Минобрнауки России

Юго-Западный государственный университет

Кафедра программной инженерии

ОТЧЕТ

о преддипломной (производственной) практике

наименование вида и типа практики

на (в) ООО «МЦОБ. Онлайн-сервисы»

наименование предприятия, организации, учреждения

Студента 4 курса, группы ПО-11б

курса, группы

Боева Антона Владимировича

фамилия, имя, отчество

Руководитель практики от предприятия, организации, учреждения

директор

должность, звание, степень

Куркина А. В.

фамилия и. о.

Оценка _____

_____

подпись, дата

Руководитель практики от университета

к.т.н. доцент

должность, звание, степень

Чаплыгин А. А.

фамилия и. о.

Оценка _____

_____

подпись, дата

Члены комиссии

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

_____ _____

подпись, дата фамилия и. о.

Курск 2025 г.

СОДЕРЖАНИЕ

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

ИС – информационная система.

ИТ – информационные технологии.

КТС – комплекс технических средств.

ПО – программное обеспечение.

РП – рабочий проект.

ТЗ – техническое задание.

ТП – технический проект.

UML (Unified Modelling Language) – язык графического описания для объектного моделирования в области разработки программного обеспечения.

CPU (Central Processing Unit) – центральный процессор.

GPU (Graphics Processing Unit) – специализированный микропроцессор, предназначенный для обработки графики и вывода изображений на экран.

1 Анализ предметной области

1.1 Понятие распознавания лица

Распознавание лиц — это технология, позволяющая идентифицировать или верифицировать личность человека на основе изображения, видеозаписи или других визуальных данных, связанных с лицом. Несмотря на разнообразие технических решений, все системы распознавания лиц стремятся к одной цели — точному сопоставлению биометрических данных с конкретным человеком из базы.[1]

Процесс распознавания можно разделить на четыре основных этапа:

1. Обнаружение лица.
2. Определение ключевых точек лица.
3. Преобразование в цифровой вид.
4. Сопоставление с базой данных.

1.2 История развития систем распознавания лица

Первые научные исследования в области автоматического распознавания лиц были начаты в 1960-х годах. На этом первоначальном этапе процесс автоматизации был реализован лишь частично: человек вручную отмечал ключевые точки лица. Это были центры зрачков, уголки глаз, средняя точка на линии роста волос в верхней части лба и другие. Для этих точек затем вычислялся набор из 20 взаимных расстояний, которые и служили формальным описанием лица.

Такая полуавтоматическая система демонстрировала производительность на уровне до 40 распознанных лиц в час, причем основным ограничивающим фактором выступала скорость работы человека, ответственного за корректную расстановку ключевых точек. Несмотря на низкую производительность по современным меркам, подобные системы отлично подходили для задач идентификации преступников по фотографиям, где требования к скорости обработки были существенно ниже, чем в реальном времени.

В 1970-х годах были предприняты первые попытки полной автоматизации процесса расстановки ключевых точек, однако достигнутые результаты не отличались достаточной надежностью.

В 1987 г. был предложен подход, основанный на представлении изображения лица собственными векторами, полученными из матрицы изображения. Подход получил название eigenface. В основе этого подхода лежало представление черно-белого изображения лица в виде матрицы яркостей пикселей. После центрирования данных путем вычитания среднего значения яркости, матрица преобразовывалась в вектор, для которого строилась матрица ковариаций. Для этой матрицы вычислялся собственный вектор (eigenvectors), причем несколько наибольших по модулю компонент такого вектора использовались в качестве компактного описания лица. Хотя данный алгоритм изначально не был специфически разработан для задач распознавания лиц, а представлял собой общий метод анализа образов, он заложил важные основы для последующего развития более специализированных алгоритмов. В дальнейшем этот подход был адаптирован для выделения и анализа отдельных частей лица: глаз, носа и рта.

В 1990-х годах специально для решения задачи распознавания лиц были собраны первые крупномасштабные базы данных лиц. Эти базы данных содержали тысячи изображений, сделанных в различных условиях освещения и с небольшими вариациями наклона головы. В отличие от ранее использовавшихся фотографий преступников, сделанных в стандартизированных условиях после задержания, новые базы данных позволяли более объективно оценивать качество алгоритмов распознавания. Например, база данных FERET включала более 14 000 изображений 1 200 различных людей и стала стандартным инструментом для сравнительного анализа методов распознавания. В 1990-х годах системы автоматического распознавания лиц начали находить первое коммерческое применение.

В 2001 г. появился быстрый алгоритм для решений задачи детекции – метод Виолы-Джонса Этот метод основывался на использовании интегрального представления изображения, где значение каждого пикселя вычисля-

лось как сумма значений всех пикселей, расположенных левее и выше данного.

Алгоритм применял скользящее окно, которое последовательно проходило по всему изображению с заданным шагом, вычисляя разность сумм пикселей внутри черных и белых областей специальных карт признаков (рис 1.1). Каждая такая карта признаков кодировала определенные характеристики частей лица. Процесс сканирования повторялся многократно с различными размерами окна и разными наборами карт признаков, что позволяло получить комплексное описание изображения в виде набора признаков Хаара.

Для повышения эффективности в алгоритме Виолы-Джонса использовался каскадный принцип вычислений: если на начальных этапах обработки определенной области изображения не обнаруживалось достаточного количества соответствий, дальнейшие вычисления для этой области прекращались. Благодаря такой оптимизации метод достиг высокой скорости работы и получил широкое распространение, в частности, в цифровых фотоаппаратах для реализации функции автоматического обнаружения лиц.

Однако этот подход имел и существенные ограничения: он мог надежно детектировать только фронтальные изображения лиц с наклоном не более 30 градусов и был подвержен относительно высокому уровню ложных срабатываний.

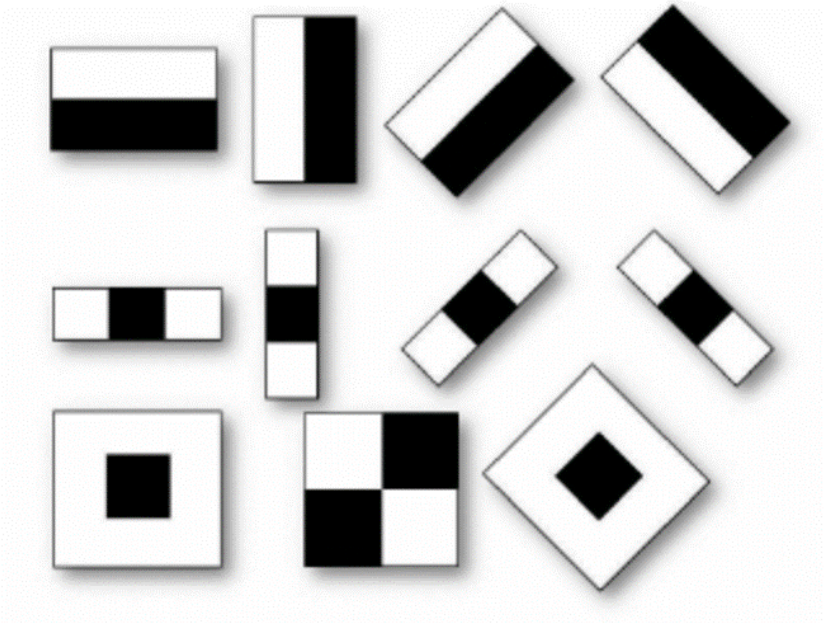

Рисунок 1.1 – Карта признаков Хаара

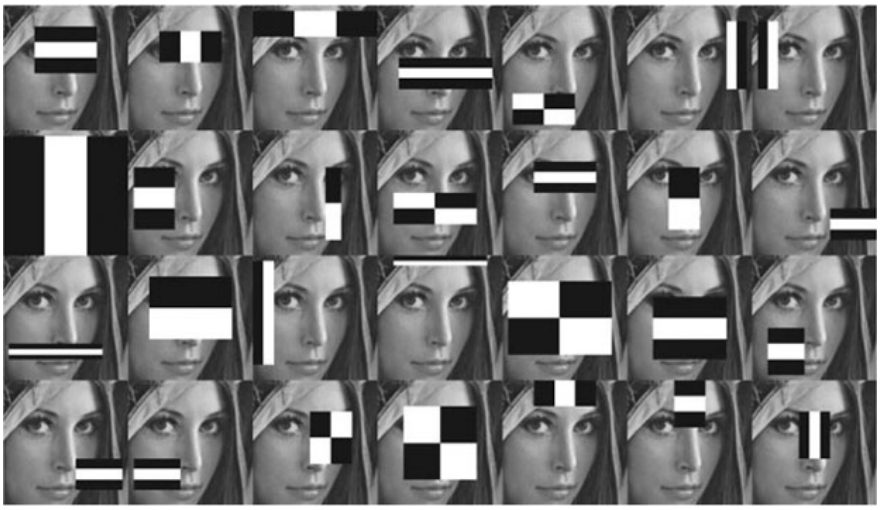

Рисунок 1.2 – Пример детекции частей лица картами признаков Хаара

С начала 2000-х гг. для распознавания лиц используется метод опорных векторов, скрытые марковские модели и начинают использоваться простые свёрточные нейронные сети.

В этот же период появились новые масштабные коллекции изображений лиц, такие как Labeled Faces in the Wild (LFW), содержащие 13 000 изображений 1 680 различных людей, сделанных в естественных условиях с вариациями освещения, выдержки и поз.

В 2007 году был предложен комбинированный подход, сочетающий локальные бинарные шаблоны (LBP) и фильтры Габора. LBP-преобразование эффективно сохраняло информацию о мелких деталях изображения, в то время как фильтры Габора обеспечивали инвариантность к масштабу и сохраняли информацию о общей форме лица. На заключительном этапе применялся метод главных компонент (PCA) для получения компактного описания лица.

Вместе с рядом прорывов в задачах классификации изображений (2012 г. – AlexNet) заметно улучшаются и системы распознавания лиц. Рост вычислительных мощностей, включая широкое использование GPU, сделал возможным обучение глубоких нейронных сетей, в частности свёрточных нейронных сетей (CNN).

Важным преимуществом CNN стало то, что они автоматически обучались оптимальным фильтрам для выделения значимых признаков, избавляя разработчиков от необходимости ручного подбора характеристик. Нейроны в различных слоях сети специализировались на распознавании признаков разного уровня абстракции: от простых геометрических примитивов (линий, углов) на начальных слоях до сложных семантических признаков (глаза, нос, рот) на глубоких слоях.

В 2014 году система DeepFace продемонстрировала точность распознавания 97% на базе данных LFW, впервые достигнув человеческого уровня.

Современные системы на основе глубокого обучения, такие как FaceNet и ArcFace, демонстрируют точность выше 99.9% на стандартных тестовых наборах данных. Они обладают устойчивостью к изменениям освещения, ракурса и частичным перекрытиям лица.[5]

1.3 Выбор метода распознавания лица

Существует множество методов распознавания лиц, различающихся по точности, скорости, устойчивости к внешним условиям и требованиям к ресурсам. Наиболее популярными являются следующие подходы: алгоритм Виолы-Джонса (Haar Cascades), метод гистограмм направленных градиентов (HOG) и многозадачная сверточная нейронная сеть (MTCNN).[4]

1.3.1 Алгоритм Виолы-Джонса (Haar Cascades)

Алгоритм Виолы-Джонса — это один из первых и наиболее известных методов обнаружения объектов на изображениях в реальном времени, особенно лиц. Он был представлен Полом Виолой и Майклом Джонсом в 2001 году и до сих пор используется в различных приложениях компьютерного зрения благодаря своей эффективности и скорости. Он основан на использовании простых признаков — так называемых признаков Хаара, напоминающих вейвлет-фильтры. Метод последовательно применяет каскад классификаторов, обученных с использованием алгоритма AdaBoost. Каждый этап каскада фильтрует изображения, уменьшая количество проверяемых областей.

Основное преимущество алгоритма — высокая скорость распознавания при малом потреблении ресурсов. Однако метод чувствителен к изменению условий: освещению, положению головы, мимике, а также плохо работает с тёмным оттенком кожи и при частичном перекрытии лица. Сегодня данный подход используется преимущественно в простых задачах с контролируемыми условиями.

1.3.2 Гистограмма направленных градиентов (HOG)

Гистограмма направленных градиентов (Histogram of Oriented Gradients) представляет изображение в виде вектора признаков, отражающего ориентацию градиентов яркости в отдельных ячейках изображения. Эти признаки являются устойчивыми к небольшим изменениям освещения и масштабирования. Далее полученный дескриптор подаётся в классификатор, как правило, линейный метод опорных векторов.

HOG был популярен в системах распознавания пешеходов и других объектов с устойчивыми очертаниями. Однако при распознавании лиц метод показывает невысокую точность. Он плохо справляется с поворотами головы и изменениями выражения лица, поскольку чувствителен к геометрическим искажениям. Кроме того, метод имеет высокую вычислительную сложность:

необходимо рассчитать градиенты для каждого пикселя, что замедляет обработку, особенно на слабых устройствах.

1.3.3 Сверточные нейронные сети и MTCNN

Современные системы распознавания лиц преимущественно основаны на нейросетевых архитектурах. Одним из таких решений является многозадачная сверточная нейронная сеть — MTCNN (Multi-task Cascaded Convolutional Neural Network). Данный подход объединяет детекцию лиц и определение ключевых точек (глаз, нос, рта) в рамках единой модели. MTCNN состоит из трёх последовательно работающих сетей: P-Net, R-Net и O-Net, каждая из которых уточняет результаты предыдущей.

Главное преимущество MTCNN — высокая точность и устойчивость к различным условиям: поворот головы, частичное перекрытие, различные оттенки кожи и освещения. Метод может эффективно работать даже с деформированными и профильными изображениями лиц. При наличии графического ускорителя (GPU) обработка осуществляется быстро, что делает этот подход оптимальным выбором для реальных систем биометрической идентификации.

1.4 Проблемы и недостатки современных методов распознавания лица

Современные системы распознавания лиц на основе искусственного интеллекта могут достигать точности более 95%, а некоторые системы достигают 99,5%.[2] Однако это возможно только в лабораторных или строго управляемых условиях: при хорошем освещении, правильном угле обзора, высоком разрешении камер и минимальных помехах. В реальных условиях эксплуатации — в общественных местах, на улице или в транспорте — точность существенно снижается, а риски ошибок возрастают.

На точность распознавания влияют следующие факторы:

1. Освещение и поза. Программы анализа лиц лучше всего работают с изображениями, на которых человек смотрит прямо в камеру. Условия осве-

щения должны быть достаточно яркими, чтобы запечатлеть все черты лица, но не засвечивать их. Положение лица также меняется при повороте головы и зависит от угла обзора наблюдателя.

2. Выражения лица. Нейтральное выражение идеально подходит для точного распознавания. Однако наши эмоции и настроение постоянно меняются, как и мимика. Эти различия меняют внешний вид лица, и для систем распознавания становится трудным его идентифицировать.

3. Старение. Естественные возрастные изменения внешности человека также влияют на способность систем распознавания лиц аутентифицировать людей.

4. Косметологические вмешательства, смена стиля. Изменения во внешности благодаря пластическим операциям, макияжу, смене прически могут негативно отразиться на точности срабатывания алгоритмов распознавания.

5. Аксессуары. Наличие на лице различных объектов (очки, борода и т.д.) или ношение других аксессуаров, таких как головные уборы, шарфы, может серьезно повлиять на работу системы распознавания. Но современные алгоритмы отрабатывают способность видеть сквозь эти препятствия.[2]

В условиях, отличных от идеальных, такие системы все ещё способны допускать серьёзные ошибки. За всю историю развития распознавания лиц в России и мире происходило много крупных скандалов, связанных с использованием автоматических систем биометрической идентификации.

Больше всего скандалов случается, когда системы распознавания лиц путают людей с похожими на них преступниками, вследствие чего людям приходится доказывать свою непричастность перед правоохранительными органами. Один из наиболее известных инцидентов произошёл в Москве, где система распознавания лиц, интегрированная в камеры видеонаблюдения, неверно определила личность прохожего, что привело к его задержанию. Впоследствии выяснилось, что ошибка была вызвана сильным визуальным сходством с разыскиваемым человеком и недостаточной уникальностью

биометрических признаков.[3] В подобных случаях ошибочное распознавание лиц является неприемлимым, а 5% ошибок - слишком большим числом.

2 Техническое задание

2.1 Основание для разработки

Основанием для разработки является задание на выпускную квалификационную работу бакалавра «Интеллектуальная система распознавания на основе изображений лица и отпечатков пальцев с интеграцией карт пропусков».

2.2 Цель и назначение разработки

Интеллектуальная система предназначена для автоматизированного распознавания личности с использованием изображений лица, отпечатков пальцев и пропускных карт. Система разрабатывается для применения в задачах контроля доступа на охраняемые объекты, в учреждениях с ограниченным доступом, а также в корпоративной среде.

Система должна иметь возможность проведения идентификации посредством встроенного модуля биометрической аутентификации, без необходимости ввода паролей или взаимодействия с охранным персоналом.

Задачами данной разработки являются:

– реализация распознавания на основе изображений лица;

– реализация распознавания на основе изображений отпечатков пальца;

– интеграция карт пропуска;

– реализация системы идентификации;

2.3 Требования к программной системе

2.3.1 Требования к данным программной системы

2.3.2 Функциональные требования к интеллектуальной системе

В разрабатываемой интеллектуальной системе должны быть реализованы следующие функции:

– захват изображения лица с видеопотока;

На рисунке 2.1 предоставлены функциональные требования к системе, представленные в виде диаграммы прецедентов.

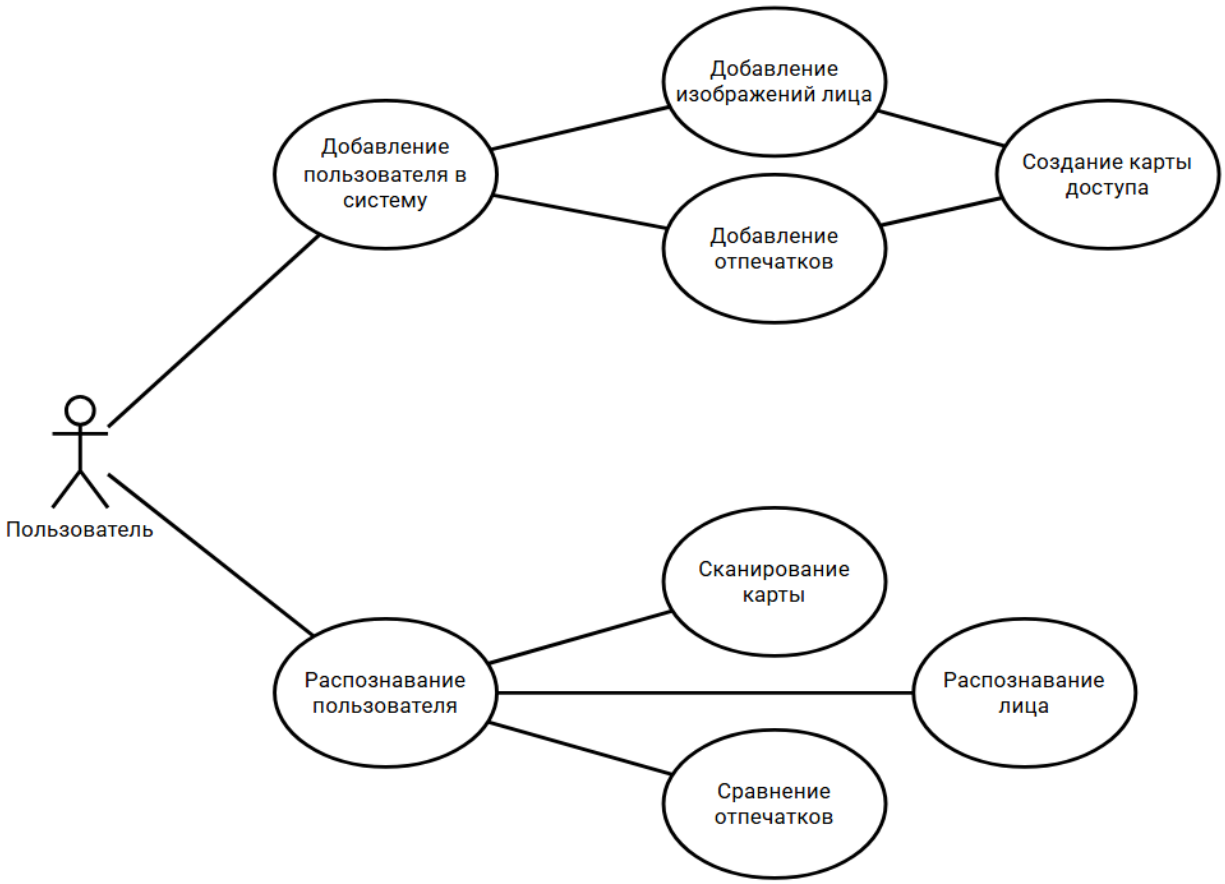

Рисунок 2.1 – Диаграмма прецедентов

2.3.2.1 Сценарий использования «Добавление изображений лица»

Заинтересованные лица и их требования: пользователь желает добавить изображения лица в систему.

Предусловие: программа запущена, выбран режим «Добавить в систему».

Постусловие: программа сохраняет изображения лица пользователя в систему.

Основной успешный сценарий:

1. Пользователь нажимает на кнопку «Добавить в систему».

2. Программа открывает диалоговое окно с указаниями для пользователя.

3. Пользователь закрывает диалоговое окно.

4. Программа запускает сканирования лица пользователя.

5. Пользователь выполняет указания программы.

6. Программа сохраняет изображения пользователя для дальнейшей обработки.

7. Программа открывает диалоговое окно с результатом работы.

8. Пользователь закрывает диалоговое окно и завершает сканирование лица.

2.3.2.2 Сценарий использования «Добавление отпечатков»

Заинтересованные лица и их требования: пользователь желает добавить изображения отпечатка в систему.

Предусловие: программа запущена, выбран режим «Добавить в систему», изображения лица добавлены.

Постусловие: программа сохраняет изображения отпечатка пользователя в систему.

Основной успешный сценарий:

1. Пользователь нажимает на кнопку «Добавить в систему».

2. Программа открывает диалоговое окно выбора файла.

3. Пользователь выбирает файл в формате .jpg, содержащий изображение отпечатка пальца.

4. Программа запускает обучение нейронной сети на основе изображения отпечатка.

5. Программа сохраняет модель нейронной сети для дальнейшей работы.

2.3.2.3 Сценарий использования «Создание карты доступа»

Заинтересованные лица и их требования: пользователь желает создать карту для доступа в систему.

Предусловие: программа запущена, выбран режим «Добавить в систему», изображения лица и отпечатка добавлены.

Постусловие: программа создает карту доступа в систему.

Основной успешный сценарий:

1. Пользователь нажимает на кнопку «Добавить в систему».
2. Программа запускает создание карты доступа.
3. Программа сохраняет карту доступа в систему.

2.3.2.4 Сценарий использования «Сканирование карты»

Заинтересованные лица и их требования: пользователь желает получить доступ в систему.

Предусловие: программа запущена, выбран режим «Распознавание пользователя».

Постусловие: программа переходит к распознаванию лица.

Основной успешный сценарий:

1. Пользователь нажимает на кнопку «Распознавание пользователя».
2. Программа открывает диалоговое окно с указаниями для пользователя.
3. Пользователь закрывает диалоговое окно.
4. Пользователь выполняет указания программы.
5. Программа запускает сканирования карты пользователя.
6. Программа получает данные с карты для дальнейшего распознавания.

2.3.2.5 Сценарий использования «Распознавание лица»

Заинтересованные лица и их требования: пользователь желает получить доступ в систему.

Предусловие: программа запущена, выбран режим «Распознавание пользователя», карта доступа распознана.

Постусловие: программа переходит к сравнению отпечатка.

Основной успешный сценарий:

1. Пользователь нажимает на кнопку «Распознавание пользователя».

2. Программа открывает диалоговое окно с указаниями для пользователя.

3. Пользователь закрывает диалоговое окно.

4. Пользователь выполняет указания программы.

5. Программа запускает сканирования карты пользователя.

6. Программа получает данные с карты для дальнейшего распознавания.

2.3.2.6 Сценарий использования «Сравнение отпечатков»

Заинтересованные лица и их требования: пользователь желает получить доступ в систему.

Предусловие: программа запущена, выбран режим «Распознавание пользователя», карта доступа и лицо распознаны.

Постусловие: программа выдает результат распознавания.

Основной успешный сценарий:

1. Пользователь нажимает на кнопку «Распознавание пользователя».

2. Программа открывает диалоговое окно с указаниями для пользователя.

3. Пользователь закрывает диалоговое окно.

4. Пользователь выполняет указания программы.

5. Программа запускает сканирования карты пользователя.

6. Программа получает данные с карты для дальнейшего распознавания.

2.3.3 Требования пользователя к интерфейсу приложения

Приложение должно иметь следующие экраны:

1. Экран «Идентификация». Основной экран, реализующий функционал распознавания пользователя на основе изображений лица и отпечатка пальца.

2. Экран «Добавление в систему». Экран, позволяющий добавлять пользователя в систему, путём сохранения изображений лица и отпечатков пальца с генерацией карты доступа.

2.3.4 Нефункциональные требования к программной системе

2.3.4.1 Требования к надежности

Программная система должна обеспечивать стабильную работу в различных условиях эксплуатации. В процессе работы приложения могут возникнуть следующие аварийные ситуации:

1. отсутствие подключения к камере для получения изображений;
2. ошибка доступа к файлам изображения;

Для предотвращения аварийных ситуаций программа должна корректно обрабатывать исключения при работе с файлами, предоставляя пользователям информативные сообщения об ошибках. В случае проблем с отсутствием прав доступа к директории сохранения файлов, полученных в результате работы программы, программа должна открывать диалоговое окно с выбором другой директории.

2.3.4.2 Требования к программному обеспечению

2.3.4.3 Требования к аппаратному обеспечению

2.4 Требования к оформлению документации

Разработка программной документации и программного изделия должна производиться согласно ГОСТ 19.102-77 и ГОСТ 34.601-90. Единая система программной документации.

Программная документация должна включать в себя:

– анализ предметной области;
– техническое задания;
– технический проект;
– рабочий проект.

3 Технический проект

3.1 Общая характеристика организации решения задачи

Необходимо спроектировать и разработать систему, которая должна выполнять распознавание на основе изображений лиц и отпечатков пальцев с использованием карт пропуска.

Система представляет собой программу, позволяющую пользователю выполнять распознавание лиц и отпечатков пальцев и сканирование карты пропуска для авторизации. Интерфейс включает в себя окно, отображающее видеопоток с камеры и результаты распознавания лица или отпечатка пальца и карты пропуска. Программа автоматически определяет лицо, отпечаток пальца или карту пропуска в кадре, сопоставляет данные и отображает результаты выполнения программы на экране.

3.2 Обоснование выбора технологии проектирования

Используемые для создания интеллектуальной системы языки программирования и технологии соответствуют современным практикам разработки, обеспечивают высокую производительность и отказоустойчивость системы.

3.2.1 Описание используемых технологий и языков программирования

В процессе разработки программы используется язык программирования Python, а также сторонние библиотеки OpenCV, PyTorch, NumPy, tkinter и Python Imaging Library.

3.2.2 Язык программирования Python

Язык программирования Python представляет собой высокоуровневый язык общего назначения, отличающийся лаконичным синтаксисом и высокой читаемостью кода. Python обеспечивает возможность реализации объектно-ориентированного, функционального и процедурного стилей программиро-

вания. Он обладает мощной стандартной библиотекой, которая охватывает множество областей применения — от работы с файлами и сетями до многопоточности и регулярных выражений. Python широко используется в научных вычислениях, благодаря таким библиотекам как NumPy, SciPy, Pandas и Matplotlib. В области машинного обучения и нейросетей Python является наилучшим языком программирования, благодаря таким инструментам как PyTorch, TensorFlow и Scikit-learn. Кроме того, язык поддерживается большим сообществом, что облегчает поиск решений и развитие проектов.

3.2.3 Библиотека OpenCV

OpenCV представляет собой мощную библиотеку компьютерного зрения и обработки изображений, предназначенную для выполнения широкого спектра задач, связанных с анализом изображений и видео. Она предоставляет высокоуровневые и низкоуровневые средства для работы с изображениями, включая функции для обработки, фильтрации, распозна-вания объектов и анализа движения. Основное преимущество OpenCV заключается в её высокой производительности, гибкости и поддержке широкого спектра алгоритмов компьютерного зрения. Благодаря интеграции с языком Python, OpenCV позволяет быстро разрабатывать приложения, связанные с обработкой изображений и видео, и является популярным инструментом как для учебных, так и для полноценных исследовательских и промышленных проектов.

3.2.4 Библиотека PyTorch

PyTorch — это современная библиотека машинного и глубокого обучения, предназначенная для создания и обучения нейронных сетей. Она предоставляет интуитивно понятные инструменты для работы с тензорами, автоматического дифференцирования и построения моделей. Одним из ключевых преимуществ PyTorch является использование динамической вычислительной графики, что делает разработку и отладку нейросетей более гибкой и наглядной. Библиотека поддерживает ускорение вычислений с помощью графических процессоров (GPU), что существенно повышает производитель-

ность при обучении моделей. Благодаря тесной интеграции с Python, а также множеству встроенных модулей и готовых архитектур, PyTorch широко применяется в задачах компьютерного зрения, обработки естественного языка, биометрии и других областях искусственного интеллекта.

3.2.5 Библиотека NumPy

NumPy представляет собой фундаментальную библиотеку для численных вычислений в Python, обеспечивающую поддержку многомерных массивов и высокоуровневых математических операций над ними. Благодаря высокоэффективным операциям с массивами и встроенным линейным алгебраическим методам, NumPy позволяет обрабатывать изображения больших размеров с минимальными затратами ресурсов, что делает её незаменимой в задачах, связанных с сжатием, декомпрессией и преобразованием графических данных.

3.2.6 Библиотека tkinter

Для построения графического интерфейса в проекте выбрана библиотека tkinter, которая является частью стандартной поставки Python и представляет собой обёртку над библиотекой Tcl/Tk. Этот выбор объясняется рядом технологических и практических преимуществ:

1. Отсутствие необходимости в установке сторонних зависимостей делает tkinter особенно удобным для использования в проектах с упрощённым процессом развёртывания. Пользователь может сразу запустить приложение без дополнительных установок, что критично в условиях учебной и демонстрационной среды.

2. Простота проектирования интерфейса позволяет легко создавать окна, формы, кнопки, меню, вкладки и другие элементы без необходимости изучения сложных графических фреймворков. Это способствует быстрому созданию прототипов и конечных версий интерфейса.

3. Полная кроссплатформенность интерфейса гарантирует его одинаковое отображение и поведение на всех операционных системах, что избавля-

ет разработчика от необходимости писать отдельные реализации под разные платформы.

4. Наличие компонентов высокого уровня, таких как текстовые поля, выпадающие списки, кнопки, панели и таблицы, даёт возможность создать функциональный и интуитивно понятный интерфейс для работы с данными.

5. Возможность динамического обновления интерфейса обеспечивает удобную визуализацию изменений.

tkinter полностью удовлетворяет требованиям к простому, лёгкому в использовании, но функциональному графическому интерфейсу, особенно в условиях ограниченного времени и ресурсов.

3.2.7 Python Imaging Library

Python Imaging Library, сокращенно PIL — это библиотека для работы с растровыми изображениями, обеспечивающая поддержку различных форматов, таких как JPEG, PNG, BMP и других. PIL предоставляет широкий набор функций для загрузки, сохранения, обработки и преобразования изображений, что упрощает различные взаимодействия с файлами изображений, такие как сохранение и загрузка изображений, поэтому она является удобным инструментом для предварительной и финальной обработки графических данных.

3.3 Архитектура программной системы

Точкой входа в программу является модуль main.py, в котором создаётся и запускается экземпляр класса App, реализующего основной графический интерфейс приложения. Архитектура программы ориентирована на модульность и разделение ответственности, что облегчает поддержку, тестирование и возможное расширение проекта.

Архитектура приложения состоит из следующих ключевых модулей:

1. programm.py — стартовый модуль, в котором производится инициализация пользовательского интерфейса. Здесь создаётся объект MainWindow,

который затем запускается методом mainloop(). Содержит в себе код основных классов приложения(App и LoadingScreen).

2. func.py — содержит функции детекции лиц, извлечения эмбеддингов, сравнения лиц и работы с моделями для распознавания. Модуль отвечает за большинство вычислений в системе.

3. model.py — реализует классы FingerprintEncoder и SiameseNetwork, предназначенные для обработки изображений отпечатков пальцев. Класс FingerprintEncoder — это нейронная сеть, основанная на архитектуре ResNet-18, модифицированная для извлечения эмбеддингов изображений отпечатков пальцев. Класс SiameseNetwork — это основная модель, использующая подход сиамской нейронной сети, где сравниваются два изображения. Эта сеть обучается таким образом, чтобы оценить, являются ли два изображения схожими или нет.

4. dataset.py — создает пользовательскую выборку для обучения модели с использованием пар изображений отпечатков пальцев. Модуль создает как положительные, так и отрицательные пары изображений, которые используются для обучения сиамской сети.

5. train.py — представляет собой процесс обучения модели для сравнения отпечатков пальцев с использованием сиамской нейронной сети и последующего сохранения обученной модели.

6. fp.py — содержит реализацию обработки векторных представлений в классе Fingerprints. Класс Fingerprints используется для сравнения двух изображений отпечатков пальцев и оценки их схожести. Система использует модель Siamese Network, обученную для нахождения эмбеддингов (векторных представлений) изображений отпечатков пальцев, и затем вычисляет схожесть между этими эмбеддингами с использованием косинусного сходства.

7. save_face.py — захват изображений лиц с камеры и их сохранения в определенную директорию для дальнейшей тренировки модели распознавания лиц.

8. embedding_create.py — используется для извлечения векторных представлений лиц из изображений и сохранения их для последующего распознавания личности по лицу.

Компоненты программы взаимодействуют следующим образом:

1. Интерфейс (класс App из файла programm.py) получает данные пользователя, например, запрос на распознавание лица или отпечатка пальца, и передаёт их на обработку соответствующим функциям.

2. Модуль func.py обрабатывает эти команды: выполняет детекцию лиц, извлекает эмбеддинги с помощью нейросетевых моделей, сравнивает их с ранее сохранёнными представлениями и возвращает результат.

3. Результаты распознавания отображаются в пользовательском интерфейсе. При этом данные могут быть изменены или обновлены в зависимости от действий пользователя.

4. Модули save_face.py и embedding_create.py используются для подготовки базы данных лиц: save_face.py захватывает изображения с камеры, а embedding_create.py извлекает и сохраняет их эмбеддинги.

5. Для распознавания отпечатков пальцев используется модель, реализованная в файле model.py и обученная с помощью модуля train.py. Сравнение векторных представлений отпечатков выполняется в модуле fp.py на основе косинусного расстояния.

6. Все изображения и векторные представления хранятся в структуре каталогов database/, что позволяет системе работать автономно, без внешней базы данных.

Между компонентами поддерживается минимальная необходимая связь: графический интерфейс не содержит логики обработки выражений, а логика ядра независима от визуального представления. Это упрощает поддержку, тестирование и адаптацию системы для различных платформ.

Архитектура гибкая, расширяемая и легко масштабируемая. Возможность добавления новых типов команд или подключения внешних источников данных или API без необходимости переписывания основной логики делает систему адаптируемой к разнообразным задачам.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Большаков И. А. Системы распознавания лиц – принцип работы и сферы применения / И. А. Большаков, С. А. Микаева. – Текст : непосредственный // Науносфера. – 2022. – № 9-2. – С. 95–98.

2. Абдуллаев А. И. Распознавание лиц по изображению лица / А. И. Абдуллаев. – Текст : непосредственный // Мировая наука. – 2021. – № 4 (49). – С. 44–47.

3. Кандидата наук Ермошина задержали как вора из-за ошибки распознавания лиц [Электронный ресурс] // Московский комсомолец. – 2021. – 19 октября. – URL: https://www.mk.ru/incident/2021/10/19/kandidata-nauk-ermoshina-zaderzhali-kak-vora-izza-oshibki-raspoznavaniya-lic.html?utm_source=yxnews&utm_medium=desktop (дата обращения: 03.05.2025).

4. Байкенов Б. С. Сравнительный анализ методов распознавания объектов / Б. С. Байкенов, Р. С. Тынчеров, А. Р. Фазылова. – Текст : непосредственный // Вестник Казахской академии транспорта и коммуникаций им. М. Тынышпаева. – 2019. – № 1 (108). – С. 185–191.

5. Ветров С. В. История развития систем распознавания лиц / С. В. Ветров. – Текст : непосредственный // Наука и образование: актуальные исследования и разработки : материалы IV Всероссийской научно-практической конференции / отв. ред. А. В. Лесков. – Чита : Забайкальский государственный университет, 2021. – С. 23–29.

6. Джеймс, Р. UML 2.0. Объектно-ориентированное моделирование и разработка / Р. Джеймс, Б. Майкл.– 2-е изд.– Санкт-Петербург : Питер, 2021.– 542 с.– ISBN 978-5-4461-9428-5.– Текст : непосредственный.

7. Биэль, М. RESTful API Design / М. Биэль.– University Press, 2016. 300 с.– ISBN 978-1-5147-3516-9.– Текст : непосредственный.

8. Аттуи, А. Real-Time and Multi-Agent Systems / А. Аттуи.– Springer Science & Business Media, 2000.– 496 с.– ISBN 978-1-85233-252-5.– Текст : непосредственный.

9. Буч, Г. Введение в UML от создателей языка / Г. Буч, И. Якобсон, Д. Рамбо.– Москва : ДМК Пресс, 2015.– 498 с.– ISBN 978-5-457-43379-3. Текст : непосредственный.

10. Мандел, Т. Разработка пользовательского интерфейса / Т. Мандел.– ДМК Пресс, 2019.– 420 с.– ISBN 978-5-04-195060-6.– Текст : непосредственный.

11. 5. Мартин, Р. Чистая архитектура. Искусство разработки программного обеспечения / Р. Мартин; пер. с англ. А. Кисилева. – Санкт-Петербург: Питер, 2018. – 351 с. – ISBN 978-5-4461-0772-8. – Текст: непосредственный.