DOI 10.37882/2223-2966.2025.03.38

ИНТЕЛЛЕКТУАЛЬНАЯ САМООБУЧАЮЩАЯСЯ СИСТЕМА РАСПОЗНАВАНИЯ ЛИЦ

INTELLIGENT SELF-LEARNING FACIAL RECOGNITION SYSTEM

A. Rusakov

Summary. The intelligent self-learning recognition system is developed in Python and has a web interface in JavaScript React. The face recognition subsystem is based on the OpenCV, Dlib, Deepface libraries. This system is capable of not only recognizing faces, but it also has several mechanisms for self-learning, which distinguishes this system from numerous existing analogues. The article provides a description of the developed software suite and the basic principles of its operation.

Keywords: face recognition, artificial intelligence, image processing, machine learning, information security.

Русаков Алексей Михайлович
старший преподаватель,
МИРЭА — Российский технологический университет
rusakov_a@mirea.ru

Аннотация. Интеллектуальная самообучающаяся система распознавания разработана на Python и имеет веб интерфейс на JavaScript React. Подсистема распознавания лиц основывается на библиотеках OpenCV, Dlib, Deepface. Данная система способна не только распознавать лица, у неё также имеется ряд механизмов для самообучения, что выгодно отличает данную систему от многочисленных имеющихся аналогов. В статье приводится описания разработанного комплекса программных средств и основных принципов его работы.

Ключевые слова: распознавание лиц, искусственный интеллект, обработка изображений, машинное обучение, информационная безопасность.

Введение

Технологии распознавания лиц стремительно развиваются и становятся неотъемлемой частью нашей повседневной жизни. Они используются в системах безопасности, смартфонах, умных городах, банковской сфере и даже в розничной торговле для персонализации сервиса. Однако традиционные системы распознавания лиц имеют ограничения: их точность снижается при изменении внешности человека, варьирующемся освещении или нестандартных ракурсах [1-3].

Самообучающиеся системы распознавания лиц предлагают новый подход к решению этих проблем [2]. В отличие от статических алгоритмов, такие системы способны адаптироваться к новым условиям, обучаясь на поступающих данных. Они используют методы глубокого обучения, автоматически совершенствуя модели распознавания, минимизируя ошибки и повышая точность идентификации. Именно этому и посвящена данная работа.

Описание архитектуры системы

Перед разработанной интеллектуальной системой распознавания лиц были поставлены задачи, связанные не только с повышением качества распознавания лиц и разработкой новых алгоритмов и методов, а в первую очередь в построении информационно-аналитической системы мониторинга на основе современных методов и алгоритмов машинного обучения с механизмами самообучения. Таким образом перед разрабатываемой системой ставятся следующие основные задачи:

- Учет посещения занятий студентами на основе алгоритмов распознавания лиц в автоматическом режиме. Повышение мотивации к обучению за счет выявления проблем с посещаемостью у студентов на ранних стадиях.
- Нахождение на территории образовательного учреждения только авторизованных посетителей и моментально реагировать на появление нежелательных лиц как в здании, так и в периметре.
- Обучение и экзаменация. Решаются вопросы контроля за соблюдением установленных правил во время экзаменов и аттестаций.

На рисунке 1 представлена архитектура информационно-аналитической системы видео мониторинга.

Информационно-аналитическая система (ИАС) состоит из выделенного сервера и хостинга, отделенных межсетевыми экранами. Информационное взаимодействие между модулями ИАС происходит через API-интерфейсы.

На хостинге располагается веб-портал, на котором реализован функционал личного кабинета пользователя ИАС; сервис оповещений; сервис проверки корректности данных уведомления пользователей для их анализа в режиме онлайн с помощью методов машинного обучения и библиотеки экспертных данных; сервис аналитики данных; экспертная система видео аналитики.

На выделенном сервере располагается система загрузки открытых данных, система хранения актуальных данных, система видео аналитики.

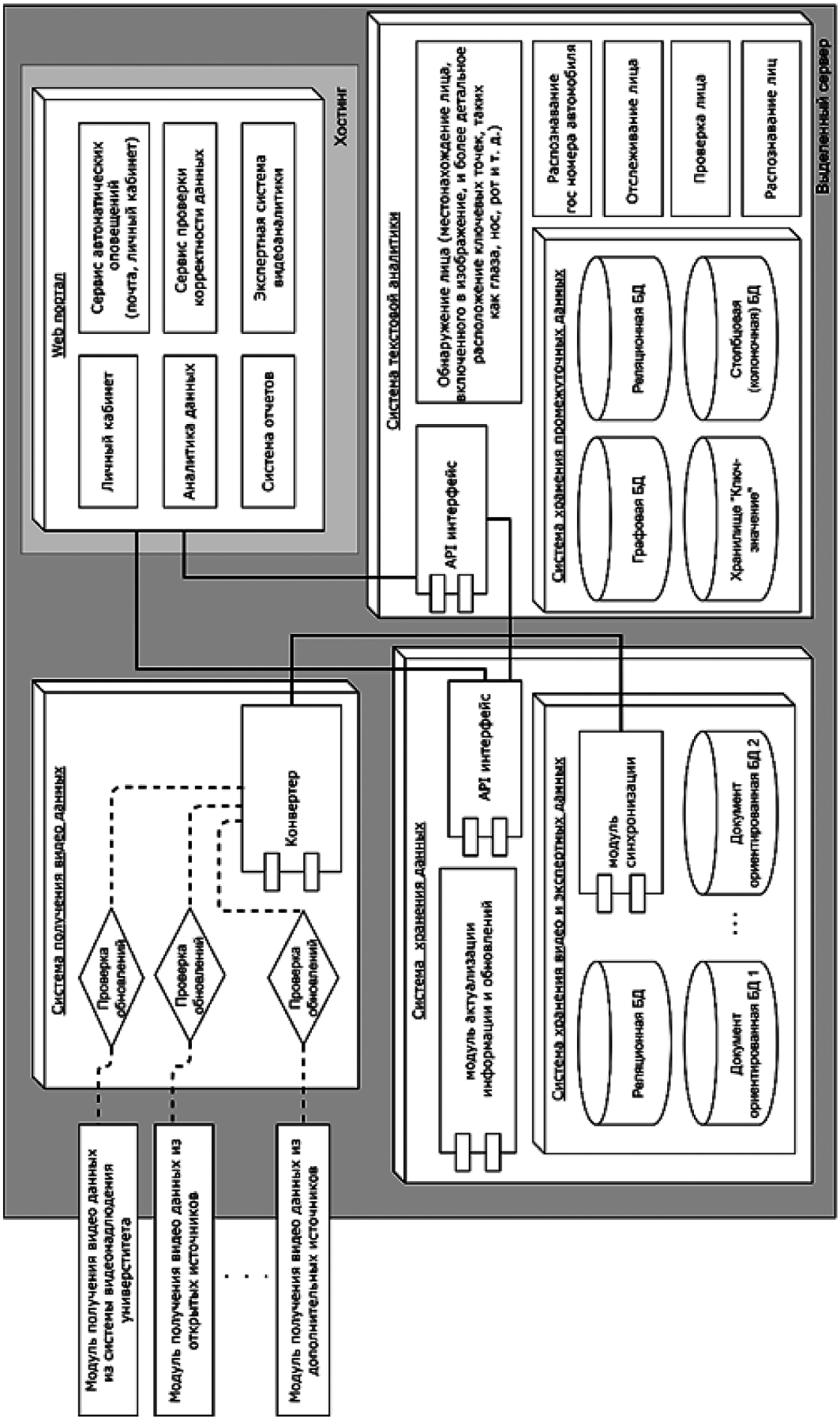

Рис. 1. Архитектура информационно-аналитической системы видео мониторинга

Система загрузки открытых данных представляет собой высоко производительную систему, обеспечивающую поиск и загрузку необходимых для анализа данных из различных источников.

Система хранения актуальных данных содержит в себе подсистему хранения открытых и экспертных данных, состоящую из реляционной базы данных для хранения библиотеки экспертных данных, видео архива. Для хранения экспертных данных предлагается использовать документ ориентированные NoSQL СУБД. Для организации своевременного обновления и актуализации данных используются модуль актуализации информации и обновлений, а также модуль синхронизации, который через конвертер производит проверку и синхронизацию обновлений с базами данных.

Проводимые исследования показали, что для успешного распознавания лица требуется, чтобы лицо было представлено не менее 160 пикселями, приходящимися на овал лица, и в идеале не менее 50 пикселей приходилось на расстояние между глазами. Таким образом имеющейся системы видеонаблюдения на начальном этапе вполне достаточно.

Аналитическим ядром информационной системы является система видео аналитики, для реализации которой используется язык программирования Python с библиотеками обработки изображений. Задачи, связанные с распознаванием лиц, включают в себя следующее:

- Обнаружение лица (предварительно определите местонахождение лица, включенного в изображение, и более детальное расположение ключевых точек, таких как глаза, нос, рот и т. д., Обложка изделия является типичным результатом обнаружения лица);
- Отслеживание лица (отслеживание изменений положения лица в видео);
- Проверка лица (введите два лица, чтобы определить, принадлежат ли они одному человеку);
- Распознавание лиц (введите лицо, чтобы определить, какой человек принадлежит ко всем записям в базе данных лиц);

Система распознавания лиц основывается на библиотеках OpenCV, Dlib, Deepface, а также на фреймворке mediapipe. На сегодняшний момент для распознавания лиц также используются: VGG-Face, Google FaceNet, OpenFace, Facebook DeepFace, DeepИД, ArcFace и Dlib. Далее опишем наиболее важные библиотеки, использующиеся в проекте.

Библиотека OpenCV (Open Source Computer Vision Library: http://opencv.org) — это библиотека с открытым исходным кодом, которая включает в себя несколько сотен алгоритмов компьютерного зрения.

Библиотека Dlib — кроссплатформенная программная библиотека общего назначения, написанная на языке программирования C++. Основные функции: обработка изображений, классификация и распознавания фрагментов изображений с помощью алгоритмов машинного обучения.

Библиотека Deepface — это достаточно удобная библиотека для распознавания лиц и анализа атрибутов лица (возраст, пол, эмоции и раса) для Python. Основным между DeepFace и другими решениями состоит в том, что DeepFace использует метод выравнивания перед обучением нейронной сети. После выравнивания лица элементы области лица фиксируются на определенных пикселях.

Библиотека MediaPipe — кроссплатформенный фреймворк для настраиваемых решений в сфере машинного обучения для потокового мультимедиа контента.

Описание основных принципов работы системы

Программное обеспечение системы разработано на основе сервисной архитектуры и состоит из следующих сервисов. Полные исходные коды доступные в свободном доступе [4, 5]. Для запуска системы необходимо последовательно запустить каждый из сервисов в следующем порядке: «app.py», «capture_streamer.py», «process.py», «cleanerDB.py», «bot.py». Для увеличения производительности при наличии необходимых аппаратных ресурсов возможен запуск нескольких сервисов «process.py».

В репозиториях [4, 5] не содержится исходный код сервиса «video_enhance.py», поскольку реализация на языке программирования Python оказалась недостаточно производительной. Данный сервис можно заменить на любой другой программный модуль, способный обрабатывать изображения в нужном формате с целью фильтрации помех и улучшении изображений лиц. К сожалению, автором не удалось найти стабильно работающий программный продукт, полностью подходящий для этой задачи. Следует отметить, что система устойчиво работает и без этого сервиса.

В одном из финальных релизов были созданы docker образы всех сервисов. Однако для более наглядного представления работы сервисов в репозиториях приведены исходные коды без упаковки в docker образы. Для более удобной установки системы на любой компьютер добавлен файл «requirements.txt» для установки всех необходимых библиотек. Каждый из сервисов может работать как на отдельном компьютере, так и на одном компьютере. Данное обстоятельство позволяет существенно уменьшить время обработки и распознавания изображений и существенно снизить время отклика системы в целом.

Опишем функциональное назначение и принцип работы каждого из взаимодействующих сервисов.

Сервис «app.py». Данный сервис реализует API интерфейс отображения информации на web портале. Из этого интерфейса через точки входа «/dash/<int:id>», «/photo/ <path: path» и «/capture/ <path: path>» происходит выдача информации о последних распознанных лицах для отображения информации для web клиента. Сервис каждые 5–7 секунд обновляет данные из таблицы БД zdash и хранит в памяти данные о уже распознанных лицах. Описание формата таблиц БД приведено в таблице 1. Переменная «ID» — это номер данных распознанных лиц, начиная от 0, который постоянно увеличивается. Если новых данных нет, то в ответ на запрос, выводится сообщение «not_found».

Данный сервис рекомендуется переделать исключая цикличные запросы с помощью технологии «long Pooling». Поменять формат данных чтобы исключить инкрементацию переменной ID. Для этого потребуется добавить точку входа с количеством распознанных лиц. Однако для исследовательского стенда даже в таком исполнение программное средство показывало устойчивую работу.

Сервис «bot.py». Данный сервис реализует функционал добавления фото пользователей и управления системой. В качестве платформы для взаимодействия с системой в самом первом приближении выбран бот в мессенджере Telegram. Удобство представлено, тем, что не нужно регистрировать хостинг и таким образом система может работать на одном компьютере. В сервисе реализован механизм защиты на основе пароля. Также имеются команды для добавления новых пользователей в систему. Возможность полной перезагрузки системы. Очистка всех данных.

Сервис «capture_streamer.py». Данный сервис реализует захват изображений лиц и их обрезку. В этом сервисе реализован функционал отображения распознанных изображений. Изначально было два разны сервиса, отдельно для отображения изображения – «streamer» и сервис захвата и обрезки лиц «capture». В представленном релизе [4, 5] эти два сервиса объединены в один.

Следует отметить, что для отображения распознанных лиц возможно применить стриминговые технологии такие как RSTP и другие (было опробовано в первых релизах). Однако при использовании стриминговых технологий связанных с передачей данных по сети появляется задержка около 2 секунд. Данный факт делает работу системы не комфортной. Найдено решение, за счет использования библиотеки «pyvirtualcam», которая позволяет рендерить изображения в виртуальную веб камеру из проекта OBS Studio. Данное решение позволяет полностью избавиться от задержек. При этом если разделить этот сервис на два сервиса, то задержек также не будет.

В сервисе реализована логика для измерения расстояния между лицами которая позволяет отображать подпись к лицу даже если нет доступных качественных изображений с удовлетворительной точностью распознавания. Все подготовленные к распознаванию изображения лиц передаются в БД в таблицу z1frame (см. описание в табл. 1).

В данном сервисе возможна реализация проверки нахождения новых изображений для внесения в базу данных эмбедингов (сжатых портретов). Этот функционал вызывается из библиотеки «zdata.py», которая может быть установлена в любом окружении для любого из используемых сервисов.

Сервис «cleanerDB.py». Данный сервис реализует функционал проверки ошибок и очистки устаревших изображений. Чтобы разгрузить все остальные сервисы этот функционал был вынесен в отдельный сервис.

Сервис «playsound.py». Данный сервис реализует функционал проигрывания звуковых сопровождений для найденных и распознанных в текущий момент изображений лиц. Реализованы механизмы позволяющие избежать постоянных повторений звукового сопровождения, в случае если лицо постоянно находится в кадре. Данная задержка настраивается. Условно можно назвать наиболее подходящим значением данной задержки в 45 секунд.

Сервис «process.py». Данный сервис реализует непосредственно процесс распознавания лиц. В открытых репозиториях [4, 5] приводится функционал распознавания лиц на основе эмбедингов. Последовательно из таблицы БД «z1frame» извлекаются данные для получения эмбединга. После извлечения данных из таблицы БД «z1frame» – они удаляется. Получение эмбединга в среднем занимает около 0,2 секунд. Полученный эмбединг записывается в таблицу БД «zdata». Во время работы сервиса происходит сравнение полученных эмбедингов и тех эбедингов, которые есть в таблице БД «zemb». Если найдено совпадение, то добавляются соответствующие данные в таблицу БД «zdash». Процедура сравнения вызывается получения эмбединга и сравнения эмбедингов вызывается из библиотеки «zdata.py».

В расширенной версии системы автором реализован функционал накопления изображений для последующего обучения нейронной сети, что позволяет существенно уменьшить время распознавания. Экспериментально получено то, что распознавание с помощью нейронной сети в 50 раз быстрее чем процесс получения эмбединга.

При этом обучение нейронной сети, выбор её архитектуры, оценка погрешностей является достаточно трудоемкой задачей. Данная задача является научно-исследовательской, и будет интересна при изучении современных технологий машинного обучения.

Сервис «video_enhance.py». Данный сервис реализует функционал улучшения изображений по цвета передаче, балансу белого, яркости и контрастности. Реализован механизм приближения к лицу за счет цифрового увеличения изображений. Следует отметить, что данный сервис достаточно требователен к ресурсам оборудования и сопоставим в этом отношении с сервисом «capture_streamer.py». В репозиториях [4,5] данный функционал не используется.

Сервис «zdata.py». Данный сервис реализует функционал получения сжатых портретов изображений (эмбедингов) для их загрузки во внутреннюю базу данных системы. Реализован механизм сравнения эмбедингов. Данные заносятся в таблицу БД «zemb». Запускать данный сервис не требуется, так как его запускают сервисы «bot.py», «capture_streamer.py» и «process.py». В технических целях если просто запустить сервис «zdata.py» без параметров, то происходит очистка БД таблицы «zemb». Сервис позволяет загружать наборы эталонов изображений лиц в виде простых файлов. Далее опишем формат хранения информации изображений лиц.

Описание формата хранения информации изображений лиц

Обработка изображений происходит в режиме реального времени. Все взаимодействие между сервисами выполнено через БД. Для загрузки новых изображений в БД используется библиотека «zdata.py», которую также можно запустить как отдельный сервис. Таким образом библиотека «zdata.py» является библиотекой для обработки изображений и должна быть установлена для сервисов «app.py», «bot.py», «capture_streamer.py», «process.py». При загрузке данных необходима фотография в формате «JPEG» на которой изображение лица занимает не менее 200x300 пикселей. На каждом изображении должно быть только одно лицо. При запуске приложения в первый раз выполняется процедура получения эмбедингов вызываемая из библиотеки «zdata.py». Далее получение эмбедингов выполняется исходя из необходимостей работы системы через сервисы «bot.py» и «capture_streamer.py» (при добавлении новых изображений в ручном режиме.

На рисунке 2 представлен паромер содержимого папки для загрузки данных о распознаваемых лицах в БД. Каждая из подпапок содержит файлы: «data.txt» (менять имя файла не допускается), файл с изображением лица и файл со звуковым приветствием. Файл со звуковым приветствием не является обязательным и если его нет в папке, то будет звучать общее звуковое приветствие для распознанных лиц. Если файл звукового сопровождения есть, то он должен быть прописан в файле «data.txt» в четвертой строке.

Описание формата файла «data.txt» имеет следующий вид (пример показан на рисунке 2):

- В первой строке файла «data.txt» должна быть информация, которая будет отображаться большим шрифтом рядом с рамкой распознанного изображения лица.

Рис. 2. Формат хранения исходных изображений лиц для распознавания системы «Персона-ИД»

- Во второй строке файла «data.txt» должна быть информация с должностью или описанием, которая будет отображаться обычным шрифтом рядом с рамкой распознанного изображения лица.
- В третьей строке файла «data.txt» должна быть информация о имени файла с изображением лица.
- В четвертой строке файла «data.txt» должна быть информация о имени файла со звуковым приветствием в формате «MP3».

Описание структуры таблиц в базе данных

Для работы системы были опробованы разные типы и виды БД: Tarantool, Redis, PostgreSql. Наиболее быстрым оказалось решение с использование Redis и структур «Sorted Sets». Довольно неплохие результаты были достигнуты с использование БД «Tarantool. Однако для простоты и удобства запуска в [4, 5] даны исходные коды для БД PostgreSql. Там же для удобства установки БД даны конфигурационные файлы Docker и резервная копия пустых таблиц.

Программное обеспечение функционирует, используя четыре таблицы: «z1frame», «zemb», «zdata», «zdash». Описание и команды для их создания проводятся в таблице 1.

Результаты апробации

Разработанная система состоит из веб клиента и серверной части. Веб клиент можно оформить под разные задачи. На рисунках 3 и 4 приведен пример использования данного программного средства на первой всероссийской научно-практической конференции с международным присутствием «Россия в Десятилетии ООН наук об океане» 2022 год г. Москва в качестве средства регистрации участников.

Для всех приглашенных участников конференции были заранее загружены их фотографии в базу данных системы. При получении Бейджа каждый из участников услышал звуковое сопровождение с приветствием. Конференция продлилась 4 дня и в последний день данных фотографий лиц было собрано достаточно чтобы обучить нейронную сеть и распознавать присутствующих в кадре быстрее, чем с использованием эмбедингов.

На рисунке 4 представлен общий вид места регистрации участников конференции с работающей системой.

Таблица 1.

Список таблиц в БД с их описанием

Наименование таблицы	Описание таблицы	SQL команда для создания таблиц
z1frame	Таблица, в которую добавляется обрезанное изображение лица сервисом «capture_streamer.py». Из этой таблицы обрезанное изображение лица забирает в обработку сервис «process.py».	CREATE TABLE public. z1frame (id bigint NOT NULL, frame bytea, milliseconds numeric, timestr character (40), bbox text);
zemb	В таблице храниться эмбединги изображений лиц с их описанием для распознавания пользователей системы. Данная таблица используется сервисом «process.py». В эту таблицу добавляются данные о сервисом «zdata.py»/	CREATE TABLE public. zemb (id bigint NOT NULL, emb text, filename text, name text, «desc» text, sound text);
zdata	Данная таблица используется сервисом «process.py» для определения какие лица присутствуют в кадре.	CREATE TABLE public. zdata (id bigint NOT NULL, zjson text, milliseconds bigint, timestr character(40));
zdash	В таблице хранится информация о тех лицах, которые находятся в кадре в текущий момент. Данные из этой таблицы используются сервисом «app.py» для формирования информации, выводимой на веб портал.	CREATE TABLE public. zdash (id bigint NOT NULL, milliseconds bigint, timestr character(40), photo text, name text, capture text, name_id bigint);

Заключение

Отлаженный исходный код системы доступен в свободном доступе в [4, 5]. К сожалению, в статье не удалось отобразить большую часть опыта при её разработке. Автор всегда готов ответить на возникающие вопросы при реализации схожих проектов. Данный проект до сих пор продолжает свою работу в различных интерпретациях для обучения студентов современным технологиям. Например, в [3] приводится одна из возможных реализаций данного проекта с использованием одноплатного компьютера.

Рис. 3. Использование системы для регистрации участников на конференции

Рис. 4. Использование системы для регистрации участников на конференции (общий вид)

ЛИТЕРАТУРА

1. Yarovoy P.A. Applying intelligent systems for face recognition on images / P.A. Yarovoy // Молодежь. Общество. Современная наука, техника и инновации. – 2021. — No. 20. — P. 275–277. — EDN SSPSKZ.
2. The Study of Mathematical Models and Algorithms for Face Recognition in Images Using Python in Proctoring System / Sh. U. Niyazbekova, A.A. Nurpeisova, A. Shaushenova [et al.] // Computation. — 2022. — Vol. 10, No. 8. — DOI 10.3390/computation10080136. — EDN EBUQYC.
3. Колмаков Н.П. Система распознавания лиц для одноплатных компьютеров на основе методов машинного обучения / Н.П. Колмаков, А.М. Русаков // Научный взгляд в будущее. — 2021. — Т. 1, № 21. — С. 50–63. — DOI 10.30888/2415–7538.2021–21-01-027. — EDN PRPEHK.
4. Исходный код интеллектуальная самообучающаяся система распознавания лиц «Персона ИД» URL: https://github.com/RusAI84/PersonaID/
5. Исходный код интеллектуальная самообучающаяся система распознавания лиц «Персона ИД» (зеркало на территории России) URL: https://gitverse.ru/RusAI84/PersonaID/

Журнал «Современная наука: актуальные проблемы теории и практики»