Министерство науки и высшего образования Российской Федерации ФГБОУ ВО РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ ГИДРОМЕТЕОРОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ (РГГМУ)

Институт Информационных систем и геотехнологий КАФЕДРА ПРИКЛАДНОЙ ИНФОРМАТИКИ

БАКАЛАВРСКАЯ РАБОТА

На тему “Нейронная сеть по распознаванию лиц”

Исполнитель Бабкин Михаил Сергеевич

Руководитель кандидат технических наук, доцент - Яготинцева Наталья Владимировна

«К защите допускаю»

Заведующий кафедрой _____ (подпись)

_____

_____ (ученая степень, ученое звание)

_____

_____ (фамилия, имя, отчество)

«___»_____ 2022 г.

Санкт–Петербург

2023

Оглавление

Введение

Нейронные сети становятся все больше популярны в 21 веке, ведь нейронные сети позволяют делать работу за человека (конкретно в нашей работе распознавание изображения человека) соответственно, упрощая деятельность, как и малых, так и средних предприятий.

Посредством нейронных сетей можно упростить пропускную систему предприятий. Например, в некоторых фитнес-залах внедрена эта система и вместо пропускных карточек, люди предоставляют свои биометрические данные для входа, а это существенно облегчает работу предприятия, ведь биометрические данные сложно подделать и не нужно будет для клиентам/сотрудникам делать специальные карты для входа. Результаты работы будут использованы для упрощения работы предприятия, путем интегрирования нашего проекта по нейронным сетям. Распознавание человека по изображению лица выделяется среди биометрических систем тем, что не требует специального дорогостоящего оборудования.

Нейронная сеть - это компьютерная модель, вдохновленная биологической нервной системой, которая используется для обработки информации и выполнения различных задач машинного обучения. Она состоит из соединенных и взаимодействующих между собой искусственных нейронов, которые обрабатывают и передают сигналы через сеть.

Каждый искусственный нейрон в нейронной сети получает входные данные, обрабатывает их и передает результат следующему нейрону в сети. Искусственные нейроны обычно имеют несколько входных соединений, которые взвешиваются определенными весами. Эти веса позволяют настраивать важность каждого входного сигнала для вычислений. После взвешивания сигналы суммируются, и затем проходят через функцию активации, которая определяет, будет ли активирован нейрон и передаст ли он свой выходной сигнал следующему нейрону.

Нейронные сети могут иметь различные архитектуры, включая простые однослойные сети, многослойные перцептроны и глубокие нейронные сети с множеством скрытых слоев. Они используются для решения широкого спектра задач, таких как классификация изображений, распознавание речи, анализ текстов, прогнозирование временных рядов и многое другое.

Обучение нейронных сетей происходит путем подачи на вход сети набора обучающих данных, состоящего из входных сигналов и соответствующих им целевых выходных значений. С помощью методов обучения нейронная сеть настраивает свои веса и параметры таким образом, чтобы минимизировать ошибку между выходными значениями, предсказанными сетью, и целевыми значениями. После обучения нейронная сеть может быть использована для выполнения предсказаний на новых данных, которые ранее не были использованы в процессе обучения.

Объектом исследования является пропускной процесс. Предметом исследования являются все малые и средние предприятия, у которых присутствует пропускная система. Анализируя формат пропускных систем многих предприятиях можно предложить внедрить проект по распознаванию лиц при помощи нейронных данных. Это позволит обезопасить предприятие от несанкционированного доступа, ускорить процесс пропуска сотрудников на предприятие, сократить расход на содержание сотрудников в отделе режима. На все вышеперечисленные предприятия, у которых существует пропуск, например, по карточкам, можно внедрить данный проект и упростить работу действующего предприятия.

Целью ВКР является анализ такого бизнес-процесса, как пропускная система, его оптимизация: Посредством внедрения системы по распознаванию лиц по нейронным сетям, то можно увеличить скорость прохождения сотрудников тем самым уменьшить риск опоздания сотрудников. Внедрение данной системы позволит предприятию не только увеличить свою безопасность, но и уменьшить взаимодействие людей друг с другом, что становится более важной задачей, ведь отсутствие контакта людей с людьми становится все более актуально в наши дни.

Задачами ВКР являются:

1. Разработка структуры сверточной нейросети
2. Обучение нейросети, на размеченной базе изображений
3. Тестирование обученной сети
4. Моделирование процессов

В процессе написания выпускной квалификационной работы на тему “Нейронная сеть по распознаванию лиц” были использованы следующие методы:

1. Сравнительный анализ
2. Многоаспектное моделирование
3. Анализ и моделирование в нотациях idef
4. Технологии проектирования баз данных и программных комплексов
5. Технологии проектирования системно-аппаратных сред
6. Стандарты управления проектами pmbok, swebok, msf.

Проводим сравнительный анализ. Система по распознаванию лиц по нейронным сетям, в отличии от других систем, использует пять точек-якорей (глаза, нос и рот). По этим точкам строится граф, рассчитывающий расстояния, и определяющийся, как отдельно взятое лицо. На данный момент для проектирования программных систем используются следующие методы:

1. BPMN

2. Построение блок-схем
3. Создание ER-диаграмм
4. UML-диаграммы
5. Разработка макетов и математических моделей
6. Метод обучения нейронной сети
7. Метод сравнения двух изображений
8. Построение AS-IS моделей

Для создания BPMN моделей использовались программы такие как:

- ARIS Express
- Lucidchart
- Draw.io

При помощи вышеописанных диаграмм, удалось провести анализ предметной области. По UML-диаграммам мы можем понять: как устроена система вживую, понять какие функции выполняют важные роли в этой системе, увидеть ее на схеме. Посредством применения методологии idef, была построена модель как есть, что позволило понять как устроены бизнес-процессы данной системы и какие нужны ресурсы для этих них. Благодаря методу обучения нейронной сети, она сможет самообучаться. *(Дописать)*

В качестве основного метода для проектирования программной системы были выбраны UML-диаграммы из-за возможности рассмотреть проектируемую систему с разных сторон, а в качестве основного программного средства — Draw.io. Также язык программирования для разработки интерфейса используется Python.

ГЛАВА 1 АНАЛИТИЧЕСКАЯ ЧАСТЬ

1.1. Анализ Предметной области

Искусственный интеллект (ИИ) представляет собой способность автоматических систем выполнять определенные функции, которые обычно связываются с интеллектом человека. Он способен анализировать внешнюю среду и принимать оптимальные решения на основе полученного опыта. ИИ также может разрабатывать алгоритмы действий и оптимизировать процессы. В основном, ИИ имитирует интеллектуальную деятельность человека.

Часто мы связываем искусственный интеллект с научными лабораториями и IT-офисами, однако на самом деле мы ежедневно пользуемся продуктами ИИ без осознания этого факта. Например, навигационные системы, такие как GPS, помогают нам выбирать оптимальные маршруты с учетом пробок, дорожных знаков и предпочтений в передвижении. Голосовые помощники, такие как Алиса или Siri на смартфонах, помимо выполнения поисковых запросов, способны ставить напоминания, играть мелодии и общаться на базовом уровне. Даже подбор авиабилетов осуществляется с помощью анализа данных, собранных в прошлом.

Искусственный интеллект давно перестал быть фантастической идеей и стал частью нашей повседневной жизни. Он широко применяется в различных отраслях бизнеса и на разных уровнях производства. На этапе проектирования, ИИ помогает повысить эффективность разработки новых продуктов, автоматизирует выбор и оценку поставщиков, анализирует требования к запчастям и деталям.

В контексте данного проекта, рассматривается использование нейросети, способной распознавать лица, как пример применения искусственного интеллекта в производственной сфере.

Процессы:

1)Получения информации;

2)Распознавания информации;

3)Обучение нейронной сети;

4)Запись информации;

5)Запись процентного совпадения.

Виды нейронных сетей

Существует несколько различных видов нейронных сетей, каждый из которых имеет свою уникальную архитектуру и предназначение. Нужно подойти ответственно к выбору нейронной сети.

Определение подходящего вида нейронной сети для конкретной задачи является важным шагом в разработке и решении задач машинного обучения. Выбор правильного вида нейронной сети может существенно повлиять на эффективность и точность модели.

Нейронные сети представляют собой алгоритмы машинного обучения, которые пытаются эмулировать работу человеческого мозга. Они состоят из множества взаимосвязанных нейронов, которые передают и обрабатывают информацию. Каждый вид нейронной сети имеет свою структуру и специфичные алгоритмы обучения.

Важность выбора подходящего вида нейронной сети связана с тем, что разные задачи требуют разных архитектур и подходов. Например, для задач классификации изображений хорошо подходят свёрточные нейронные сети (CNN), которые специально разработаны для обработки двумерных входных данных, таких как изображения. Для обработки последовательных данных, таких как текст или речь, хорошо подходят рекуррентные нейронные сети (RNN) или их вариации, например, LSTM или GRU. Сейчас рассмотрим наиболее распространенный виды нейронных сетей.

1)Прямое распространение (Feedforward) нейронные сети: Это самый простой и широко используемый тип нейронных сетей. В прямом распространении информация перемещается только в одном направлении, от входов к выходам. Они состоят из слоев нейронов, где каждый нейрон связан с нейронами в следующем слое. Прямое распространение нейронных сетей эффективно работают с структурированными данными, такими как числовые значения или фиксированный размер изображений. Многослойные перцептроны (MLP) являются примером прямого распространения нейронных сетей.

2)Рекуррентные нейронные сети (Recurrent Neural Networks, RNN): Рекуррентные нейронные сети обрабатывают последовательные данные, такие как временные ряды, текст или речь, где текущий вход зависит от предыдущих. Они имеют обратные связи, позволяющие информации перемещаться в обратном направлении, что делает их более подходящими для моделирования долгосрочных зависимостей. Однако обычные RNN могут столкнуться с проблемой затухающего градиента при обучении длинных последовательностей. Долгая краткосрочная память (LSTM) и GRU (Gated Recurrent Unit) - это расширения RNN, которые помогают справиться с этой проблемой и обрабатывать долгосрочные зависимости.

3)Свёрточные нейронные сети (Convolutional Neural Networks, CNN): Свёрточные нейронные сети широко применяются в обработке и анализе изображений, но также могут использоваться для других типов данных, имеющих пространственную структуру. Они используют операцию свертки для обнаружения локальных шаблонов или признаков во входных данных. Свёрточные слои нейронных сетей изучают иерархические представления изображений, начиная с низкоуровневых при

знаков, таких как границы и текстуры, и двигаясь к более высокоуровневым признакам, таким как формы и объекты. CNN также используют слои

подвыборки для уменьшения размерности данных и полносвязные слои для классификации или регрессии.

4)Глубокие автокодировщики (Deep Autoencoders): Глубокие автокодировщики - это нейронные сети, которые обучаются без учителя для извлечения наиболее важных признаков из входных данных. Они состоят из кодировщика и декодера. Кодировщик преобразует входные данные в более компактное представление (скрытое пространство), а декодер пытается восстановить исходные данные из этого представления. Глубокие автокодировщики широко используются для сжатия данных, предобучения или изучения скрытых структур в данных. Они также могут использоваться для генерации новых данных, когда кодировщик и декодер настроены на разные наборы данных.

5)Генеративно-состязательные сети (Generative Adversarial Networks, GAN): GAN состоят из двух моделей: генератора и дискриминатора, которые соревнуются друг с другом. Генератор создает новые образцы данных, пытаясь обмануть дискриминатор, а дискриминатор пытается отличить сгенерированные образцы от реальных данных. Обучение GAN приводит к тому, что генератор способен генерировать все более реалистичные данные. GAN широко используются для генерации изображений, текста, звука и других типов данных, и они позволяют создавать новые синтезированные данные, которые могут быть использованы в различных приложениях, включая искусство, дизайн и улучшение данных.

В дипломной работе выбор остановился именно на свёрточной нейронной сети, сейчас обозначу ряд преимуществ, которыми обладает данный вид системы и которые подходят под нашу задачу.

1)Специализация на обработке изображений: CNN были разработаны специально для работы с двумерными входными данными, такими как изображения. Они обладают способностью автоматически извлекать иерархические признаки из изображений, начиная с простых геометрических форм и текстур, и заканчивая более сложными объектами и контекстом.

2)Инвариантность к пространственным трансформациям: Сверточные слои в CNN имеют свойство инвариантности к пространственным трансформациям, таким как сдвиг, масштабирование и вращение. Это позволяет нейронным сетям обнаруживать лица независимо от их положения, размера и ориентации в изображении.

3)Параметрическая эффективность: CNN обладают параметрической эффективностью благодаря своей архитектуре. Они используют разделяемые веса (shared weights) и сверточные операции, что позволяет значительно уменьшить количество обучаемых параметров. Это особенно полезно при работе с большими наборами данных и упрощает процесс обучения.

4)Автоматическое извлечение признаков: CNN обучаются автоматически извлекать наиболее информативные признаки из изображений для задачи классификации лиц. Это особенно полезно при распознавании лиц, где важными признаками могут быть форма лица, расположение глаз, носа и рта, текстуры кожи и т. д. CNN позволяют извлекать эти признаки без необходимости их ручного определения.

5)Составные признаки: CNN имеют возможность обнаруживать составные признаки, объединяя множество более простых признаков в более высокоуровневые представления. Это позволяет моделям CNN учитывать

сложные зависимости между различными частями лица и делать более точные предсказания.

6)Примеры успешного применения: CNN уже продемонстрировали впечатляющую эффективность в задачах распознавания лиц. Они использовались в широком спектре приложений, включая системы безопасности, системы видеонаблюдения, социальные сети, автоматическую идентификацию и многое другое.

В целом, сверточные нейронные сети обладают мощными возможностями для распознавания лиц, позволяя автоматически извлекать иерархические признаки из изображений и делать точные предсказания независимо от вариаций внешних условий. Это делает их предпочтительным выбором для задач распознавания лиц.

Применение

Нейронные сети являются мощным инструментом, который нашел широкое применение в различных областях благодаря своим уникальным способностям. Одна из главных причин их популярности заключается в их способности обрабатывать сложные данные и решать сложные задачи.

Нейронные сети обладают способностью обрабатывать большие объемы данных и извлекать из них полезную информацию. Они способны анализировать и классифицировать данные, выделять особенности и закономерности, а также делать предсказания и прогнозы на основе имеющихся данных. Вот некоторые из основных областей, где нейронные сети активно применяются:

1)Компьютерное зрение: Нейронные сети играют важную роль в обработке и анализе изображений и видео. Они используются для задач распознавания и классификации объектов, детектирования лиц, сегментации изображений, распознавания образов, анализа и интерпретации содержимого изображений. Примеры включают автоматическую классификацию фотографий, распознавание лиц в системах безопасности, анализ медицинских изображений и т.д.

2)Обработка естественного языка: Нейронные сети применяются для обработки и анализа текстовых данных, включая задачи классификации текста, машинного перевода, анализа тональности, генерации текста и ответов, распознавания именованных сущностей и многое другое. Примеры включают голосовых помощников, автоматический перевод текстов, системы автоматического анализа текстовых данных и др.

3)Рекомендательные системы: Нейронные сети используются для построения рекомендаций на основе анализа пользовательских предпочтений и поведения. Они позволяют предлагать персонализированные рекомендации в различных сферах, таких как фильмы, музыка, товары, новости и т.д.

Примеры включают системы рекомендаций на основе просмотров фильмов или покупок товаров.

4)Автоматическое управление и робототехника: Нейронные сети применяются в системах автоматического управления и робототехнике для принятия решений, планирования движений, обработки сенсорных данных и обучения роботов. Они используются для разработки самоуправляемых автомобилей, роботов-помощников, систем автопилотирования и многих других приложений.

5)Финансовые прогнозы и анализ данных: Нейронные сети применяются в финансовой сфере для прогнозирования цен акций, анализа финансовых данных, автоматического торговли и рискового управления. Они помогают в принятии решений на основе исторических данных и обнаружении тенденций.

6)Медицинская диагностика и обработка данных: Нейронные сети используются в медицине для диагностики болезней, анализа медицинских изображений, прогнозирования результатов лечения, анализа генетических данных и многое другое. Они помогают в автоматическом выявлении патологий, классификации изображений, поддержке принятия решений врачами и т.д.

7)Промышленность и производство: Нейронные сети применяются в промышленности для контроля качества продукции, прогнозирования отказов оборудования, оптимизации процессов производства и т.д. Они позволяют автоматизировать и оптимизировать производственные операции и улучшить эффективность.

Это лишь некоторые из множества областей, в которых нейронные сети находят применение. Они продолжают активно развиваться и находить новые области применения, превосходя ожидания в обработке и анализе данных.

Алгоритмы

Стоит отметить одну из немало важных частей нейронной системы как её алгоритмы. Алгоритмы распознавания являются методами и подходами к обработке данных и решению задач распознавания. Это конкретные процедуры и шаги, которые применяются для извлечения признаков из данных, классификации объектов и принятия решений на основе полученных результатов. Алгоритмы могут быть использованы внутри конкретного вида нейронной сети или в других методах машинного обучения.

1)Прямое распространение (Feedforward): Это основной алгоритм для применения нейронных сетей на входных данных. В прямом распространении данные проходят через слои нейронной сети от входа к выходу без обратных связей. Каждый нейрон в слое вычисляет свою активацию на основе входных данных и весов, и передает активацию следующему слою.

2)Функции активации: Функции активации определяют выходное значение каждого нейрона в нейронной сети. Некоторые популярные функции активации включают сигмоидную функцию, гиперболический тангенс, ReLU (Rectified Linear Unit) и softmax. Они добавляют нелинейность в нейронные сети, позволяя моделировать сложные зависимости в данных.

3)Алгоритм обратного распространения (Backpropagation): Хотя алгоритм обратного распространения обычно ассоциируется с обучением нейронных сетей, он также может быть использован для оценки градиента во время тестирования или инференса. Это позволяет анализировать влияние входных данных на выходы сети и выполнять различные операции, такие как визуализация активаций или поиск важных признаков.

4)Проход по ансамблю (Ensemble Forwarding): Этот алгоритм используется для комбинирования предсказаний нескольких нейронных сетей, образуя ансамбль моделей. Каждая нейронная сеть может иметь свои собственные веса и архитектуру, и входные данные проходят через каждую

сеть, а затем совместные предсказания используются для принятия окончательного решения. Ансамблирование позволяет повысить точность и надежность модели.

5)Метод оптимизации (Optimization Algorithms): Это алгоритмы, которые применяются для настройки весов и параметров нейронных сетей в процессе обучения или оптимизации. Некоторые популярные методы оптимизации включают градиентный спуск, стохастический град

6)Алгоритм обратного распространения ошибки (Backpropagation): Это основной алгоритм для обучения нейронных сетей. В процессе обучения он вычисляет ошибку между выходными значениями сети и ожидаемыми значениями и распространяет эту ошибку обратно через сеть, корректируя веса нейронов. Алгоритм обратного распространения ошибки использует градиентный спуск для поиска оптимальных весов, минимизирующих ошибку.

7)Стохастический градиентный спуск (Stochastic Gradient Descent, SGD): Это оптимизационный алгоритм, который применяется для обновления весов во время обучения нейронных сетей. Вместо вычисления градиента по всей обучающей выборке, SGD вычисляет градиент и обновляет веса по одному образцу за раз. Это делает его более эффективным и подходящим для больших наборов данных.

8)Градиентный спуск с импульсом (Momentum Gradient Descent): Это улучшенный вариант градиентного спуска, который учитывает предыдущие изменения весов при обновлении. Он добавляет импульс, который помогает преодолеть локальные минимумы и ускоряет сходимость обучения.

9)Алгоритм адаптивного шага обучения (Adaptive Learning Rate): Эти алгоритмы позволяют динамически регулировать скорость обучения в зависимости от изменений в процессе обучения. Некоторые популярные алгоритмы адаптивного шага обучения включают AdaGrad, RMSprop и Adam.

10)Dropout: Dropout является техникой регуляризации, которая применяется во время обучения нейронных сетей. Он случайным образом отключает некоторые нейроны во время прохода по обучающим данным. Это помогает предотвратить переобучение и улучшает обобщающую способность модели.

11)Сверточные и пулинг слои: Эти алгоритмы используются в сверточных нейронных сетях для извлечения локальных признаков из входных данных. Сверточные слои применяют фильтры для выделения особенностей, а пулинг слои уменьшают размерность данных, сохраняя важные информационные признаки.

Это лишь некоторые примеры алгоритмов, используемых в нейронных сетях. Различные алгоритмы и их комбинации могут применяться в зависимости от конкретной задачи и требований модели.

Преимущества

Нейронные сети являются одной из самых мощных и востребованных технологий в области искусственного интеллекта. Они имеют множество преимуществ, которые делают их незаменимыми инструментами для обработки данных и решения сложных задач. Вот некоторые из основных преимуществ нейронных сетей:

1)Обработка сложных данных: Нейронные сети способны обрабатывать сложные и многомерные данные, включая тексты, изображения, звуковые сигналы и видео. Они могут автоматически извлекать признаки и структуры из этих данных, что делает их эффективными для решения разнообразных задач.

2)Обучение на основе опыта: Нейронные сети могут обучаться на основе опыта и данных. Они способны адаптироваться и улучшать свою производительность с течением времени, постепенно повышая точность и эффективность решений.

3)Распознавание сложных образов и паттернов: Нейронные сети обладают способностью распознавать сложные образы и паттерны в данных. Они могут выявлять скрытые связи и зависимости, что помогает в решении сложных задач, например, в области компьютерного зрения или обработки естественного языка.

4)Гибкость и адаптивность: Нейронные сети предлагают гибкие модели, которые могут быть адаптированы под различные типы данных и задачи. Они могут быть настроены для работы с различными архитектурами и конфигурациями, что делает их универсальными инструментами для различных областей применения.

5)Параллельная обработка: Некоторые типы нейронных сетей, такие как сверточные нейронные сети, обладают способностью выполнять параллельную обработку данных. Это позволяет им эффективно работать с большими объемами информации и повышать скорость обработки.

6)Автоматическое извлечение признаков: Нейронные сети могут автоматически извлекать важные признаки из данных, не требуя ручного определения и предварительной обработки. Это особенно полезно в задачах, где сложно или невозможно явно определить признаки, например, в задачах распознавания образов или анализе текста.

7)Глубокое обучение: Нейронные сети могут использовать глубокое обучение, которое позволяет им строить сложные иерархические модели с множеством слоев. Глубокое обучение позволяет нейронным сетям выявлять более абстрактные и сложные понятия и представления данных, что приводит к улучшенной производительности и точности моделей.

8)Расширяемость и модульность: Нейронные сети могут быть расширены и модифицированы для решения новых задач или адаптации к новым данным. Модели нейронных сетей могут быть комбинированы,

масштабированы и сочетаться с другими методами машинного обучения для достижения лучших результатов.

1.2. Анализ аналогов

Распознавание лиц стало одной из ключевых технологий в современном мире, и ряд компаний занимается разработкой и предоставлением продуктов в этой области.

Amazon Rekognition:

Плюсы: Amazon Rekognition предоставляет широкий спектр функций для распознавания лиц, включая идентификацию, поиск по лицам, анализ выражений лица и возраста, а также детектирование эмоций. Он обладает высокой точностью распознавания и масштабируемостью, а также интегрируется с другими сервисами Amazon Web Services (AWS).

Минусы: Возникают опасения относительно приватности и безопасности данных, поскольку Amazon Rekognition может использоваться для массового наблюдения и отслеживания людей. Кроме того, некоторые исследования показали, что система может быть предвзятой и не всегда точно распознавать лица между различными расами и полами.

Face++ (Megvii):

Плюсы: Face++ является одним из ведущих провайдеров технологий распознавания лиц с высокой точностью и скоростью обработки. Он предлагает множество функций, включая идентификацию, верификацию, анализ лица и эмоций. Face++ широко применяется в различных областях, включая безопасность, финансы, медицину и розничную торговлю.

Минусы: Некоторые возражения были высказаны относительно приватности данных и возможного злоупотребления системой для наблюдения и отслеживания граждан. Также возникают опасения относительно ошибок и предвзятости системы при распознавании лиц различных рас и полов.

Система могла допускать ложные срабатывания и ошибочно идентифицировать лица, что могло приводить к неправильным обвинениям или неверному идентифицированию личности

Microsoft Azure Face API:

Плюсы: Microsoft Azure Face API предоставляет набор функций для распознавания лиц, включая идентификацию, верификацию, анализ эмоций и возраста. Он обладает высокой точностью и поддерживает интеграцию с другими сервисами Azure. Кроме того, Microsoft активно работает над улучшением прозрачности и этичности в использовании технологий распознавания лиц.

Минусы: Возникают опасения относительно приватности и безопасности данных, а также вопросы о недостаточной защите от ошибок и предвзятости системы при распознавании лиц.

При использовании Microsoft Azure Face API возникали проблемы с точностью распознавания идентичных близнецов или людей с сходными физическими особенностями, что может приводить к ошибочной идентификации.

Анализ системы as is to be

Пропускная система с распознаванием лиц и без распознавания лиц имеют существенные отличия в функциональности, точности и удобстве использования.

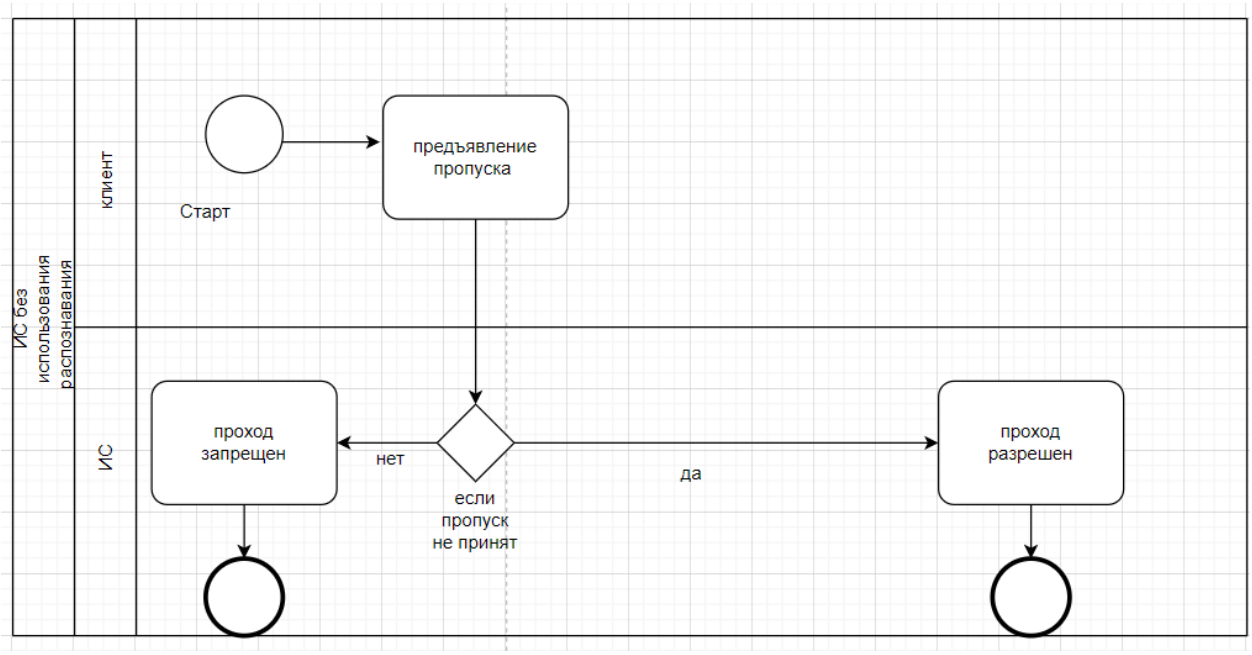


Рис.1.2.1- Модель BPMN без использования ИС

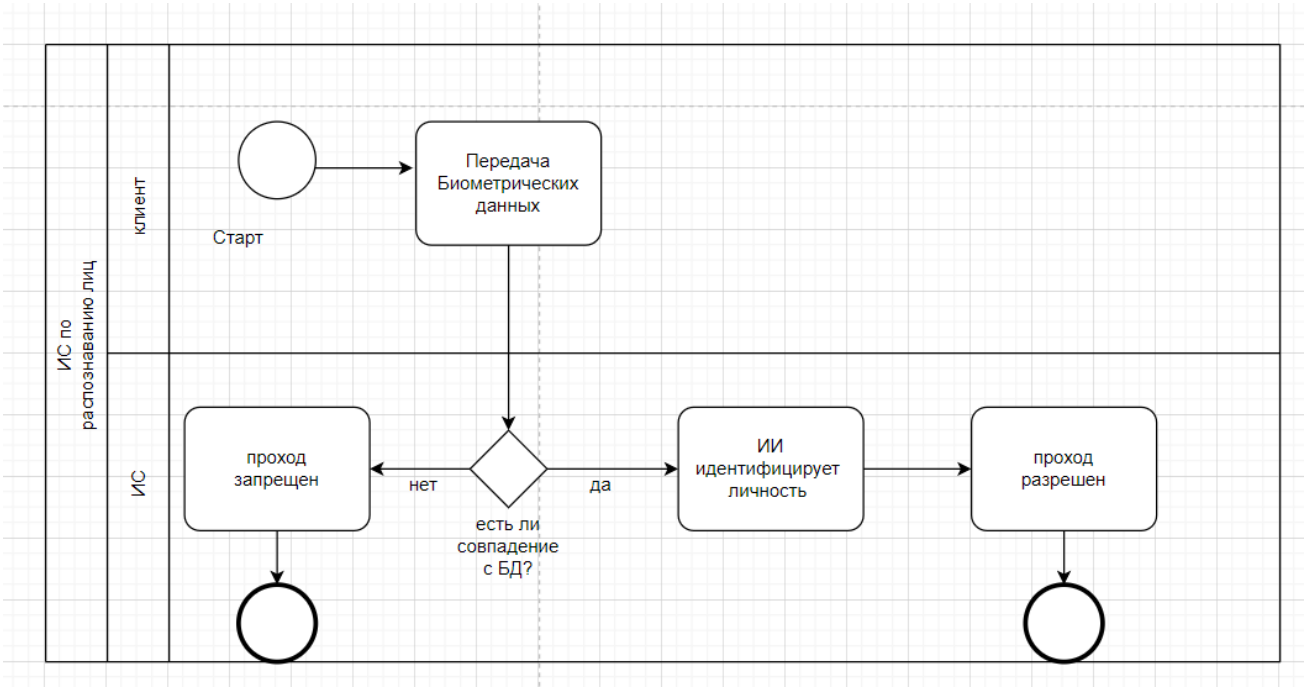


Рис 1.2.2.-Модель BPMN с использованием ИС

1)Идентификация и аутентификация: Пропускная система с распознаванием лиц позволяет идентифицировать личность человека на основе его лица. Это позволяет автоматически определять и аутентифицировать людей без необходимости использования физических карт, брелоков или пин-кодов. В отличие от этого, пропускная система без распознавания лиц может требовать физических пропусков или других методов идентификации, таких как пин-коды или биометрические отпечатки пальцев.

2)Точность и надежность: Пропускные системы с распознаванием лиц, основанные на нейронных сетях, обеспечивают более высокую точность и надежность идентификации, чем системы без распознавания лиц. Это связано с способностью нейронных сетей распознавать уникальные черты лица и отличать их от других. В системах без распознавания лиц возможны ошибки идентификации или возможность обхода системы с использованием поддельных карт или пин-кодов.

3)Удобство использования: Пропускные системы с распознаванием лиц обладают высоким уровнем удобства использования. Пользователям не нужно носить с собой физические карты или помнить пин-коды, что упрощает процесс прохода через контрольный пункт. Системы без распознавания лиц могут быть менее удобными, так как требуют наличия и поддержки физических пропусков или ввода пин-кодов.

4)Автоматизация и скорость: Пропускные системы с распознаванием лиц позволяют автоматизировать процесс идентификации и прохода через контрольный пункт. Это увеличивает скорость прохода и уменьшает очереди. В случае систем без распознавания лиц требуется более ручное взаимодействие, что может замедлить процесс и увеличить время ожидания.

5)Безопасность: Пропускные системы с распознаванием лиц обладают более высоким уровнем безопасности. Они могут определять и предотвращать

попытки мошенничества или несанкционированного доступа, таких как использование поддельных лиц или масок. В системах без распознавания лиц существует возможность подделки физических пропусков или использования чужих карт.

В целом, пропускные системы с распознаванием лиц предлагают более современный, удобный и безопасный способ контроля доступа, чем системы без распознавания лиц. Однако, они также могут быть более дорогостоящими внедрением и требовать определенной инфраструктуры и технической поддержки для эффективной работы.

Глава2 Проектирование

2.1 Концептуальная модель

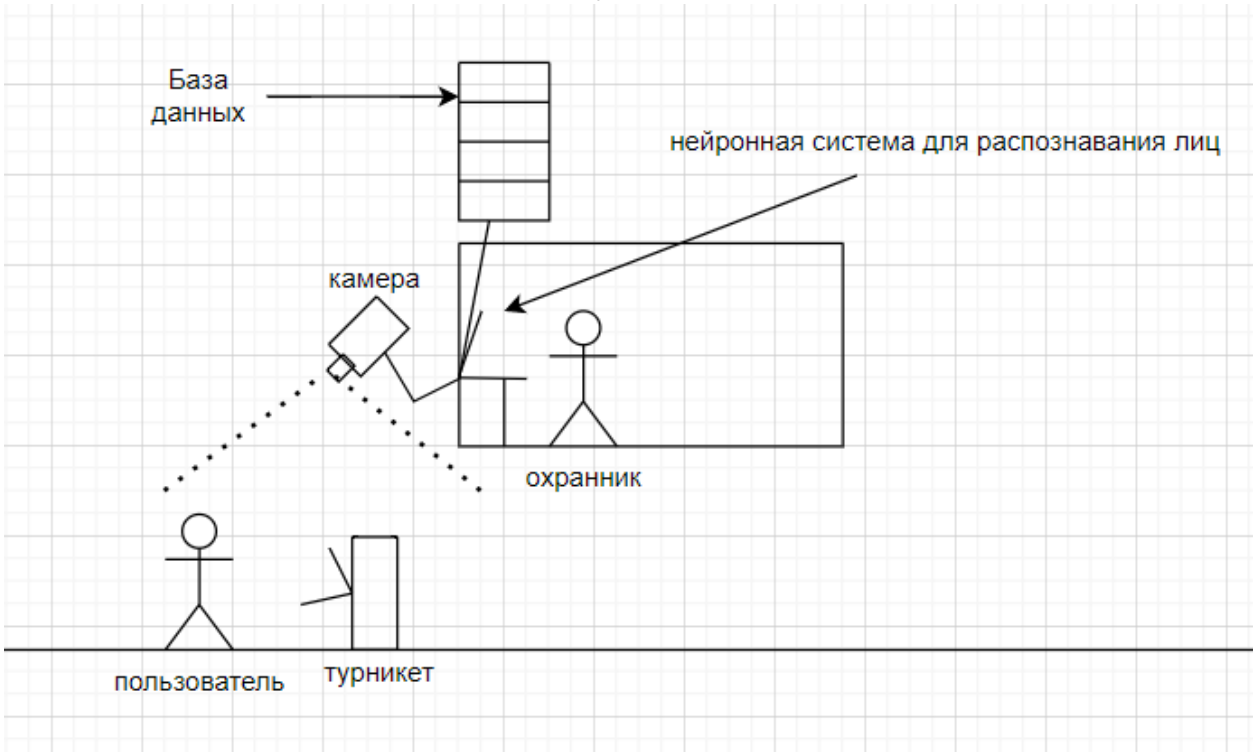


Рис. 2.1.1.-Концептуальная модель системы

Концептуальная модель - это графическое представление концепции или идеи, которое помогает визуализировать и объяснить основные элементы, взаимосвязи и принципы работы системы или процесса. Она обычно используется для общего понимания и обсуждения концепции, без углубления в технические детали.

Концептуальная диаграмма может включать блоки, стрелки, текстовые описания и другие элементы, которые помогают иллюстрировать ключевые идеи и связи между ними. Она может быть использована для представления концепции бизнес-процесса, системы, архитектуры или любого другого комплексного понятия.

Это графическое средство позволяет легче понять, визуализировать и коммуницировать концепцию между людьми с различными фонами и

знаниями. Концептуальная диаграмма может служить основой для более подробного проектирования, разработки и реализации системы или процесса.

Описание концептуальной модели:

1)Пользователь приближается к турникету для получения доступа.

2)Камера автоматически активируется и захватывает изображение лица пользователя.

3)Изображение проходит предварительную обработку и детектируется наличие лица.

4)Нейронная сеть извлекает признаки лица и создает эмбеддинг.

5)Эмбеддинг сравнивается с базой данных известных лиц.

6)Если идентификация успешна, турникет разрешает доступ пользователю.

7)В случае неуспешной идентификации, турникет остается заблокированным.

2.2 Математическое описание

Концептуально, нейронная сеть представляет собой математическую модель, состоящую из набора связанных нейронов, которые работают вместе для выполнения задачи обработки информации. Нейроны в нейронной сети могут быть представлены как математические функции, которые принимают входные данные, выполняют вычисления и генерируют выходные значения.

Формально, математический описатель нейронной сети включает в себя:

1)Архитектуру нейронной сети: Это определяет структуру и организацию нейронов в сети. Включает в себя количество слоев нейронов, количество нейронов в каждом слое и типы связей между нейронами.

2)Веса и смещения: Каждая связь между нейронами имеет соответствующий вес, который определяет важность этой связи. Каждый нейрон также имеет смещение (bias), который влияет на активацию нейрона. Веса и смещения являются числовыми параметрами, которые определяются в процессе обучения нейронной сети.

3)Функции активации: Функции активации определяют поведение нейронов и определяют выходной сигнал в зависимости от входных данных и их взвешенных сумм. Примеры функций активации включают сигмоидную функцию, гиперболический тангенс, ReLU (Rectified Linear Unit) и другие.

4)Метод обучения: Это определяет способ, которым нейронная сеть обучается на основе входных данных и ожидаемых выходных значений. Распространенные методы обучения включают градиентный спуск, обратное распространение ошибки и стохастический градиентный спуск.

Математическое описание нейронной сети может включать уравнения, которые описывают процессы передачи сигналов и обновления весов. Эти уравнения обеспечивают математическую основу для работы нейронной сети и позволяют ей обрабатывать и анализировать данные.

Рис.2.2.1.- Схема нейронной сети

Входы -входные данные

Синапсы- имеют важное значение для обработки и передачи информации в нейронной сети. Они позволяют нейронам коммуницировать друг с другом и передавать сигналы, обеспечивая передачу информации и выполнение различных вычислительных операций в нейронной сети. Синапсы также обладают свойством пластичности, то есть способностью изменять свою силу и эффективность в ответ на опыт и обучение.

Ячейка нейрона- в этом месте происходит обработка сигналов

Аксон-это провод, который передает выходной сигнал от нейрона к другим нейронам или органам тела. Аксоны могут быть очень длинными и связывать нейроны через синапсы.

Выход-выходные данные

Изображение показывает, что искусственный нейрон, аналогично живому, включает в себя синапсы, которые соединяют входы нейрона с ядром; ядро нейрона, которое выполняет обработку входных сигналов, и аксон, который связывает нейрон с нейронами следующего слоя. Каждый синапс имеет свой вес, определяющий степень влияния соответствующего входа на

состояние нейрона. Состояние нейрона определяется с использованием определенной формулы.

Формула определения состояния нейрона так же в приложенииА

$$S = \sum_{i=1}^{M} x_i \cdot wi'$$

где

n – число входов нейрона

x_i – значение i-го входа нейрона

w_i – вес i-го синапса.

Нейронные сети обратного распространения

Нейронная сеть обратного распространения, или сеть с обратным распространением ошибки, является одним из наиболее распространенных и эффективных алгоритмов обучения нейронных сетей. Он основан на принципе обратного распространения ошибки, где ошибка, полученная на выходном слое сети, распространяется обратно к входным слоям, весам и смещениям, чтобы обновить их значения и улучшить общую производительность сети.

Процесс обратного распространения начинается с передачи входных данных через нейронную сеть, после чего вычисляются выходные значения. Затем сравниваются полученные выходы с ожидаемыми значениями, и вычисляется ошибка, которая является разностью между наблюдаемыми и ожидаемыми результатами. Далее эта ошибка распространяется назад по сети, влияя на веса и смещения нейронов.

В процессе обратного распространения используется алгоритм градиентного спуска, чтобы минимизировать ошибку и оптимизировать веса и смещения нейронов. Ошибка на выходном слое распространяется обратно через скрытые слои, где происходит обновление весов в соответствии с градиентом ошибки. Этот процесс повторяется до тех пор, пока ошибка не достигнет минимального значения или сеть не достигнет желаемой точности.

Нейронная сеть обратного распространения позволяет обучать сложные модели, способные решать разнообразные задачи, такие как классификация, регрессия или обработка изображений. Она обладает способностью автоматически извлекать и адаптировать признаки из данных, что делает ее мощным инструментом в области машинного обучения и искусственного интеллекта.

В общем случае задача обучения НС сводится к нахождению некой функциональной зависимости *Y=F(X)* где *X* – входной, а *Y* – выходной векторы

Формула нахождения весов при обратном распространении, так же в Приложении Б

$$E(w) = \frac{1}{2}\sum_{j=1}^{P}[y_i - d_i]^2$$

Где:

y_j – значение j-го выхода нейросети,

d_j – целевое значение j-го выхода,

p – число нейронов в выходном слое.

Метод обучения градиентного спуска (Gradient Descent) является одним из наиболее распространенных методов обучения нейронных сетей. Он используется для настройки весов нейронов в сети с целью минимизации ошибки предсказания.

Принцип работы метода градиентного спуска заключается в следующем:

1)Инициализация весов: В начале обучения инициализируются случайные значения весов для всех нейронов в сети.

2)Прямое распространение: Входные данные подаются на вход нейронной сети, и сигналы проходят через нейроны по направлению от входов к выходам. Каждый нейрон вычисляет свой выход на основе входных данных и текущих весов.

3)Вычисление ошибки: Сравнивается выход сети с ожидаемым выходом для данного входа, и вычисляется ошибка. Ошибка может быть выражена в виде функции потерь, такой как среднеквадратичная ошибка (MSE) или перекрестная энтропия (Cross-Entropy).

4)Обратное распространение ошибки: Ошибка распространяется обратно через сеть, начиная с выходного слоя и перемещаясь к входному слою. Каждый нейрон вычисляет свою частную производную ошибки по своему выходу и обновляет веса в соответствии с этой производной.

5)Обновление весов: Веса нейронов обновляются в направлении, противоположном градиенту функции потерь. Это делается путем умножения градиента на скорость обучения (learning rate) и вычитания результата из текущих весов. Скорость обучения определяет величину шага, с которым изменяются веса.

6)Повторение шагов 2-5: Шаги прямого и обратного распространения ошибки повторяются для каждого входного примера в обучающем наборе.

Обновление весов происходит после каждого примера (пакетный градиентный спуск) или после прохождения всех примеров (стохастический градиентный спуск).

Прекращение обучения: Обучение может быть прекращено, когда достигнут критерий остановки, такой как достаточно малая ошибка или определенное количество эпох обучения.

Метод градиентного спуска позволяет настраивать веса нейронов таким образом, чтобы минимизировать ошибку предсказания. Он основан на идее итеративного обновления весов в направлении, обратном градиенту функции потерь. Этот процесс повторяется до достижения сходимости, когда веса достигают значения, при которых ошибка минимальна.

Формула вычисления минимальной ошибки весов, так же в Приложении В

$$\Delta w_{ij} = -n \cdot \frac{\partial E}{\partial w_{ij}}$$

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_i} \cdot \frac{dy_i}{dS_j} \cdot \frac{\partial S_j}{\partial w_{ij}}$$

где

y_j – значение выхода j-го нейрона,

S_j – взвешенная сумма входных сигналов, определяемая по формуле (1).

При этом множитель

$$\frac{\partial S_i}{\partial w_{\ddot{y}}} = x_i$$

где

x_i – значение i-го входа нейрона.

Выявления первого множителя функции

$$\frac{\partial E}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{dy_k}{dS_k} \cdot \frac{\partial S_k}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{dy_k}{dS_k} \cdot w_{jk}^{(n+1)}$$

где

k – число нейронов в слое $n+1$.

Так же нужно ввести вспомогательную переменную

$$\delta_J^{[n]} = \frac{\partial E}{dy_j} \cdot \frac{dy_j}{dS}$$

В этом случае мы можем определить рекурсивную формулу для n-ного слоя для последующего

$$\delta_J^{(n)} = \sum_k \delta_k^{(n+1)} \cdot w \frac{(n+1)}{jk} \cdot \frac{dy_i}{dS_j}$$

В дальнейшем выведем итоговую формулу корректировки весов

$$w_{ij}^{[n]}(-t) = wJ_i^{(n)}(t-1) + \Delta w_{ij}{}^{(n)}(t)$$

2.3 IDEF0

IDEF0 предоставляет структурированный подход к описанию функций, потоков данных, управления и ресурсов в рамках системы или процесса. Он представляет процессы в виде функциональных блоков и стрелок, которые показывают потоки данных и управления между блоками.

Основная цель IDEF0 - это создание ясного и понятного представления бизнес-процессов, которое помогает визуализировать последовательность действий, взаимосвязи между функциями и их взаимодействие с внешними сущностями. Это позволяет улучшить понимание процессов, выявить проблемные места и предложить изменения для оптимизации производительности и качества работы.

1 уровень

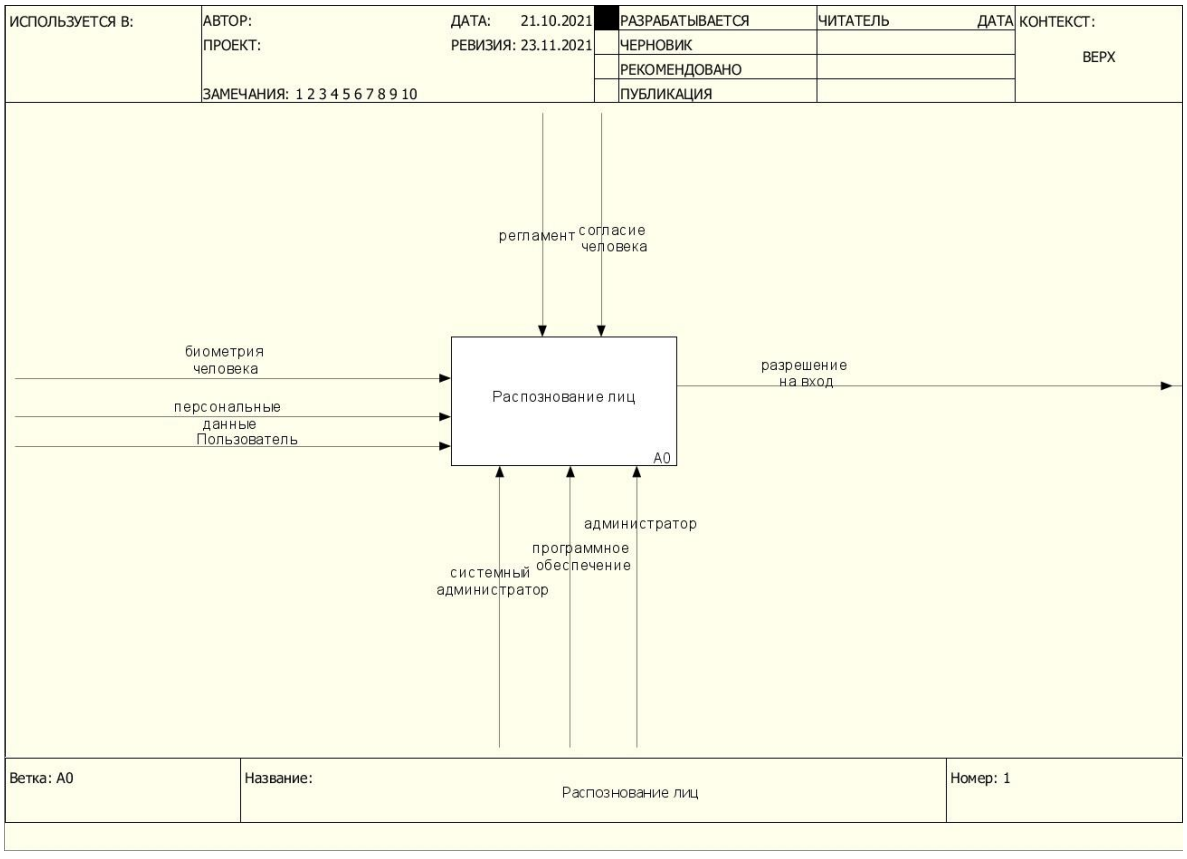


Рис2.3.1- Первый уровень модели IDEF0 ”Распознавание лиц”.

На первом уровне мы видим главный процесс и данные, которые поступают в него и из него.

2 уровень

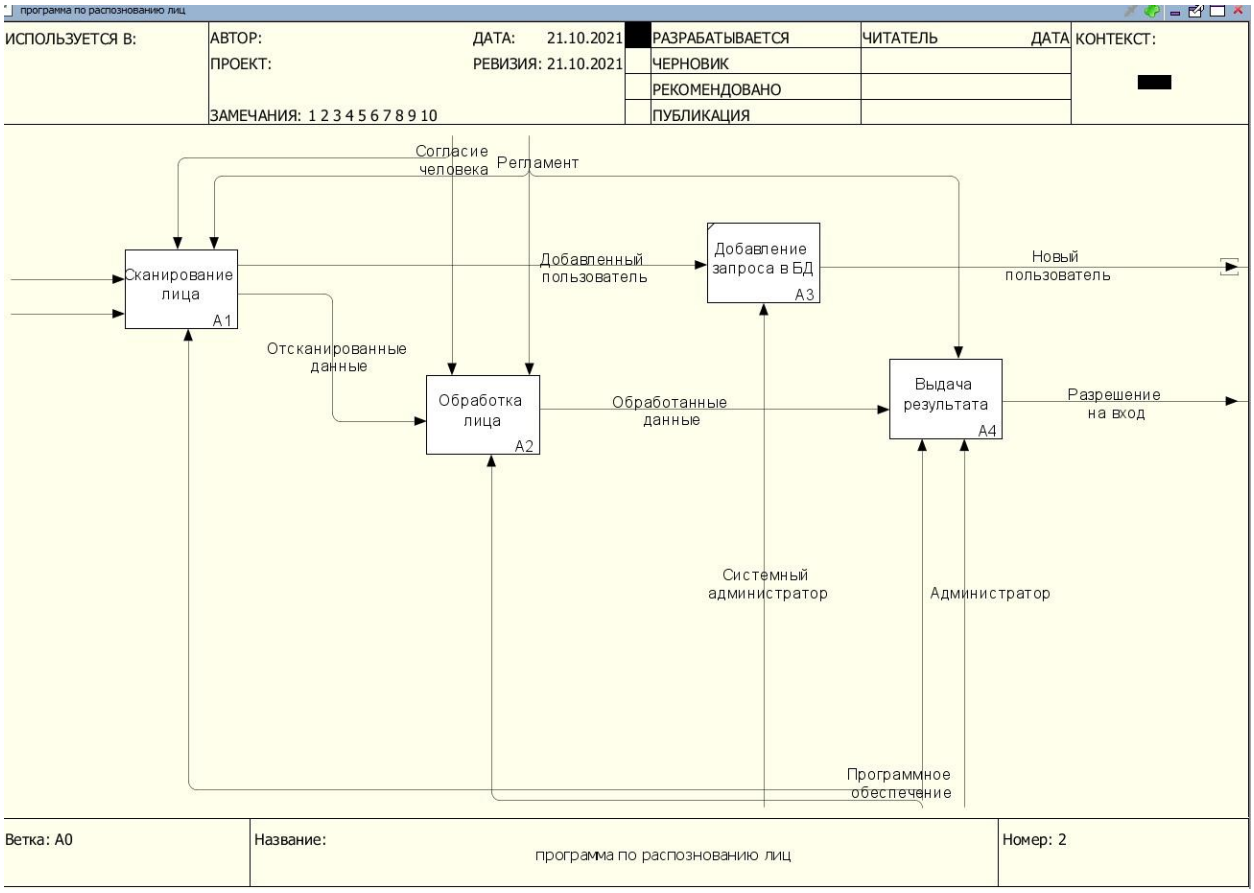


Рис.2.3.2- Второй уровень модели IDEF0 ”Программа по распознаванию лиц”

На втором уровне видно, как главный процесс “программа по распознаванию лиц” расписан на 4 дочерних процесса такие как : ”Сканирование лица”, ”Обработка лица”, ”Добавление запроса в БД”, ”Выдача результата”. В дальнейшем мы рассмотрим каждый процесс более детально.

3 уровень

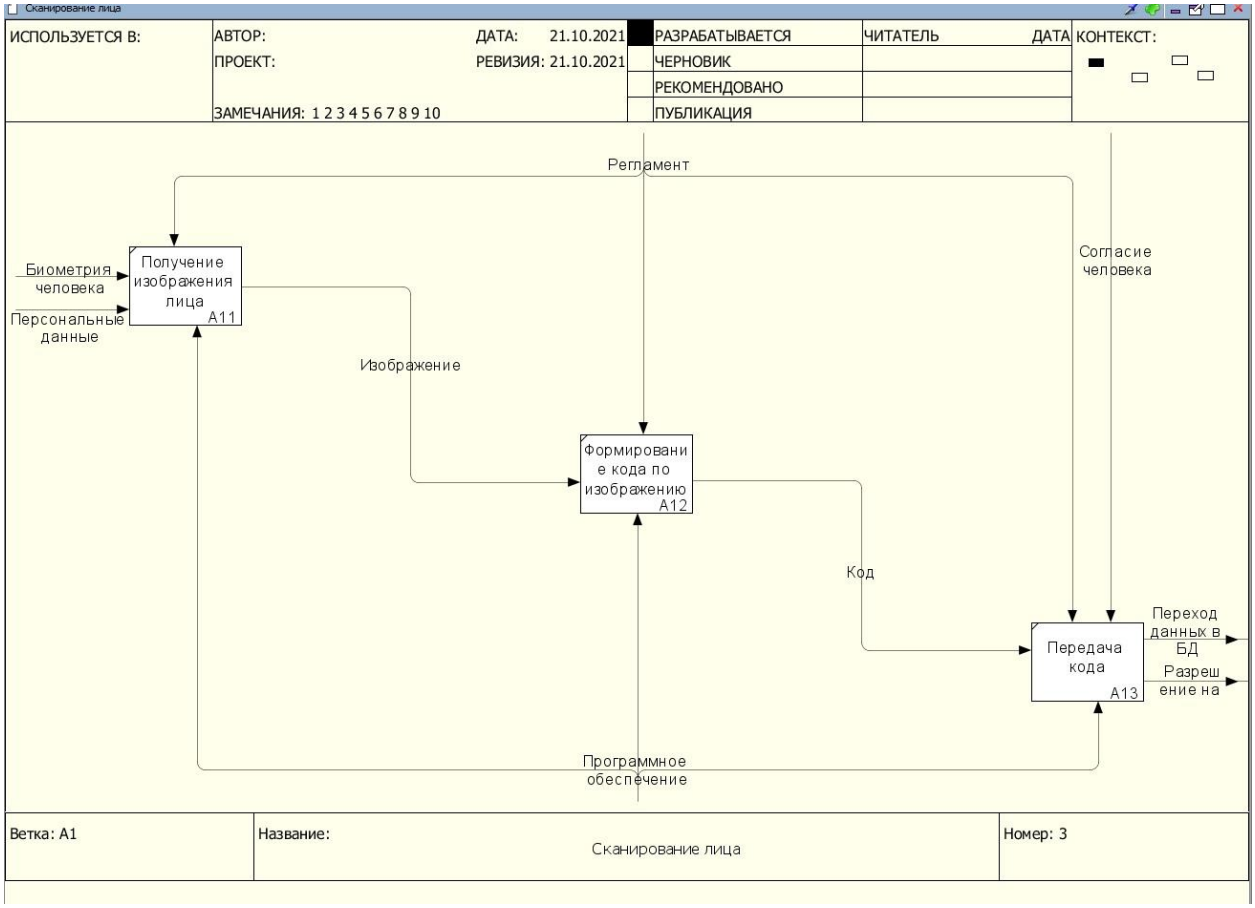


Рис.2.3.3-Третий уровень модели IDEF0”Сканирование лица”

На этом уровне мы подробнее рассматриваем процесс ”Сканирование лица ”.

4 уровень

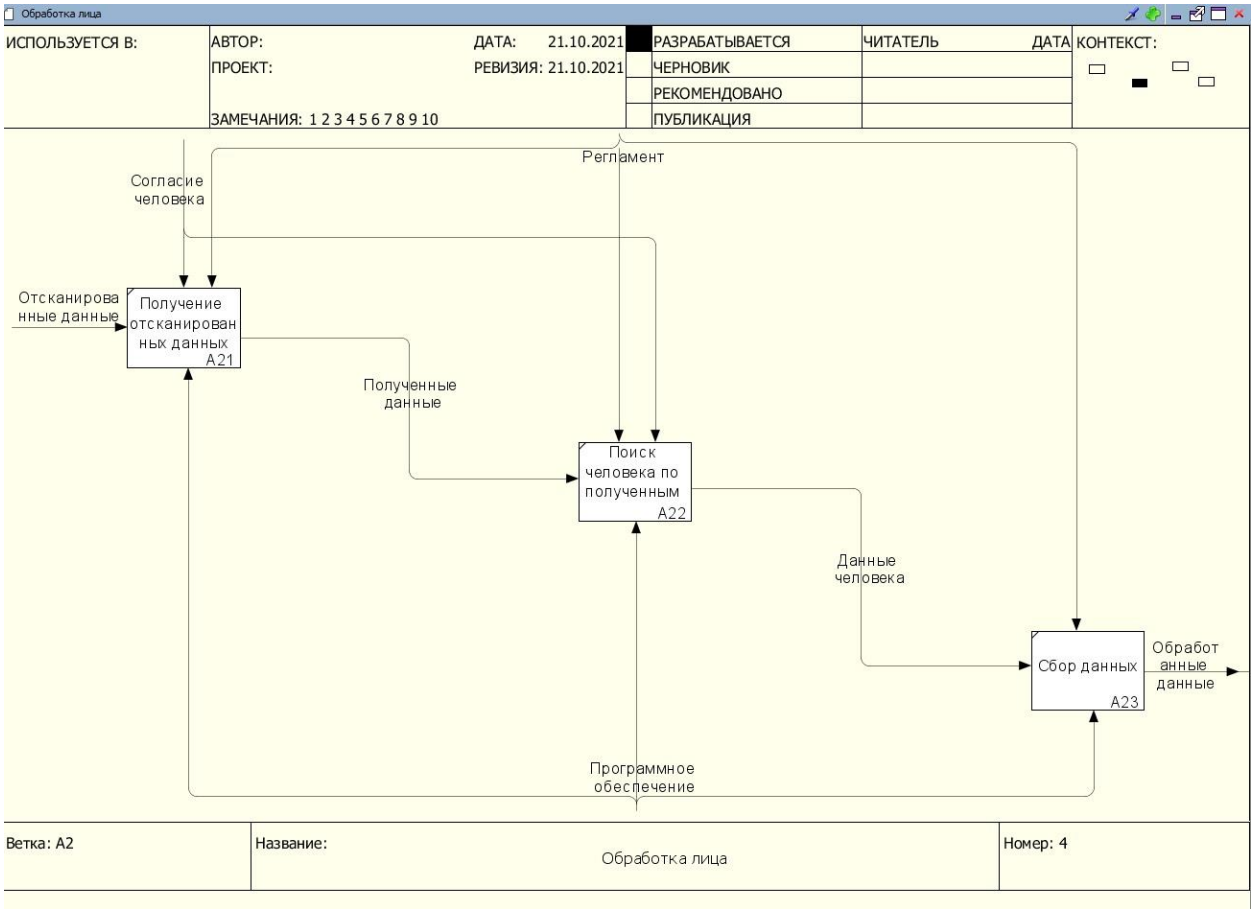


Рис.2.3.4-Четвертый уровень модели IDEF0 ”Обработка лица”

Данный уровень представляет собой процесс ”Обработка лица”.

5 уровень

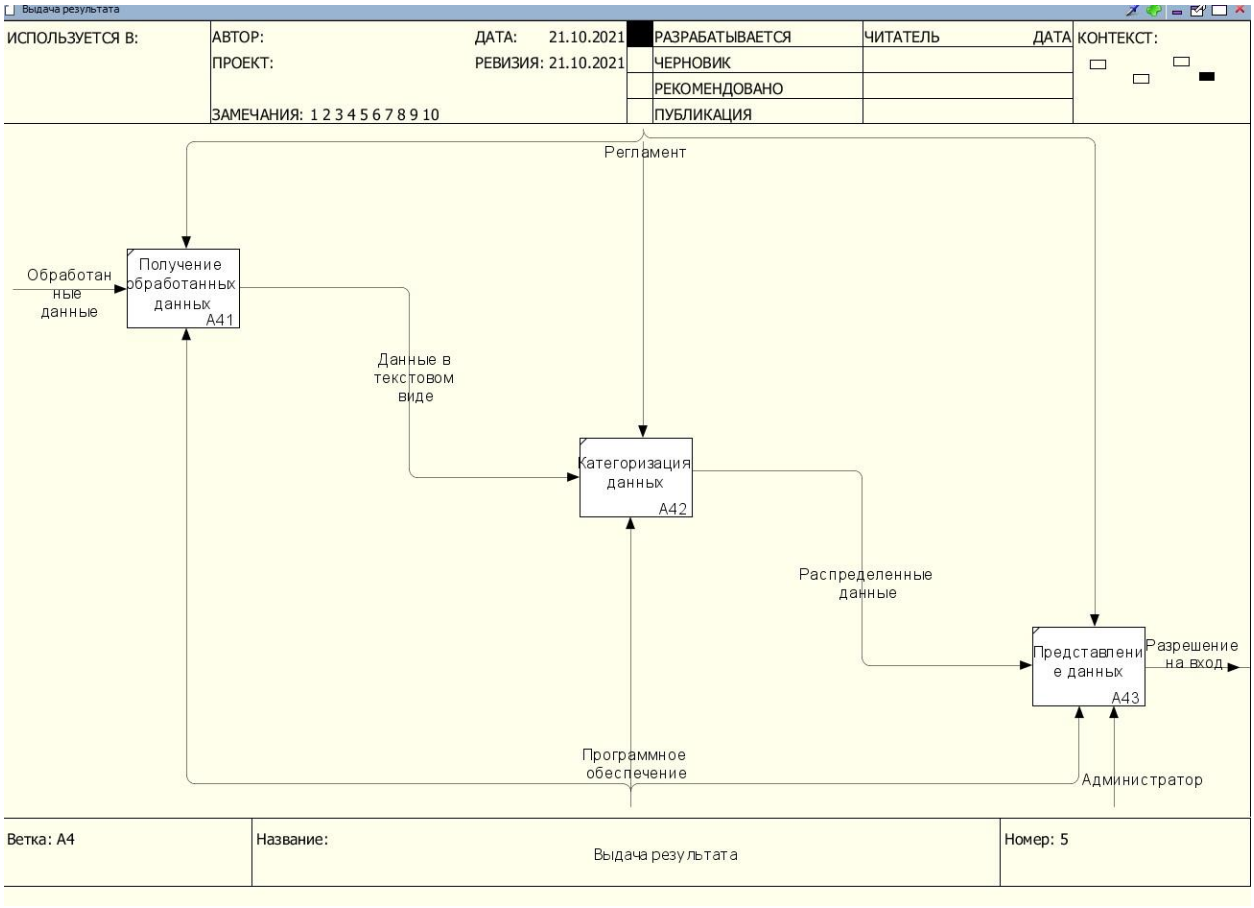


Рис.2.3.5-Пятый уровень модели IDEF0”Выдача результата”

Последний уровень включает в себя описание процесса ”Выдача результата”.

2.4 Диаграмма последовательности

Диаграмма последовательности - это графическое представление, которое иллюстрирует взаимодействие между объектами или компонентами в определенной последовательности. Она позволяет визуализировать, как объекты взаимодействуют друг с другом в рамках определенной функциональности или процесса.

Диаграммы последовательности широко используются в разработке программного обеспечения для моделирования взаимодействия объектов во время выполнения. Они представляют собой последовательность сообщений, которые передаются между объектами во время выполнения определенной операции или сценария.

На диаграмме последовательности объекты или компоненты представлены вертикальными линиями, называемыми "активациями". Временные оси на диаграмме показывают последовательность выполнения операций или сообщений между объектами. Сообщения обычно представлены стрелками, которые указывают направление передачи информации или управления.

Диаграммы последовательности обеспечивают наглядное представление взаимодействия между объектами, помогают идентифицировать потоки управления и передачи данных, а также выявлять возможные проблемы или улучшения в дизайне системы или процесса.

Они широко применяются в разработке программного обеспечения, проектировании систем, анализе бизнес-процессов и других областях, где требуется визуальное представление последовательности действий и взаимодействия между объектами.

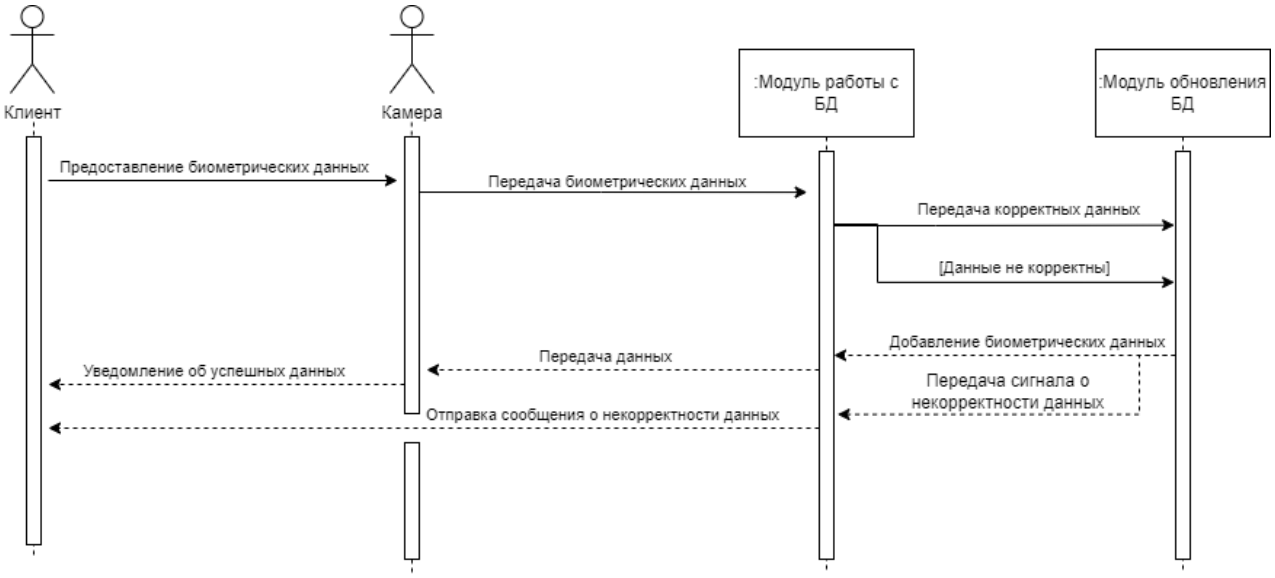


Рис. 2.4.1. Диаграмма последовательности “Фотографирование лица”

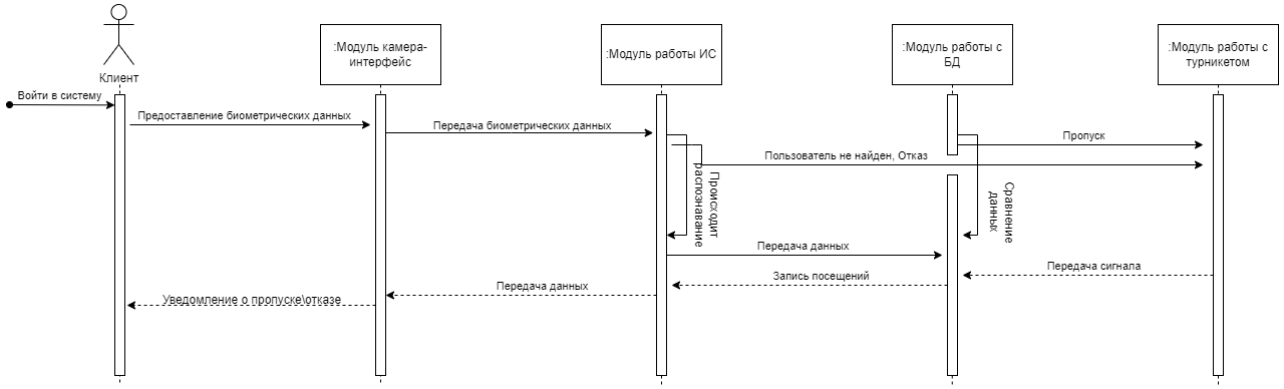


Рис. 2.4.2 Диаграмма последовательности “Сканирование лица и распознавании лица”

3 Глава. Разработка прототипа

3.1 Разработка Нейронной сети

Описание разработки Нейронной сети

Для разработки нейронной сети я использовал python, а именно версию 3.8. основными библиотеками, которыми я пользовался open cv для захвата изображения с камеры и face recognition для сопоставления изображений

Для начала создал объект для захвата видео с веб-камеры и подключил библиотеку open cv для захвата с нее изображения и библиотеку face recognition для обнаружения лица и его кодировок в дальнейшем для сопоставления с базой данных (рис 1)

```
cap = cv2.VideoCapture(0)

is_logging_enabled = True
start_time = datetime.now()
write_to_log = True

encodeListKnown, namesListKnown = findEncodingsFromDB()
print("Декодирование закончено")

while True:
    success, img = cap.read()
    imgS = cv2.resize(img, (0, 0), None, 0.25, 0.25)
    imgS = cv2.cvtColor(imgS, cv2.COLOR_BGR2RGB)

    facesCurFrame = face_recognition.face_locations(imgS)
    encodeCurFrame = face_recognition.face_encodings(imgS, facesCurFrame)

    for encodeFace, faceLoc in zip(encodeCurFrame, facesCurFrame):
        matches = face_recognition.compare_faces(encodeListKnown, encodeFace)
        faceDis = face_recognition.face_distance(encodeListKnown, encodeFace)

        matchIndex = np.argmin(faceDis)

        if matches[matchIndex]:
            name = namesListKnown[matchIndex]

            y1, x2, y2, x1 = faceLoc
            y1, x2, y2, x1 = y1 * 4, x2 * 4, y2 * 4, x1 * 4
            cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 2)
            cv2.rectangle(img, (x1, y2 - 35), (x2, y2), (0, 255, 0), cv2.FILLED)
            cv2.putText(img, name, (x1 + 6, y2 - 6), cv2.FONT_HERSHEY_COMPLEX, 1, (255, 255, 255), 2)
            current_time = datetime.now()
            time_diff = current_time - last_recognition if last_recognition is not None else start_time
            if is_logging_enabled and (
                    datetime.now() - start_time).total_seconds() >= 5:  # Раз в 5 секунд будут записываться данные в БД
                start_time = datetime.now()
                write_to_log = True
            else:
                write_to_log = False

            if write_to_log:
                markAttendance(name, current_time)
```

Рис3.1.1 Первоначальное представления кода нейронной сети

К тому же был создана функция, при которой при каждом распознавании пользователя идентифицировалась его личность и записывалась дата и время сканирования на вход и тоже самое на выход (Рис. 2). Так же это реализовано частично и на (Рис.1). Все эти данные так же записываются и хранятся в базе данных.

```
def markAttendance(name, current_time):
    mydb = connect_to_db()
    cursor = mydb.cursor()
    cursor.execute("SELECT COUNT(*) FROM recognition_log WHERE first_name = %s AND last_name = %s AND patronymic = %s AND time_exit IS NULL", name.split())
    result = cursor.fetchone()
    count = result[0]
    if count == 0:
        sql = "INSERT INTO recognition_log (first_name, last_name, patronymic, time_entry) VALUES (%s, %s, %s, %s)"
        val = name.split() + [current_time]
    else:
        sql = "UPDATE recognition_log SET time_exit = %s, time_spent = TIMEDIFF(time_exit, time_entry) WHERE first_name = %s AND last_name = %s AND patronymic = %s
        val = [current_time] + name.split()
    cursor.execute(sql, val)
    mydb.commit()
    mydb.close()
```

Рис3.1.2-Код для идентификации личности и записи времени

Так же есть функция, которая извлекает кодировки из базы данных, для дальнейшей работы нейронной сети(Рис.3)

```
def findEncodingsFromDB():
    encodeList = []
    namesList = []
    mydb = connect_to_db()

    cursor = mydb.cursor()
    cursor.execute("SELECT first_name, last_name, patronymic, face_photo FROM people")
    rows = cursor.fetchall()

    for row in rows:
        first_name, last_name, patronymic, photo = row
        img = cv2.imdecode(np.asarray(bytearray(photo), dtype=np.uint8), cv2.IMREAD_COLOR)
        img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
        name = f"{first_name} {last_name} {patronymic}"
        namesList.append(name)

    mydb.close()
    return encodeList, namesList
```

Рис3.1.3-Код для извлечения кодировок из базы данных

Описание разработки БД

Для разработки БД и её структуры было решено использовать программу MySQL Workbench, которая в свою очередь удобная для хранения и записи данных и разработки БД. После установки и создания логина и пароля для БД программа выглядит вот так (Рис.4)

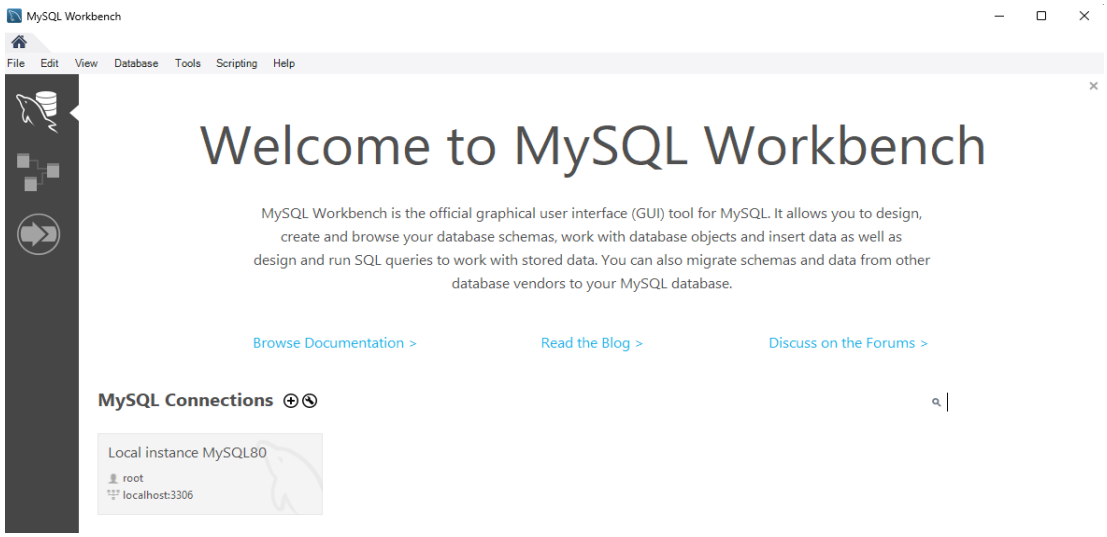


Рис3.1.4-Программа для разработки БД

Также было создано подключение к базе данных, как локальный администратор с ранее созданными данными (в качестве логина - root, в качестве пароля - qw123)

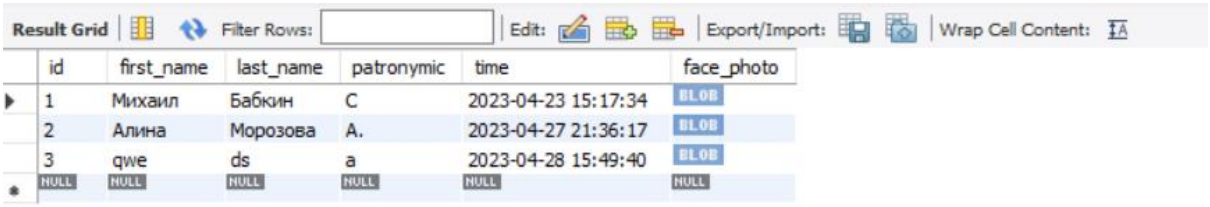
Для реализации распознавания была создана таблица People

Таблица с людьми, которые проходят через систему (people)(Рис5)

```
1  CREATE TABLE People (
2      id INT PRIMARY KEY AUTO_INCREMENT,
3      first_name VARCHAR(50) NOT NULL,
4      last_name VARCHAR(50) NOT NULL,
5      patronymic VARCHAR(50) NOT NULL,
6      time TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
7      face_photo LONGBLOB
8  );
```

Рис3.1.5-Создание таблицы “people”

Из этой таблицы берутся данные для распознавания и идентификации, именно кодировки изображения берутся из 6 столбца (Рис.6)


Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

id	first_name	last_name	patronymic	time	face_photo
1	Михаил	Бабкин	С	2023-04-23 15:17:34	BLOB
2	Алина	Морозова	А.	2023-04-27 21:36:17	BLOB
3	qwe	ds	a	2023-04-28 15:49:40	BLOB
NULL	NULL	NULL	NULL	NULL	NULL

Рис3.1.6-Таблица People

Обучение нейронной сети

Обучение нейронной сети - это процесс, в ходе которого нейронная сеть "узнает" из данных и адаптируется для выполнения конкретной задачи.

Вначале нейронная сеть инициализируется случайными весами. Затем на вход сети подаются обучающие примеры, состоящие из входных данных (например, изображений) и соответствующих им правильных меток (например, ФИО). Сеть пропускает входные данные через свои слои и выдаёт предсказания.

Сравнивая предсказания с правильными ответами, вычисляется ошибка, которая отражает насколько сеть ошиблась в своих предсказаниях. Затем, с помощью алгоритма обратного распространения ошибки, веса нейронной сети корректируются таким образом, чтобы уменьшить ошибку и улучшить предсказания.

Процесс обучения повторяется на множестве обучающих примеров в течение нескольких эпох, пока сеть не достигнет приемлемого уровня точности на обучающей выборке. После этого сеть может быть протестирована на новых, ранее не виденных данных, чтобы оценить её способность обобщать и делать правильные предсказания.

Таким образом, обучение нейронной сети заключается в оптимизации её весов с целью минимизации ошибки и повышения её способности к предсказаниям на основе предоставленных данных.

Мой пример когда с реализацией обучения нейронной сети с помощью библиотеки Tensorflow (Рис.7)

```
import tensorflow as tf
from tensorflow.keras import layers


train_image_paths = ['train_image']
train_labels = ['train_label']

train_dataset = tf.data.Dataset.from_tensor_slices((train_image_paths, train_labels))

model = tf.keras.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(64, 64, 3)),
    layers.MaxPooling2D((2, 2)),
    layers.Flatten(),
    layers.Dense(128, activation='relu'),
    layers.Dense(num_classes, activation='softmax')
])


model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(),
              metrics=['accuracy'])


epochs = 10
batch_size = 32

model.fit(train_dataset, epochs=epochs, batch_size=batch_size)


test_loss, test_accuracy = model.evaluate(test_dataset)


predictions = model.predict(new_images)
```

Рис3.1.7-реализация кода обучения нейронной сети

Интерфейс системы

После запуска файла main.py нас встречает окно Авторизации (рис.8).

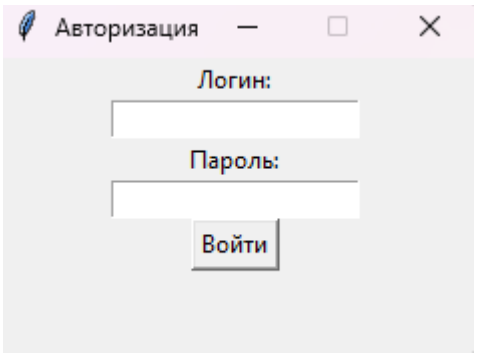


Рис3.1.8-Окно Авторизации

Как только оно открылась, мы входим и авторизируемся под пользователем так как в нашем случае в роле пользователя выступает охранник, который непосредственно следит за проходом сотрудников предприятия. Само окно выглядит следующим образом (Рис.9)

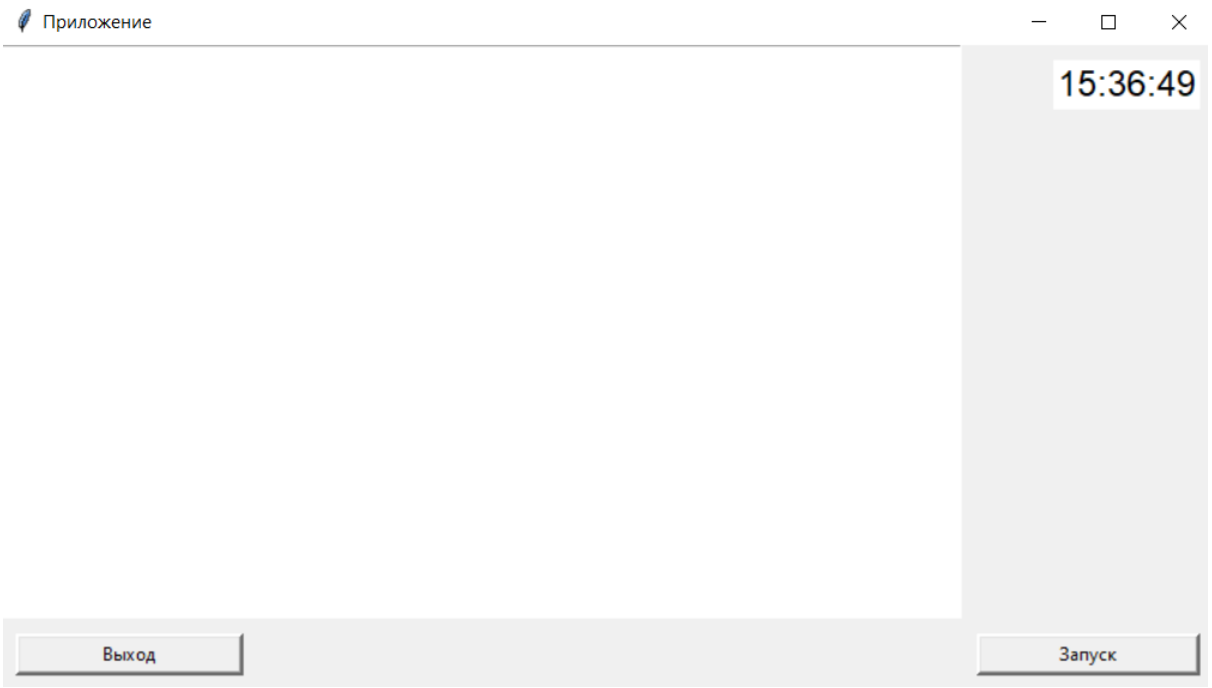


Рис3.1.9-Окно интерфейса охранника

В окне мы наблюдаем кнопку “Выхода”, которая в случае нажатия закроет приложения, так же видим часы с реальным временем и самы кнопку “Запуск” ,как раз таки для запуска распознавания (Рис10)

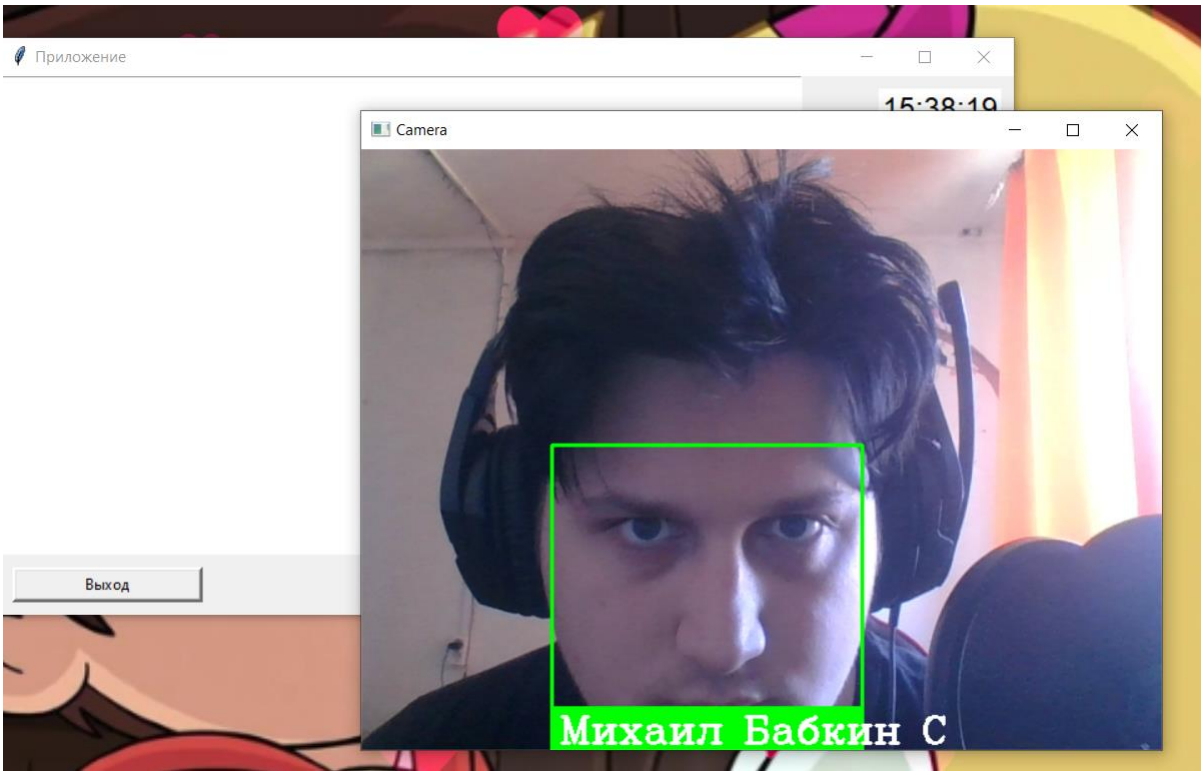


Рис3.1.10-Демонстрация работы нейронной сети

3.2 Разработка модулей обучений нейронной сети

Обучение нейронной сети - это процесс настройки параметров и связей между нейронами сети на основе имеющихся данных, с целью создания модели, способной решать конкретную задачу или выполнять определенную функцию. Вот подробный обзор процесса обучения нейронной сети:

1)Подготовка данных: Первым шагом в обучении нейронной сети является подготовка данных. Это включает в себя сбор и предварительную обработку данных, которые будут использоваться для обучения. Входные данные могут быть числовыми, текстовыми, изображениями или другими форматами в зависимости от задачи, которую пытается решить сеть.

2)Определение архитектуры сети: Следующим шагом является определение архитектуры нейронной сети, то есть определение числа слоев, количества нейронов в каждом слое и типов связей между нейронами. Архитектура сети зависит от задачи, которую она должна решать, и может варьироваться от простых моделей до сложных глубоких нейронных сетей.

3)Инициализация весов: Веса являются параметрами, определяющими силу связей между нейронами. На этом этапе веса инициализируются случайными значениями, их начальная инициализация имеет важное значение для успешного обучения сети.

4)Прямое распространение (Forward Propagation): Во время прямого распространения данные подаются на вход сети, и сигналы проходят через каждый нейрон в сети, преобразуясь по определенным правилам. Результатом является выходной сигнал сети, который представляет собой предсказание или вывод модели.

5)Расчет функции потерь: Функция потерь измеряет расхождение между предсказанными значениями и истинными значениями целевой переменной. Целью обучения является минимизация этой функции потерь. Популярные

функции потерь включают среднеквадратичную ошибку (MSE) для задач регрессии и кросс-энтропию для задач классификации.

6)Обратное распространение (Backpropagation): Обратное распространение - это алгоритм, используемый для вычисления градиентов функции потерь по отношению к весам в сети. Градиенты вычисляются с использованием цепного правила и передаются обратно через сеть. Это позволяет определить, какие веса сети необходимо изменить для улучшения ее производительности.

7)Обновление весов: После вычисления градиентов веса сети обновляются с использованием методов оптимизации, таких как стохастический градиентный спуск (SGD) или его вариации. Это приводит к обновлению параметров сети, что позволяет ей становиться все более точной в решении задачи.

8)Итеративный процесс: Шаги 4-7 повторяются множество раз в процессе обучения. Каждая итерация состоит из прямого и обратного распространения, расчета функции потерь и обновления весов. Число итераций зависит от сложности задачи и размера данных обучения.

9)Оценка производительности: После завершения обучения сети ее производительность оценивается на отложенных тестовых данных. Это позволяет определить, насколько хорошо сеть обучилась и способна ли она достичь требуемого уровня точности или производительности.

10)Настройка и повторение: В случае недостаточной производительности сети возможно внесение изменений в архитектуру, параметры или данные обучения, а затем повторение процесса обучения сети для достижения лучших результатов.

Метод обратного распространения ошибки

Алгоритм обратного распространения ошибки (Backpropagation) является ключевым компонентом обучения нейронной сети. Он позволяет вычислить градиенты функции потерь по отношению к весам сети, что в свою очередь позволяет обновить веса и улучшить производительность сети. Вот подробный алгоритм обратного распространения ошибки:

1)Прямое распространение (Forward Propagation):

Подается входной сигнал на входной слой сети.Сигнал распространяется через каждый слой сети, начиная с входного слоя и до выходного слоя. Для каждого нейрона в каждом слое вычисляется выходное значение, используя активационную функцию и веса связей между нейронами.

2)Вычисление функции потерь (Loss Calculation):

После прямого распространения сравниваются выходные значения сети с ожидаемыми значениями (целевой переменной). Вычисляется функция потерь, которая измеряет расхождение между предсказанными значениями и истинными значениями.

3)Обратное распространение (Backward Propagation):

Начиная с выходного слоя, вычисляется градиент функции потерь по отношению к выходу каждого нейрона в слое. Градиенты вычисляются с использованием частных производных и цепного правила. Градиенты передаются назад через сеть, и для каждого нейрона в каждом слое вычисляется градиент по отношению к его входу.

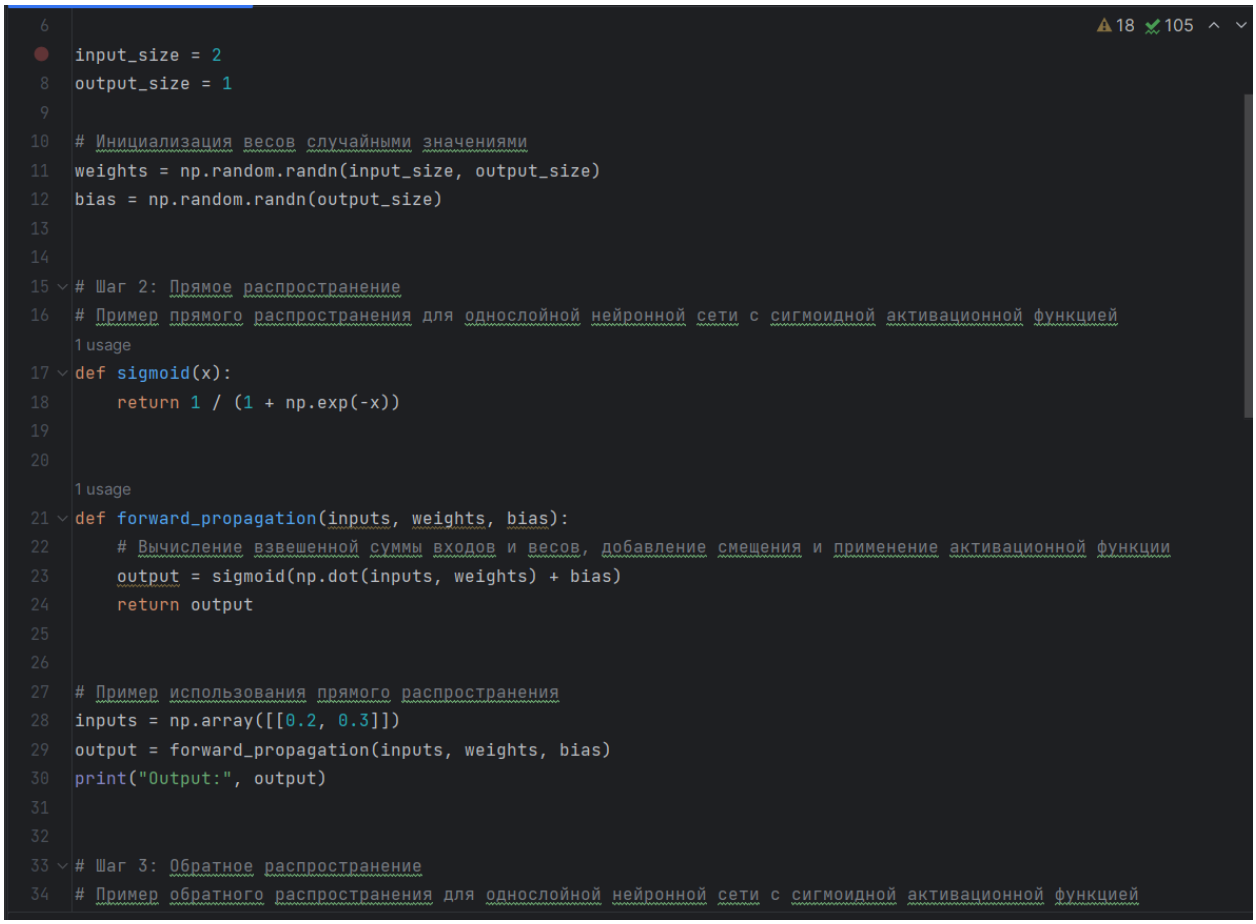
4)Обновление весов (Weight Update):

После вычисления градиентов по отношению к входам нейронов, градиенты передаются далее для вычисления градиентов по отношению к весам связей. Веса обновляются в направлении, противоположном градиенту, с использованием методов оптимизации, таких как стохастический

градиентный спуск (SGD) или его вариации. Обновление весов происходит с целью минимизации функции потерь и улучшения производительности сети.

5)Повторение:

Шаги 1-4 повторяются для каждого примера из обучающего набора данных или мини-пакета данных. Весь процесс итеративно повторяется до достижения заданного критерия остановки, например, определенного числа эпох или сходимости функции потерь.



```
input_size = 2
output_size = 1

# Инициализация весов случайными значениями
weights = np.random.randn(input_size, output_size)
bias = np.random.randn(output_size)


# Шаг 2: Прямое распространение
# Пример прямого распространения для однослойной нейронной сети с сигмоидной активационной функцией
1 usage
def sigmoid(x):
    return 1 / (1 + np.exp(-x))


1 usage
def forward_propagation(inputs, weights, bias):
    # Вычисление взвешенной суммы входов и весов, добавление смещения и применение активационной функции
    output = sigmoid(np.dot(inputs, weights) + bias)
    return output


# Пример использования прямого распространения
inputs = np.array([[0.2, 0.3]])
output = forward_propagation(inputs, weights, bias)
print("Output:", output)


# Шаг 3: Обратное распространение
# Пример обратного распространения для однослойной нейронной сети с сигмоидной активационной функцией
```

Рис3.2.1-Код реализации обратного распространения ошибки

Градиентный метод

Градиентный метод обучения является одним из наиболее распространенных и эффективных методов обновления весов нейронных сетей на основе градиента функции потерь. Этот метод использует информацию о градиентах функции потерь по отношению к весам сети для определения направления обновления весов, с целью минимизации функции потерь и улучшения производительности сети. Вот подробное описание градиентного метода обучения нейронной сети:

1)Вычисление градиентов: В начале процесса обучения нейронной сети вычисляются градиенты функции потерь по отношению к весам сети с помощью обратного распространения ошибки (backpropagation). Градиенты показывают, как изменение каждого веса влияет на значение функции потерь.

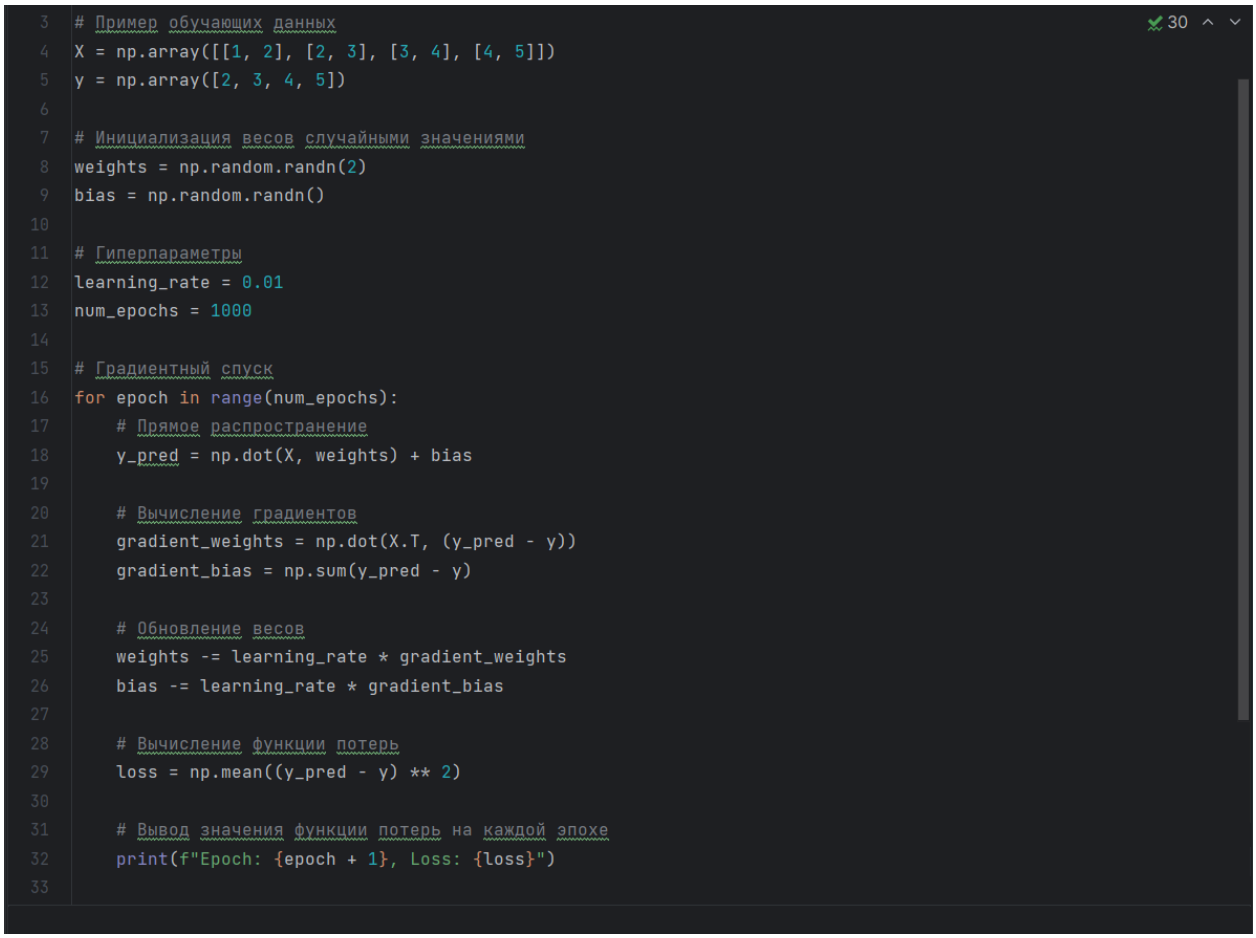
2)Обновление весов: Используя вычисленные градиенты, происходит обновление весов сети. Обычно используется простая формула:

новый_вес = старый_вес - скорость_обучения * градиент

Здесь "скорость_обучения" (learning rate) - это гиперпараметр, определяющий шаг обновления весов. Он контролирует величину изменений весов на каждой итерации.

3)Повторение процесса: Шаги 1 и 2 повторяются множество раз (называемое эпохой) для всех примеров обучающего набора данных или для небольших мини-пакетов данных. Это позволяет сети постепенно улучшать свою производительность и приближаться к минимуму функции потерь.

4)Контроль сходимости: Во время обучения можно отслеживать значение функции потерь на отложенном наборе данных или на каждой эпохе. Если значение функции потерь перестает значительно изменяться или превышает заданный порог, обучение может быть остановлено.



```python
# Пример обучающих данных
X = np.array([[1, 2], [2, 3], [3, 4], [4, 5]])
y = np.array([2, 3, 4, 5])

# Инициализация весов случайными значениями
weights = np.random.randn(2)
bias = np.random.randn()

# Гиперпараметры
learning_rate = 0.01
num_epochs = 1000

# Градиентный спуск
for epoch in range(num_epochs):
    # Прямое распространение
    y_pred = np.dot(X, weights) + bias

    # Вычисление градиентов
    gradient_weights = np.dot(X.T, (y_pred - y))
    gradient_bias = np.sum(y_pred - y)

    # Обновление весов
    weights -= learning_rate * gradient_weights
    bias -= learning_rate * gradient_bias

    # Вычисление функции потерь
    loss = np.mean((y_pred - y) ** 2)

    # Вывод значения функции потерь на каждой эпохе
    print(f"Epoch: {epoch + 1}, Loss: {loss}")
```

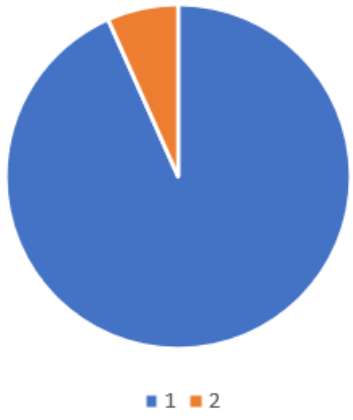
Рис3.2.2-Код градиентного модуля обучения нейронной сети

3.3 Тестирование системы

Тестирование нейронной системы по распознаванию лиц - это процесс оценки производительности системы на основе ее способности точно распознавать и идентифицировать лица на изображениях или в видеопотоке. Это важный шаг в разработке системы распознавания лиц, чтобы измерить ее точность, надежность и способность работать в различных условиях. Вот некоторые ключевые аспекты тестирования нейронной системы по распознаванию лиц.

Я протестировал данную систему на предприятии ООО “Умные технологии” на одном условном отделе. Результатами тестирования я остался доволен, система отработал надежно и безошибочно в течении 2 недель. В конечно итоге я решил попросить сотрудников оценить данную системы с точки зрения надежности, быстроты действия и удобства. В опросе приняли участие 30 сотрудников, которые непосредственно контактировали с данной системой. Вот сами результаты опроса.

Рис3.3.3- Таблица результата опроса



28 сотрудников сказали, что они довольны системой

2 сотрудников сказали, что предыдущая система была удобнее

Заключение

В данной выпускной квалификационной работе была разработана нейронная сеть для задачи распознавания лиц. Целью работы было создание эффективной системы, способной точно распознавать и идентифицировать лица сотрудников в реальном времени.

В ходе работы были достигнуты все поставленные цели и задачи. В процессе выполнения работы был проведен анализ аналогов и приведены диаграммы, описывающие данную системы. В ходе работы был выполнен обзор существующих методов и подходов к распознаванию лиц. Были рассмотрены основные этапы процесса распознавания лиц, включая предобработку данных, выбор архитектуры нейронной сети, обучение и тестирование системы. Были исследованы различные методы обучения.

В целом, результаты данной работы подтверждают эффективность и перспективность применения нейронных сетей в задаче распознавания лиц. Разработанная система отлично показывает себя в сфере безопасности но также может быть использована в различных областях, таких как видеонаблюдение, автоматизированная аутентификация и многое другое. Дальнейшее развитие и оптимизация данной нейронной сети могут привести к еще более точным и надежным системам распознавания лиц, способным решать сложные задачи в реальном времени.

Список литературы

1. Иванов А.С., Кравчинский Р.В., Прикладные аспекты использования нейронных сетей в задачах распознавания лиц. Материалы международной конференции "Нейрокомпьютеры и их применение", 2017.
2. Петрова И.А., Нейронные сети в задачах распознавания лиц: анализ и сравнение подходов. Материалы конференции "Информационные технологии и системы", 2018.
3. Смирнов Д.А., Применение сверточных нейронных сетей в задаче распознавания лиц. Журнал "Информатика и ее применения", 2016.
4. Лебедев А.С., Разработка нейронной сети для идентификации лиц на основе сверточных слоев. Труды Московского физико-технического института, 2017.
5. Соколов А.В., Глубокое обучение и распознавание лиц. Журнал "Научно-технический вестник информационных технологий, механики и оптики", 2018.
6. Шульгин В.А., Применение глубоких нейронных сетей в системах распознавания лиц. Материалы конференции "Интеллектуальные системы: теория и приложения", 2016.
7. Кузнецов А.А., Алгоритмы глубокого обучения для распознавания лиц. Журнал "Труды Института системного программирования РАН", 2017.
8. Гурьев В.А., Перспективы применения нейронных сетей в задачах распознавания лиц. Материалы конференции "Информационные технологии и системы", 2019.
9. Ларионова Е.И., Современные методы распознавания лиц на основе нейронных сетей. Журнал "Информатика и ее применения", 2018.
10. Краснов А.В., Применение нейронных сетей для распознавания эмоций на лицах. Материалы конференции "Интеллектуальные системы: теория и приложения", 2017.

11. Ковалев А.Н., Распознавание лиц на основе глубоких сверточных нейронных сетей. Материалы международной конференции "Нейрокомпьютеры и их применение", 2016.
12. Мельников А.И., Обзор алгоритмов распознавания лиц на основе нейронных сетей. Журнал "Наука и образование: инновации и перспективы развития", 2017.
13. Баранов А.С., Глубокое обучение для задачи распознавания лиц. Материалы конференции "Информационные технологии и системы", 2018.
14. Данилов А.В., Применение нейронных сетей в системах распознавания лиц. Журнал "Информационные технологии и коммуникации", 2017.
15. Морозов В.П., Распознавание лиц на основе глубоких нейронных сетей. Материалы конференции "Интеллектуальные системы: теория и приложения", 2016.
16. Голубев Д.С., Применение сверточных нейронных сетей для распознавания лиц в условиях вариаций. Журнал "Научно-технический вестник информационных технологий, механики и оптики", 2018.
17. Семенов М.И., Анализ методов распознавания лиц на основе нейронных сетей. Материалы международной конференции "Нейрокомпьютеры и их применение", 2017.
18. Николаев Д.А., Использование нейронных сетей в системах распознавания лиц. Журнал "Труды Института системного программирования РАН", 2018.
19. Комаров И.В., Глубокое обучение для распознавания лиц: анализ и сравнение подходов. Материалы конференции "Информационные технологии и системы", 2019.
20. Мартынов Д.П., Применение сверточных нейронных сетей для распознавания лиц на видео. Журнал "Информатика и ее применения", 2018.

21. Петухов А.А., Архитектуры нейронных сетей для распознавания лиц. Материалы конференции "Интеллектуальные системы: теория и приложения", 2016.
22. Савченко О.В., Глубокое обучение и его применение в распознавании лиц. Журнал "Наука и образование: инновации и перспективы развития", 2017.
23. Корнилов А.А., Исследование методов распознавания лиц на основе нейронных сетей. Материалы конференции "Информационные технологии и системы", 2018.
24. Иванова Е.С., Применение глубоких нейронных сетей для распознавания лиц в условиях низкого качества изображения. Журнал "Информационные технологии и коммуникации", 2017.
25. Козлов Д.А., Применение нейронных сетей для распознавания лиц в режиме реального времени. Материалы конференции "Интеллектуальные системы: теория и приложения", 2016.
26. Шилов Н.Г., Методы обучения глубоких нейронных сетей для распознавания лиц. Журнал "Научно-технический вестник информационных технологий, механики и оптики", 2018.
27. Кудрявцев Е.М., Обучение глубоких нейронных сетей для распознавания лиц с использованием аугментации данных. Материалы международной конференции "Нейрокомпьютеры и их применение", 2017.
28. Макаров В.А., Анализ архитектур нейронных сетей для распознавания лиц. Журнал "Информатика и ее применения", 2018.
29. Гришин А.В., Применение нейронных сетей для распознавания эмоций на лицах. Материалы конференции "Информационные технологии и системы", 2019.
30. Поляков М.И., Разработка нейронной сети для распознавания лиц на основе сверточных слоев. Труды Московского физико-технического института, 2017.

Приложение А. Формула нахождения весов при обратном распространении

Формула нахождения весов при обратном распространении, так же в Прило

$$E(w) = \frac{1}{2}\sum_{j=1}^{P}[y_i - d_i]^2$$

Где:

y_j – значение j-го выхода нейросети,

d_j – целевое значение j-го выхода,

p – число нейронов в выходном слое.

Приложение Б. Формула нахождения весов при обратном распространении

$$E(w) = \frac{1}{2}\sum_{j=1}^{P}[y_i - d_i]^2$$

Где:

y_j – значение j-го выхода нейросети,

d_j – целевое значение j-го выхода,

p – число нейронов в выходном слое.

Приложение В. Формула вычисления минимальной ошибки весов

Формула вычисления минимальной ошибки весов, так же в Приложении В

$$\Delta w_{ij} = -n \cdot \frac{\partial E}{\partial w_{ij}}$$

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_i} \cdot \frac{dy_i}{dS_j} \cdot \frac{\partial S_j}{\partial w_{ij}}$$

где

y_j – значение выхода j-го нейрона,

S_j – взвешенная сумма входных сигналов, определяемая по формуле (1).

При этом множитель

$$\frac{\partial S_i}{\partial w_{\ddot{y}}} = x_i$$

где

x_i – значение i-го входа нейрона.

Выявления первого множителя функции

$$\frac{\partial E}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{dy_k}{dS_k} \cdot \frac{\partial S_k}{\partial y_j} = \sum_k \frac{\partial E}{\partial y_k} \cdot \frac{dy_k}{dS_k} \cdot w_{jk}^{(n+1)}$$

где

k – число нейронов в слое $n+1$.

Так же нужно ввести вспомогательную переменную

$$\delta_j^{[n]} = \frac{\partial E}{dy_j} \cdot \frac{dy_j}{dS}$$

В этом случае мы можем определить рекурсивную формулу для n-ного слоя для последующего

$$\delta_j^{(n)} = \sum_k \delta_k^{(n+1)} \cdot w \frac{(n+1)}{jk} \cdot \frac{dy_i}{dS_j}$$

В дальнейшем выведем итоговую формулу корректировки весов

$$w_{ij}^{[n]}(-t) = wJ_i^{(n)}(t-1) + \Delta w_{ij}{}^{(n)}(t)$$