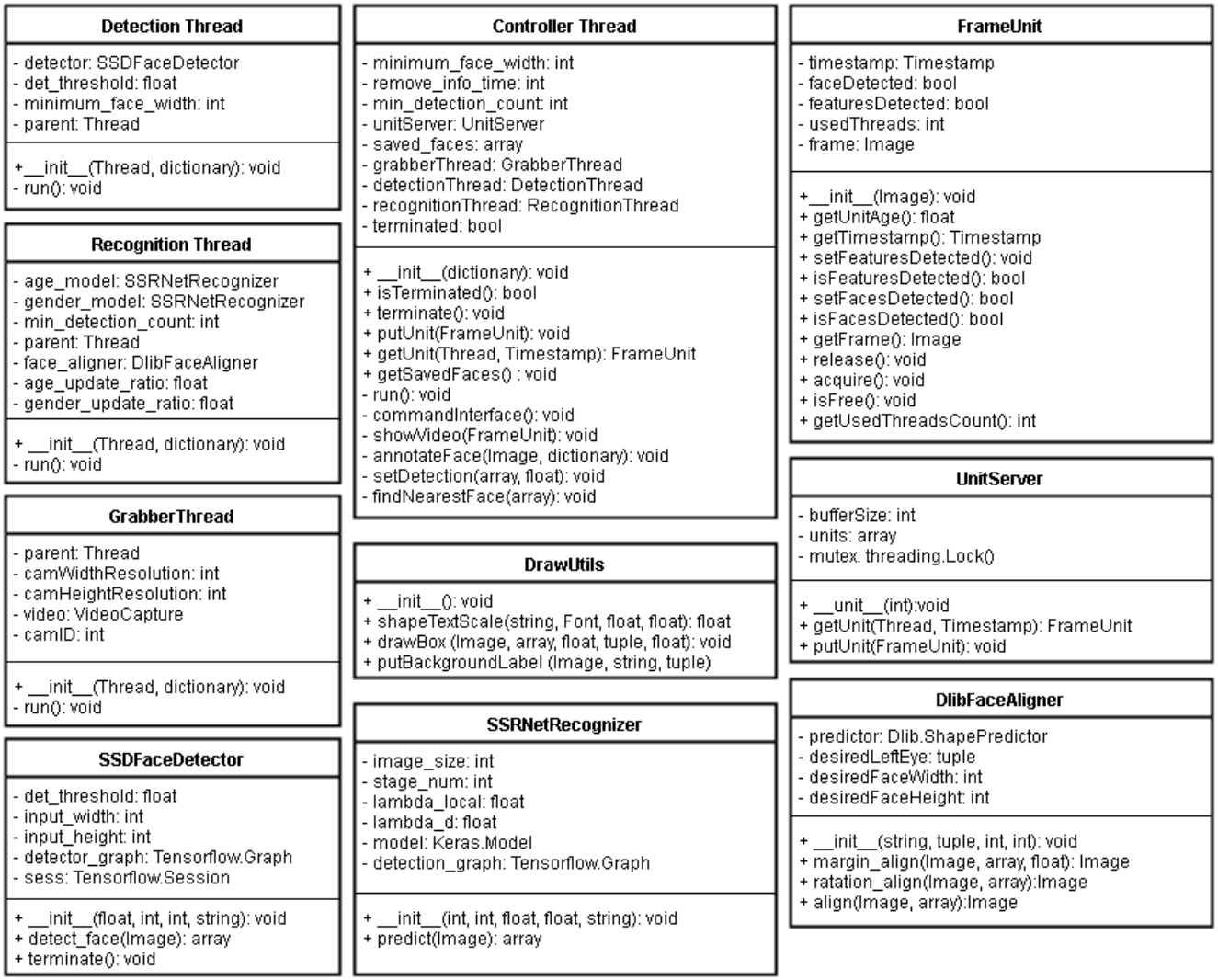


Рисунок 16. Диаграмма классов системы распознавания демографических характеристик

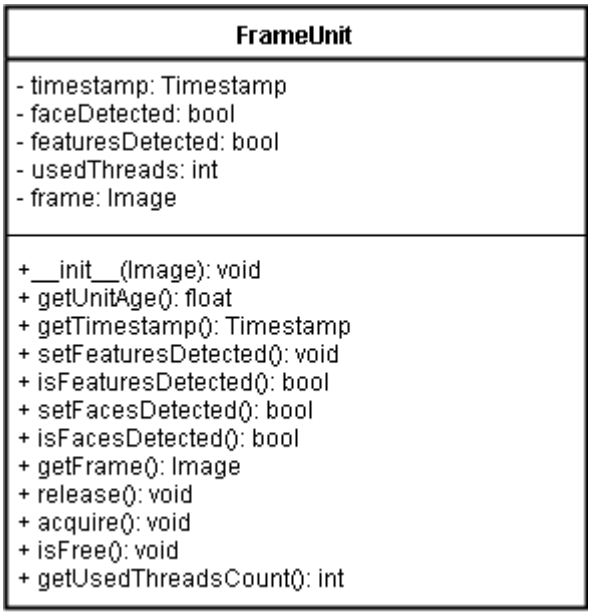


Рисунок 17. Класс FrameUnit

Класс *FrameUnit* (см. рисунок 17) представляет собой класс-обертку с добавлением мета-информации для получаемого с видеокамеры кадра. В таблице 14 приведено описание полей класса.

Таблица 14. Описание полей класса FrameUnit

Поле	Описание
timestamp	Время получения системой кадра из видеопотока
faceDetected	Обозначение был ли кадр обработан методом детектирования лица
featuresDetected	Обозначение был ли кадр обработан методами распознавания
usedThreads	Количество потоков, которыми захвачен кадр
frame	Объект видеокадра, полученного с видеокамеры

Методы класса *FrameUnit* являются *геттерами* и *сеттерами* для описанных полей.

Класс *UnitServer* (см. рисунок 18) представляет собой реализацию *FIFO* буфера для объектов *FrameUnit*. Полями класса является: размер буфера *bufferSize*, экземпляр буфера кадров *units* и экземпляр мьютекса *mutex*,

используемый для корректной обработки взаимодействия нескольких потоков с буфером.

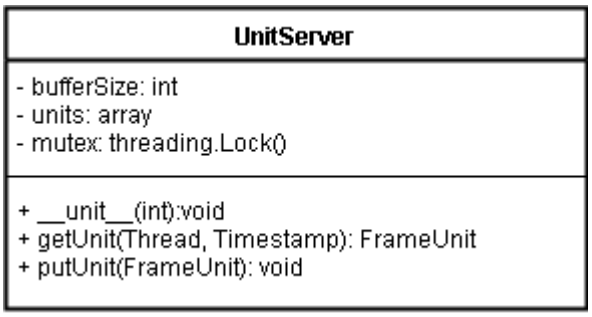


Рисунок 18. Класс UnitServer

В классе реализованы методы: метод расположения элемента в буфер *putUnit* и метод извлечения элемента из него *getUnit* с опцией получения конкретного элемента по его мета-информации: такая опция необходима на этапе распознавания характеристик для получения изображения лица по ранее детектированным координатам его положения на входном изображении.

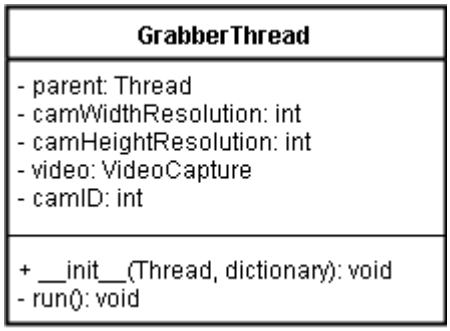


Рисунок 19. Класс GrabberThread

Класс *GrabberThread* (см. рисунок 19) отвечает за получение с камеры видеокадра, создание экземпляра объекта *FrameUnit* и его передачу в *Controller Thread* для размещения в буфере. Поля класса представляют собой: ссылку на родительский *ControllerThread* поток *parent*, идентификатор камеры *camID*, параметры разрешения для входного кадра: *camWidthResolution* и *camHeightResolutio*n, ссылку на открытый поток кадров *video*.

Весь процесс обработки кадров вынесен в переопределенный метод *run*.

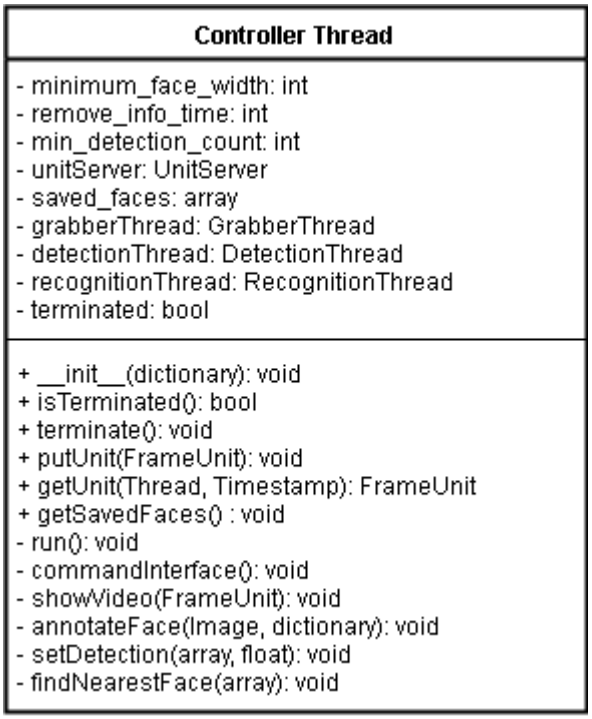


Рисунок 20. Класс Controller Thread

Класс *Controller Thread* является контролирующим и родительским потоком: все остальные потоки порождаются им. В таблице 15 и 16 приведены описания полей и методов класса *Controller Thread*.

Таблица 15. Описание полей класса Controller Thread

Поле	Описание
unitServer	Экземпляр FIFO буфера FrameUnit-объектов
saved_faces	Массив уже детектированных лиц с координатами расположения на изображении и текущими данными по распознаванию, используемыми для уточнения результата распознавания по одному лицу
minimum_detection_count	Пороговое значение необходимого количества детекций одного лица: используется для формирования подвыборки лиц при отрисовке
remove_info_time	Пороговое значение количества секунд, прошедшего с момента последней детекции

	лица, после которого информация о нем будет удалена из массива saved_faces
minimum_face_width	Минимально допустимый размер лица: устанавливается как максимальное расстояние между центроидами для одного и того же лица
terminated	Обозначение завершения работы потока
grabberThread	Экземпляр класса GrabberThread — дочерний поток
detectionThread	Экземпляр класса DetectionThread — дочерний поток
recognitionThread	Экземпляр класса RecognitionThread — дочерний поток

Таблица 16. Описание методов класса Controller Thread

Метод	Описание
putUnit()	Размещение кадра в буфер
getUnit()	Извлечение кадра из буфера
showVideo()	Визуализация кадра на экране
findNearestFace()	Поиск ближайшего к заданному координатами лицу с использованием проверки расстояния между центроидами
setDetection()	Обновление массива saved_faces в соответствии с новой информацией о детекции
annotateFace()	Аннотирование кадра в соответствии с результатами детектирования и распознавания из массива saved_faces
terminate()	Прерывание работы потока
getSavedFaces()	Геттер для поля saved_faces
isTerminated()	Геттер для поля terminated

commandInterface()	Реализация интерфейса для управления работой системы
run()	Переопределенный метод класса Thread

Метод *setDetection()* используется при изменении информации о детектированных лицах: требуется выяснить является ли результат новой детекции — обновлением информации для уже имеющихся элементов в массиве saved_faces, и требуется только добавить новую информацию для соответствующего элемента (прогнозируемые координаты лица, время детекции); либо детектировано новое лицо, и, соответственно, требуется добавить новый элемент в массив.

С целью определения совпадения лиц используется отслеживание расстояния между центроидами лиц, реализованное в методе *findNearestFace()*: результатом работы метода является элемент массива *saved_faces* с наименьшим расстоянием до нового элемента. В случае, если расстояние между центроидами не превосходит минимально допустимый размер лица, установленный параметром *minimum_face_width*, считается, что лица совпадают.

Так как требуется искать совпадения только среди недавно детектированных лиц, в методе *setDetection()* реализован соответствующий механизм: лица с временным промежутком в секундах до последнего результата детекции большим чем пороговое значение параметра *remove_info_time* удаляются из дальнейшего рассмотрения при поиске совпадений.

Реализация методов *findNearestFace()* и *setDetection()* приведена в приложении 1 и 2 соответственно.

Класс *DetectionThread* (см. рисунок 21) в асинхронном режиме выполняет процесс детектирования лица. Описание полей класса приведено в таблице 17.

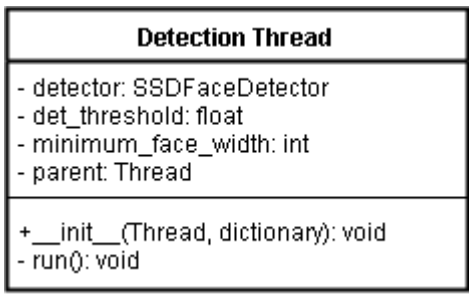


Рисунок 21. Класс Detection Thread

Таблица 17. Описание полей класса Detection Thread

Поле	Описание
parent	Ссылка на родительский поток класса Controller Thread
det_threshold	Пороговое значения оценки точности детектирования
minimum_face_width	Пороговое значение минимального размера лица
detector	Экземпляр класса SSDFaceDetector

Алгоритм процесса детектирования лица представляет собой следующие шаги:

1. Проверка состояния родительского потока: если работа родительского потока прервана — завершение процесса детектирования;
2. Обращение к родительскому потоку *Controller Thread* для получения из буфера кадров объекта без результата детекции. В случае если таких объектов в буфере нет — поток приостанавливается на некоторое время и происходит переход к началу шага 1;
3. Детектирование лиц с использованием класса *SSDFaceDetector* для полученного кадра;
4. Формирование выборки лиц из списка детекции, удовлетворяющей пороговым значениям *det_threshold* и *minimum_face_width*;

5. Передача сформированной выборки детектированных лиц в *Controller Thread* и переход к началу шага 1.

Алгоритм процесса детектирования лица реализован в переопределенном методе *run* (см. приложение 3).

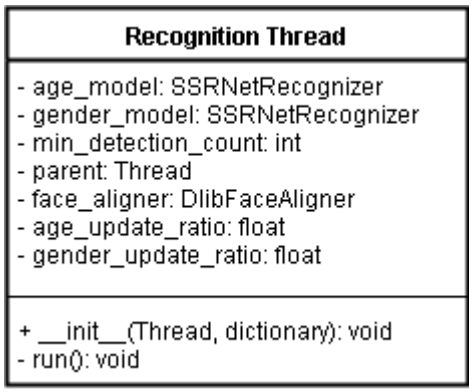


Рисунок 22. Класс Recognition Thread

Класс *Recognition Thread* (см. рисунок 22) в асинхронном режиме выполняет процесс распознавания пола и возраста по изображению лица. В таблице 18 приведено описание полей класса.

Таблица 18. Описание полей класса Recognition Thread

Поле	Описание
age_model	Экземпляр класса SSRNetRecognizer с моделью для распознавания возраста
gender_model	Экземпляр класса SSRNetRecognizer с моделью для распознавания пола
min_detection_count	Пороговое значение минимально необходимо числа детекций лица для проведения распознавания
parent	Ссылка на родительский поток класса Controller Thread
age_update_ratio	Коэффициент значимости новых результатов распознавания возраста относительно старых при усреднении
gender_update_ratio	Коэффициент значимости новых результатов распознавания пола относительно старых при

	усреднении
face_aligner	Экземпляр класса DlibFaceAligner

Выходной результат распознавания демографических характеристик по лицу представляется взвешенной комбинацией получаемого результата распознавания на текущем этапе и ранее полученных результатов.

Пусть $a \in [0; 100]$ — общий результат распознавания возраста по лицу, а $a_0 \in [0; 100]$ — полученный результат распознавания на текущей итерации, p_a — весовой коэффициент влияния нового результата на общий результат по возрасту. Тогда

$$a = a * (1 - p_a) + a_0 * p_a \quad (6)$$

Пусть $g \in [0; 1]$ — общая прогнозируемая вероятность принадлежности лица к классу, где 0 — это 100% вероятность отнесения к классу женского пола, а 1 — это 100% вероятность отнесения к классу мужского пола и, соответственно, 0.5 — пороговое значение; $g_0 \in [0; 1]$ — полученная на текущей итерации прогнозируемая вероятность принадлежности, а p_g — весовой коэффициент влияния нового результата на общий результат по полу. Тогда

$$g = g * (1 - p_g) + g_0 * p_g \quad (7)$$

Поля класса *Recognition Thread*: *age_update_ratio* и *gender_update_ratio* реализуют коэффициенты p_a и p_g соответственно.

Алгоритм процесса распознавания демографических характеристик представляет собой следующие шаги:

1. Проверка состояния родительского потока: если работа родительского потока прервана — завершение процесса распознавания;
2. Получение коллекции детектированных лиц из родительского потока *Controller Thread*. В случае если коллекция пуста — поток приостанавливается на некоторое время и происходит переход к началу шага 1;

3. Формирование из полученной коллекции выборки лиц, удовлетворяющей пороговому значению *min_detection_count*. В случае если выборка пуста — переход к началу шага 1.
4. Для каждого лица из выборки:
    1) Выравнивание лица с использованием класса *DlibFaceAligner*;
    2) Распознавание пола и возраста с использованием моделей класса *SSRNetRecognizer*;
    3) Формирование общего результата детекции по лицу в соответствии с описанным выше методом взвешенной комбинации.
5. Переход к началу шага 1.

Описанный алгоритм процесса распознавания демографических характеристик реализован в переопределенном методе *run* (см. приложение 4).

ГЛАВА 4. ТЕСТИРОВАНИЕ РАЗРАБОТАННОЙ СИСТЕМЫ

4.1. Работа системы

Разработанная система представляет собой *Python*-скрипт, запускаемый из командной строки. Все основные параметры системы доступны для конфигурирования при помощи инициализационного файла *config.ini* [37]. В таблице 19 приведены основные конфигурационные параметры системы, разделенные по блокам.

Таблица 19. Конфигурационные параметры системы

Блок системы	Параметры
Получение кадра	Идентификационный номер камеры; Разрешение входного кадра: ширина и высота
Буфер кадров	Размер буфера кадров
Отрисовка	Параметры окна отрисовки: положение, размер, заголовок и др. Параметры шрифтов: гарнитура шрифта, вид шрифта, цвет и др. Параметры объектов (границы детекции лица, лейблы с результатами распознавания): цвета оформления, положение, размер и др.
Детекция лиц	Параметры модели детектирования: путь к файлам со структурой и весами, размерность входного слоя; Параметры фильтрации: пороговое значение минимального размера лица, пороговое значения времени с момента последней детекции и др.
Распознавание характеристик	Параметры моделей распознавания; Параметры усреднения результатов по распознаваемым характеристикам

Управление системой реализовано при помощи командного интерфейса со следующими опциями:

- (S)peed — вывод информации по скорости работы отдельных компонентов системы (отображение, детекция лица, распознавание) в *FPS*;
- (I)nfo — вывод информации по обрабатываемым лицам: количество детекций для лица, прошедшее время с момента последней детекции, текущие результаты распознавания, количество итераций распознавания и др.
- (V)isualization — запуск \ остановка визуализации;
- (W)rite — сохранение обрабатываемых кадров;
- (Q)uit — завершение работы системы.

Визуализация происходит в отдельном окне, запускаемом при старте системы, и представляет собой отрисовку кадра, полученного с камеры, где для каждого детектированного лица отображаются его прогнозируемые границы и результат распознавания демографических характеристик (см. рисунок 23).

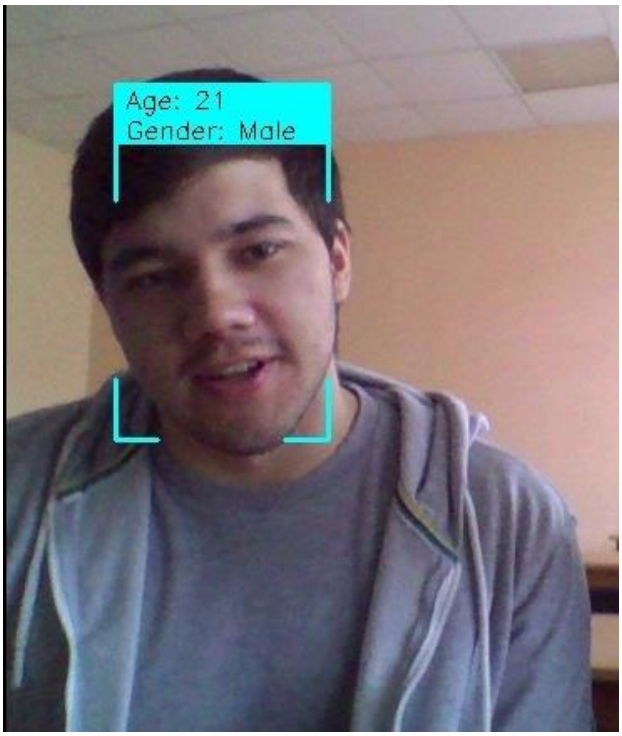


Рисунок 23. Пример работы блока визуализации в системе

Информация по скорости работы на CPU-устройстве блоков системы и характеристиками оборудования, на котором проводились замеры, приводится в таблице 20.

Таблица 20. Скорость работы блоков системы

Характеристика		Значение
Оборудование тестирования	Операционная система	Ubuntu, 18.04
	CPU	Intel(R) Core(TM) i3-4030U CPU @ 1,90GHz
	Кадровая частота камеры	30 FPS
	Разрешение кадров	WGA: 640x480
Скорость работы блоков системы	Скорость работы блока визуализации	30 FPS
	Скорость работы блока детекции лиц	24 FPS
	Скорость работы блока распознавания демографических характеристик человека	11 FPS

Визуализация процесса обработки системой кадра с момента его получения представлена на рисунке 24.

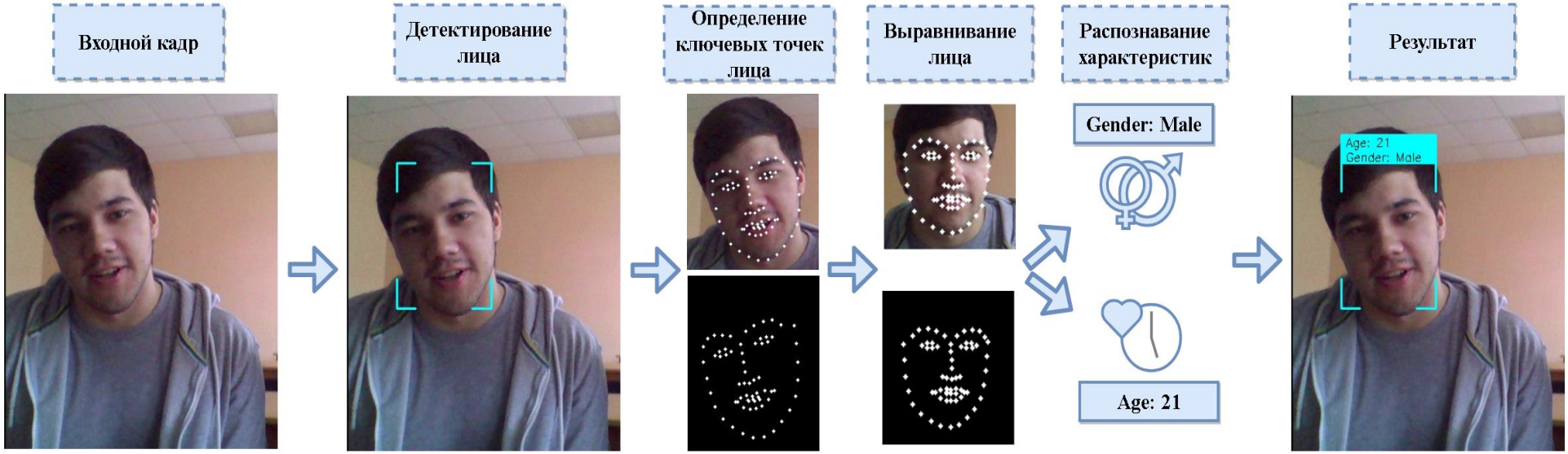


Рисунок 24. Визуализация процесса обработки кадра системой распознавания демографических характеристик человека

4.2. Параметры системы

Данные для обучения

При тестировании точности работы системы в реальных условиях для обучения используемых моделей *CHC* требуется выбрать соответствующие задаче данные. И, если используемая модель детектирования лица, обученная на датасете WIDER Face, может быть применена при работе в реальных условиях, то для моделей распознавания характеристик человека по изображению проблемой является отсутствие необходимого объема данных, собранного в реальных условиях с точной аннотацией характеристик. Описанные при проведении экспериментов базы не подходят для обучения моделей распознавания возраста, используемых для работы в реальных условиях, так как либо имеют неточности в аннотациях, либо были собраны в контролируемой среде. Сравнение датасетов приводится в таблице 21.

Таблица 21. Сравнение датасетов

Датасет	Данные по полу	Данные по возрасту	Среда сбора данных	Кол-во элементов	Ошибки в данных
IMDB + WIKI	Да	Да	Реальные условия	37 665 + 167 300	Да
MORPH 2	Да	Да	Контролируемая среда	38 499	Нет
CVPR 2016 LAP	Да	Нет	Реальные условия	3906	Нет

Датасет *CVPR 2016 LAP* представляет собой базу изображений лиц, собранных в реальных условиях, с точной аннотацией возраста от 0 до 89 лет [38]. Модель распознавания возраста была обучена, в соответствии с ранее описанными параметрами (см. параграф 2.3.1), на датасете *CVPR 2016 LAP* с предобучением на датасете *IMDB-WIKI*. Распределение возраста в базе *CVPR 2016 LAP* приведено на рисунке 25.

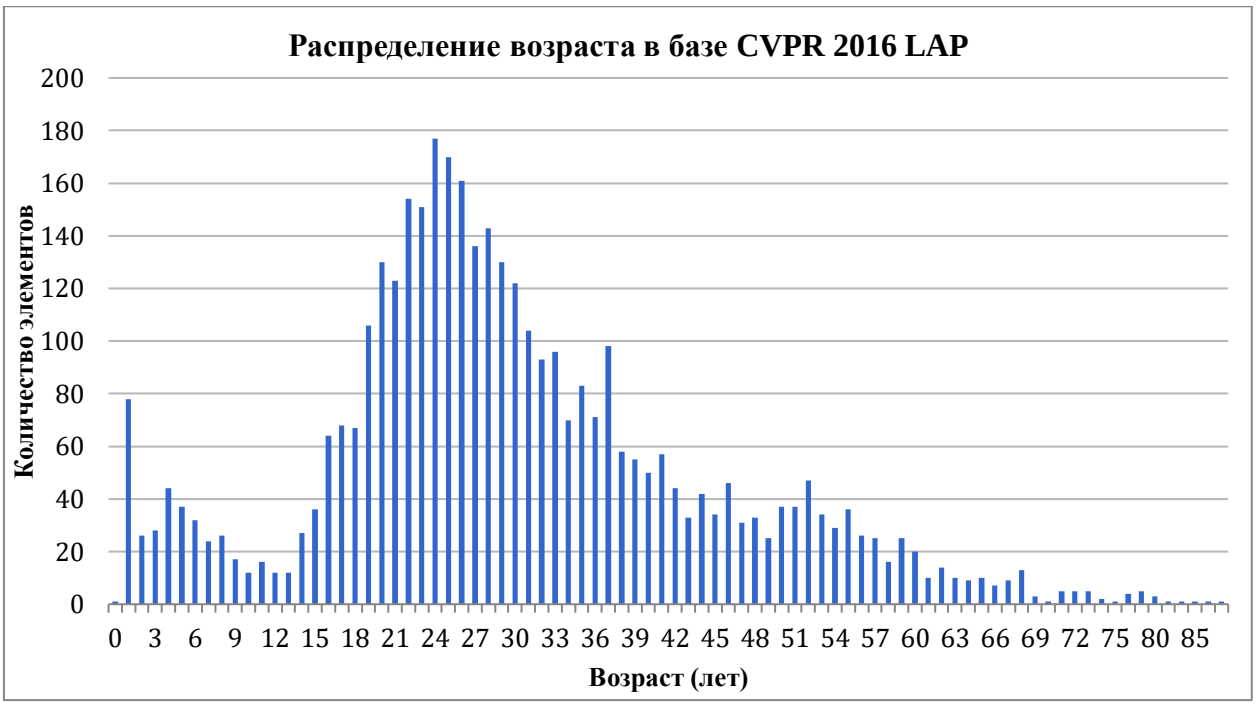


Рисунок 25. Распределение возраста в базе CVPR 2016 LAP

Согласно таблице 21, так как датасет *MORPH2* не пригоден для обучения модели, используемой в реальных условиях в силу сбора данных в контролируемой среде, то вариантом для обучения модели распознавания пола остается датасет *IMDB-WIKI*. Параметры моделей детекции и распознавания, используемых при тестировании точности работы системы приведены в таблице 22.

Таблица 22. Характеристики моделей, используемых при тестировании

Модель	Задача	Предобучение (датасет)	Обучение (датасет)	Точность
SSD MobileNet 240x180x3 α = 0.5	Детектирование лица	Нет	WIDER Face	0.67 mAP (IoU 0.5) для WIDER Face
SSR-Net 64x64x3	Распознавание возраста	IMDB + WIKI	CVPR 2016 LAP	4.91 MAE
SSR-Net 64x64x3	Распознавание пола	IMDB	WIKI	93.4 % BCA

Конфигурационные параметры системы

При проведении тестирования системы были заданы определенные конфигурационные параметры, основные из которых приведены в таблице 23.

Таблица 23. Параметры системы при проведении тестирования

Параметр	Значение
Размер буфера кадров (buffer_size)	8
Разрешение входного кадра (input_resolution)	WGA: 640x480
Пороговое значение точности для детекции лица (detection_threshold)	0.7
Пороговое значение минимального размера лиц (minimum_face_width): значение в пикселях	40
Пороговое значение промежутка с момента последней детекции (remove_face_info_time): значение в секундах	0.5
Пороговое значение минимального числа детекций для одного лица (min_detection_count)	2
Весовой коэффициент влияния нового результата распознавания на общий результат по возрасту	0.1
Весовой коэффициент влияния нового результата распознавания на общий результат по полу	0.2

4.3. Проведение тестирования

Тестирование проводилось на группе из 60 человек 19 возрастных категорий от 6 до 50 лет: 57% человек — мужчин, 43% — женщин.

Для тестирования точности распознавания возраста используется метрика *MAE*, а для распознавания пола — точность бинарной классификации (см. параграф 2.3.2).

Результат распознавания представляет собой комбинацию основного результата распознавания и нового, полученного на данном этапе результата. Так как обработка кадров компонентом распознавания ведется с высокой скоростью (см. таблицу 20), то за небольшой промежуток времени может быть обработано несколько кадров для одного лица, что должно увеличить точность распознавания при усреднении результатов. Для сравнения

результатов при проведении тестирования собирались значения по распознаванию для 1 и 30 кадра (около 3 секунд обработки).

Средний результат *MAE* для 1 кадра по 60 элементам — 4.97, в то время как для 30 кадра — 4.06. Видно значительное улучшение точности в среднем случае. На рисунке 26 приведен график результатов тестирования точности по возрасту.

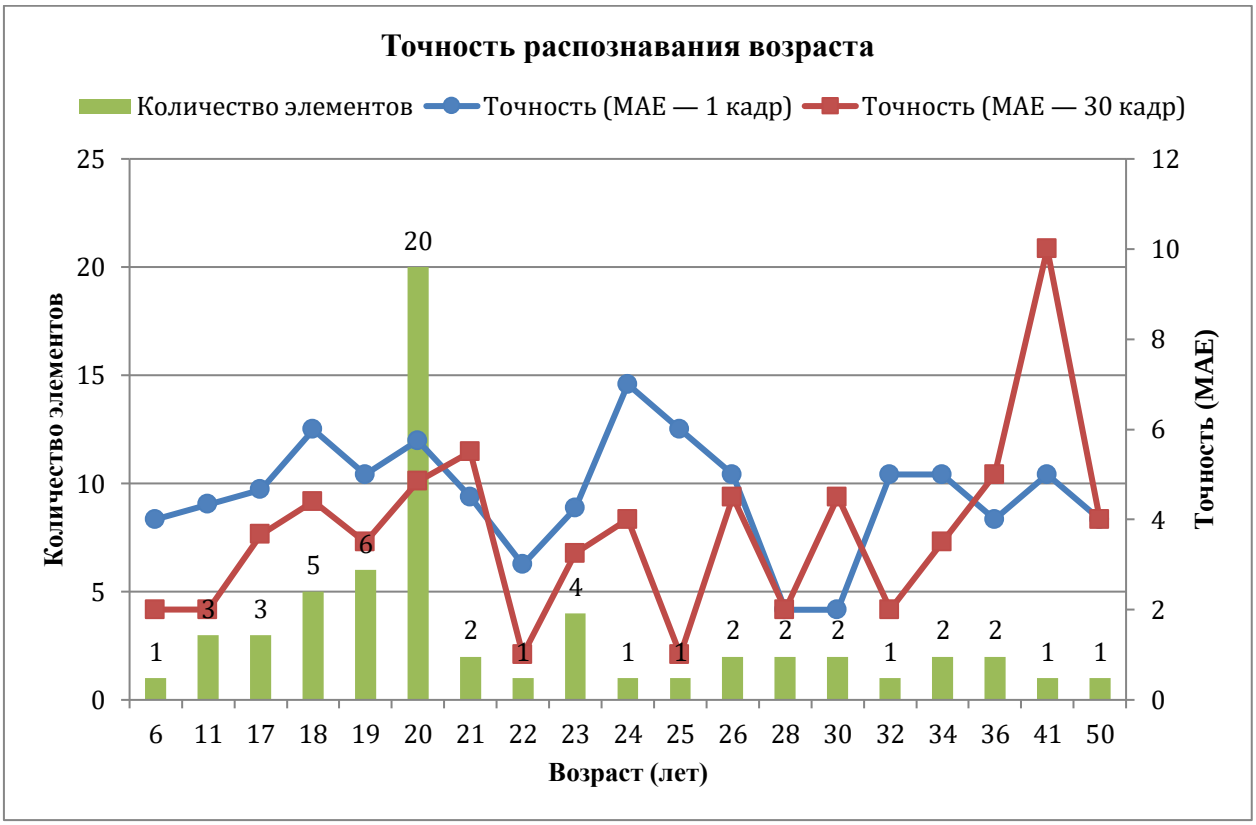


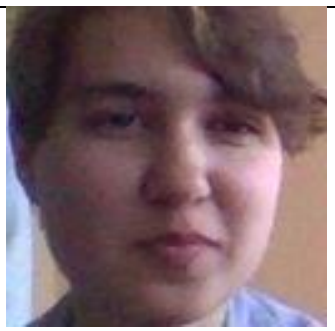
Рисунок 26. Результаты тестирования точности распознавания возраста

Точность бинарной классификации пола для 1 кадра составила при тестировании 91.2 % и 92.3 % для мужского и женского пола соответственно. На 30 кадре точность распознавания для каждого из классов — 100%.

По двум показателям удалось добиться значительного улучшения точности при использовании подхода с комбинированием основного и нового результата распознавания.

В таблице 24 приведены примеры изображений лиц людей из тестовой группы со значениями реального и спрогнозированного системой возраста для 30 кадра.

Таблица 24. Примеры изображений со значениями возраста

Реальный: 32	Реальный: 21	Реальный:23
		
Прогноз: 34	Прогноз: 26	Прогноз: 21
Реальный: 20	Реальный: 41	Реальный: 6
		
Прогноз: 23	Прогноз: 31	Прогноз: 8
Реальный: 36	Реальный: 11	Реальный:20
		
Прогноз: 34	Прогноз: 13	Прогноз: 18

Дальнейшими этапами улучшения точности работы системы распознавания демографических характеристик человека в видеопотоке может являться:

- Проведение кросс-тестирования различных параметров обучения для архитектуры сети;

- Увеличение инвариантности архитектуры к разнице реальных условий работы и условий обучающей выборки (освещение, положение камеры относительно объекта распознавания и др.) с помощью применения различных параметров предобработки данных обучения и внесения изменений в архитектуру сети (изменение сверточной структуры, изменение параметров регуляризации и пр.);
- Подготовка данных (разработка датасета), соответствующих поставленной задаче (реальным условиям работы системы).

Требуется также проведение в дальнейшем дополнительных этапов тестирования системы с целью получения более полных данных о точности её работы.

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы была разработана система распознавания демографических характеристик человека в видеопотоке с реализацией вычислений на CPU, для этого были решены следующие задачи:

- Сформулированы требования к разрабатываемой системе;
- Изучены подходы к детектированию лиц на изображении и распознаванию характеристик человека по изображению его лица;
- Проведен сравнительный анализ подходов по показателям точности и скорости работы, выбраны модели, наиболее точно отвечающие поставленным требованиям;
- Спроектирована и разработана на языке *Python* система распознавания;
- Проведено тестирование точности работы системы в реальных условиях;
- Намечены планы дальнейшей работы в рамках разработанной системы и направления в целом.

По результатам тестирования в реальных условиях система показала точность распознавания пола — 100% и среднюю абсолютную ошибку в распознавании возраста — 4.07.

Дальнейшими этапами является работа над увеличением точности работы системы и её устойчивости к различным условиям среды, а также разработка системы учета посетителей торговых помещений с использованием разработанной в рамках данной работы системы как полноценного модуля.

СПИСОК ЛИТЕРАТУРЫ

1. Verschae, Rodrigo & Ruiz-del-Solar, Javier. Object Detection: Current and Future Directions. (2015). [Электронный ресурс]. – Режим доступа: https://www.researchgate.net/publication/284167004_Object_Detection_Current_and_Future_Directions (дата обращения 16.02.2019).
2. Mackin, Alex & Zhang, Fan & Bull, David. A Study of High Frame Rate Video Formats. (2018). [Электронный ресурс]. – Режим доступа: https://www.researchgate.net/publication/328896584_A_Study_of_High_Frame_Rate_Video_Formats (дата обращения 24.02.2019).
3. Viola, Paul & Jones, Michael. Rapid Object Detection using a Boosted Cascade of Simple Features. (2001). [Электронный ресурс]. – Режим доступа: https://www.researchgate.net/publication/3940582_Rapid_Object_Detection_using_a_Boosted_Cascade_of_Simple_Features (дата обращения 14.02.2019).
4. Neubeck, Alexander & Van Gool, Luc. Efficient Non-Maximum Suppression. (2006). [Электронный ресурс]. – Режим доступа: https://www.researchgate.net/publication/220929789_Efficient_Non-Maximum_Suppression (дата обращения 16.02.2019).
5. N. Dalal and B. Triggs, Histograms of oriented gradients for human detection. (2005). [Электронный ресурс]. – Режим доступа: https://www.researchgate.net/publication/220929789_Efficient_Non-Maximum_Suppression (дата обращения 19.03.2019).
6. K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. (2016). [Электронный ресурс]. – Режим доступа: https://arxiv.org/abs/1604.02878 (дата обращения 27.02.2019).
7. Liu, Wei & Anguelov, Dragomir & Erhan, Dumitru & Szegedy, Christian & Reed, Scott & Fu, Cheng-Yang & C. Berg, Alexander. SSD: Single Shot MultiBox Detector. (2016). [Электронный ресурс]. – Режим доступа: https://arxiv.org/abs/1512.02325 (дата обращения 07.03.2019).

8. Zhang, Shifeng & Zhu, Xiangyu & Lei, Zhen & Shi, Hailin & Wang, Xiaobo & Li, Stan. FaceBoxes: A CPU Real-time Face Detector with High Accuracy. (2017). [Электронный ресурс]. – Режим доступа: https://arxiv.org/abs/1708.05234 (дата обращения 06.04.2019).
9. Detection evaluation [Электронный ресурс]. – Режим доступа: http://cocodataset.org/#detection-eval (дата обращения 20.03.2019).
10. Shuo Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. WIDER FACE: A face detection benchmark. (2016). [Электронный ресурс]. – Режим доступа: https://arxiv.org/abs/1511.06523 (дата обращения 16.04.2019).
11. Кеннет Рейтц, Таня Шлюссер. Автостопом по Python. – СПб.: Питер, 2017. – 336 с.
12. Tensorflow detection model zoo [Электронный ресурс]. – Режим доступа: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md (дата обращения 12.04.2019).
13. Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. (2017). [Электронный ресурс]. – Режим доступа: https://arxiv.org/abs/1704.04861 (дата обращения 14.05.2019).
14. Mark Sandler Andrew Howard Menglong Zhu Andrey Zhmoginov Liang-Chieh Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks. (2019). [Электронный ресурс]. – Режим доступа: https://arxiv.org/abs/1801.04381 (дата обращения 26.05.2019).
15. Aspect ratios in digital photography [Электронный ресурс]. – Режим доступа: https://expertphotography.com/aspect-ratio-photography (дата обращения 18.04.2019).
16. Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Neural Networks [Электронный ресурс]. – Режим доступа: https://github.com/kpzhang93/MTCNN_face_detection_alignment (дата обращения 15.05.2019).

17. OpenCV. Haar cascades [Электронный ресурс]. – Режим доступа: https://github.com/opencv/opencv/tree/master/data/haarcascades (дата обращения 11.05.2019).
18. Dlib face detector. [Электронный ресурс]. – Режим доступа: http://dlib.net/face_detector.py.html (дата обращения 14.05.2019).
19. N. Gourier, D. Hall, J. L. Crowley. Estimating Face Orientation from Robust Detection of Salient Facial Features. (2004). [Электронный ресурс]. – Режим доступа: https://www.researchgate.net/publication/240617517_Estimating_Face_orientation_from_Robust_Detection_of_Salient_Facial_Structures (дата обращения 21.05.2019).
20. Shichuan Du, Yong Tao, and Aleix M. Martinez. Compound facial expressions of emotion. (2014). [Электронный ресурс]. – Режим доступа: https://www.pnas.org/content/111/15/E1454 (дата обращения 13.05.2019).
21. Саймон Хайкин. Нейронные сети. Полный курс. – Москва: Вильямс, 2018. – 1104 с.
22. How to Reduce Overfitting With Dropout Regularization in Keras. [Электронный ресурс]. – Режим доступа: https://machinelearningmastery.com/how-to-reduce-overfitting-with-dropout-regularization-in-keras/ (дата обращения 26.04.2019).
23. Yang, Tsun-Yi & Huang, Yi-Hsuan & Lin, Yen-Yu & Hsiu, Pi-Cheng & Chuang, Yung-Yu. SSR-Net: A Compact Soft Stagewise Regression Network for Age Estimation. (2018). [Электронный ресурс]. – Режим доступа: https://www.researchgate.net/publication/326208517_SSR-Net_A_Compact_Soft_Stagewise_Regression_Network_for_Age_Estimation (дата обращения 20.05.2019).
24. Rasmus Rothe and Radu Timofte and Luc Van Gool. DEX: Deep EXpectation of apparent age from a single image. (2015). [Электронный ресурс]. – Режим доступа: https://www.researchgate.net/publication/306023833_Deep_Expectation_of_Re

al_and_Apparent_Age_from_a_Single_Image_Without_Facial_Landmarks (дата обращения 05.05.2019).

25. Karl Ricanek Jr and Tamirat Tesafaye. MORPH: A Longitudinal Image Database of Normal Adult Age-Progression. (2006). [Электронный ресурс]. – Режим доступа: https://ieeexplore.ieee.org/document/1613043 (дата обращения 03.05.2019).
26. Naimish Agarwal, Artus Krohn-Grimberghe, Ranjana Vyas. Facial Key Points Detection. (2017). [Электронный ресурс]. – Режим доступа: https://www.sparrho.com/item/facial-key-points-detection-using-deep-convolutional-neural-network-naimishnet/19950e4/ (дата обращения 14.05.2019).
27. Vahid Kazemi and Josephine Sullivan. One Millisecond Face Alignment with an Ensemble of Regression Trees (2014). [Электронный ресурс]. – Режим доступа: https://www.academia.edu/22496549/One_Millisecond_Face_Alignment_with_an_Ensemble_of_Regression_Trees (дата обращения 08.05.2019).
28. Dlib face landmark detector. [Электронный ресурс]. – Режим доступа: http://dlib.net/face_landmark_detection.py.html (дата обращения 19.05.2019).
29. Usage of metrics. [Электронный ресурс]. – Режим доступа: https://keras.io/metrics (дата обращения 21.05.2019).
30. Yunxuan Zhang, Li Liu, Cheng Li, and Chen Change Loy. Quantifying facial age by posterior of age comparisons. (2017). [Электронный ресурс]. – Режим доступа: https://arxiv.org/abs/1708.09687 (дата обращения 21.05.2019).
31. Eirikur Agustsson, Radu Timofte, and Luc Van Gool. Anchored regression networks applied to age estimation and super resolution. (2017). [Электронный ресурс]. – Режим доступа: http://openaccess.thecvf.com/content_ICCV_2017/papers/Agustsson_Anchored_Regression_Networks_ICCV_2017_paper.pdf (дата обращения 17.05.2019).

32. Rasmus Rothe, Radu Timofte, and Luc Van Gool. Some like it hot-visual guidance for preference prediction. (2016). [Электронный ресурс]. – Режим доступа: https://arxiv.org/abs/1510.07867 (дата обращения 16.05.2019).
33. Shixing Chen, Caojin Zhang, Ming Dong, Jialiang Le, and Mike Rao. Using ranking-CNN for age estimation. (2017). [Электронный ресурс]. – Режим доступа: https://www.researchgate.net/publication/316493165_Using_Ranking_CNN_for_Age_Estimation (дата обращения 20.05.2019).
34. Грегори Эндрюс. Основы многопоточного, параллельного и распределенного программирования. – Москва: Вильямс, 2003. – 512 с.
35. CS231n: Convolutional Neural Networks for Visual Recognition. [Электронный ресурс]. – Режим доступа: http://cs231n.stanford.edu (дата обращения 28.05.2019).
36. Библиотека Threading. [Электронный ресурс]. – Режим доступа: https://docs.python.org/3/library/threading.html (дата обращения 11.05.2019).
37. Configuration file parser. [Электронный ресурс]. – Режим доступа: https://docs.python.org/3/library/configparser.html (дата обращения 18.05.2019).
38. ChaLearn Looking at People and Faces of the World: Face AnalysisWorkshop and Challenge 2016. [Электронный ресурс]. – Режим доступа: https://ieeexplore.ieee.org/document/7789583 (дата обращения 04.06.2019).

ПРИЛОЖЕНИЕ 1

Реализация метода findNearestFace

```
def findNearestFace(self,coords):
    distances = []
    x1,y1,x2,y2 = coords

    face_center = [(x1+x2)/2,(y1+y2)/2]

        for saved_face in self.saved_faces:
            x1,y1,x2,y2 = np.mean(saved_face['boxes'],axis=0)
            saved_face_center = [(x1+x2)/2,(y1+y2)/2]

            distance = np.hypot(saved_face_center[0]-face_center[0],
                                saved_face_center[1]-face_center[1])
            distances.append(distance)

            if len(distances) == 0:
                minDist = None
                minIndx = None

            else:
                minDist = np.min(distances)
                minIndx = np.argmin(distances)

    return minIndx, minDist
```

ПРИЛОЖЕНИЕ 2

Реализация метода setDetection

```
def setDetection(self,boxes,timestamp):
    for box in boxes:

        #send coords to function
        indx, dist = self.findNearestFace(box)
        #check if detection is finded and dist is lower that minimum face width

        if dist is not None and dist < self.minimum_face_width:
          self.saved_faces[indx]['boxes'].append(box)
          self.saved_faces[indx]['timestamps'].append(timestamp)
          #we got new face

        else:
          self.saved_faces.append({'boxes':[box],'timestamps':[timestamp]})

    #remove old faces
    face_indx_to_remove = []
    now = time.time()

    for indx, saved_face in enumerate(self.saved_faces):

        #check if the latest detection was later than threshold value
        if now - saved_face['timestamps'][-1] > self.remove_info_time:
            face_indx_to_remove.append(indx)

    #remove bad faces
    try:
        for i in face_indx_to_remove:
            self.saved_faces.pop(i)

    except:
        pass
```

ПРИЛОЖЕНИЕ 3

Реализация метода run класса Detection Thread

```
def run(self):
    while not self.parent.isTerminated():
        unit = None
        while unit == None:
            unit = self.parent.getUnit(self)
            if unit == None: #We don't have avaliable units yet
                time.sleep(0.1)

            if self.parent.isTerminated():
                break

        if self.parent.isTerminated():
            break

        img = unit.getFrame()

        #copy image for detection and release unit
        detection_image = img.copy()
        timestamp = unit.getTimestamp()

        unit.release()

        #find all faces on the image and return array of coords
        boxes = self.detector.detect_face(detection_image)
        #remove all faces witch width is lower that threshold
        thresholded_width_indx = []
        for i, box in enumerate(boxes):
            width = box[2] - box[0]
            if width > self.minimum_face_width:
                thresholded_width_indx.append(i)
        selected_boxes = boxes[thresholded_width_indx]

        self.parent.setDetection(selected_boxes,timestamp)
        self.parent.addFrameDect()

    self.detector.terminate()
```

ПРИЛОЖЕНИЕ 4

Реализация метода run класса Recognition Thread

```
def run(self):
   while not self.parent.isTerminated():
      faces = self.parent.getSavedFaces()
      while faces == None: #no faces avaliable
         time.sleep(0.1)
         faces = self.parent.getSavedFaces()

         if self.parent.isTerminated():
            break

      if self.parent.isTerminated():
         break

      validFaces = [f for f in faces if len(f['boxes']) > self.min_detection_count]

      for face in validFaces:
      #get the timestamp of the most recent frame of face
         timestamp = face['timestamps'][-1]
         unit = self.parent.getUnit(self,timestamp)

         if unit is not None:
            img = unit.getFrame()
            #copy image for recognition and release unit
            detection_image = img.copy()

            box = np.mean(face['boxes'],axis=0)

            #Align face
            aligned_face = self.face_aligner.align(detection_image,box)
            #detect age
            faces_input = np.empty((1, self.feature_models_img_size, self.feature_models_img_size, 3))
            faces_input[0, :, :, :] = aligned_face
            faces_input[0, :, :, :] = cv2.normalize(faces_input[0, :, :, :], None, alpha=0, beta=255,
                                    norm_type=cv2.NORM_MINMAX)

            with self.graph_age.as_default():
               age = self.model_age.predict(faces_input)

            with self.graph_gender.as_default():
               gender = self.model_gender.predict(faces_input)

            #update age info
            if 'age' in face.keys():
               face['age'] = self.age_update_ratio * age + (1-self.age_update_ratio) * face['age']
            else:
               face['age'] = age
               face['recognition_round'] = 0

            #update gender info
            if 'gender' in face.keys():
               face['gender'] = self.gender_update_ratio * gender + (1-self.gender_update_ratio) *
face['gender']
            else:
```

```
            face['gender'] = gender

        #update recognition round step
        face['recognition_round'] += 1
if(len(validFaces) > 0):
    self.parent.addFrameRecg()
```