**ДОКУМЕНТАЦИЯ ПРОЕКТА «АВТОМАТИЧЕСКИЙ АНАЛИЗ ИЗОБРАЖЕНИЙ: ОБНАРУЖЕНИЕ ОБЪЕКТОВ И РАСПОЗНАВАНИЕ ТЕКСТА. СИСТЕМА VISTA – VISUAL AND TEXTUAL ANALYSIS»**

Подготовили:

Тупович Назар Сергеевич

Сорока Тимофей Денисович

Сидоров Данил Дмитриевич

Козаченко Николай Максимович

Хомяк Елизавета Дмитриевна

Донецк, 2025

**СОДЕРЖАНИЕ**

1. Описание проблемы, цели и задачи проекта ………………………….... 3
2. Архитектура системы …………………………………………………..... 5
3. Используемые технологии …………………………………………….… 7
4. Структура кода, классов и методов ……………………………..………. 9
5. Особенности реализации ………………………………...……………… 11
6. Инструкция по запуску ………………………………………………….. 13
7. Заключение ………………………………………………………………. 15

**ОПИСАНИЕ ПРОБЛЕМЫ, ЦЕЛИ И ЗАДАЧИ ПРОЕКТА**

В современном мире наблюдается огромный рост объемов визуальной информации. Каждый день создаются и распространяются миллиарды изображений и видео, содержащих огромный потенциал для извлечения ценной информации. Однако, ручной анализ такого количества визуальных данных является чрезвычайно трудоемким, затратным и неэффективным.

Существующие инструменты для анализа изображений часто ограничены в своих возможностях, требуя ручной настройки, не обеспечивают достаточную точность или не способны одновременно обрабатывать различные типы информации. Это создает ряд проблем, такие как: задержка в получении информации, высокая стоимость, ограниченная масштабируемость, ограниченные возможности.

Эти проблемы приводят к тому, что значительная часть визуальной информации остается неиспользованной, лишая организации ценных знаний и конкурентных преимуществ. Необходимость в автоматизированных и эффективных решениях для анализа изображений становится все более острой.

Целью проекта является разработка высокоэффективной и масштабируемой системы автоматического анализа изображений "VISTA – Visual and Textual Analysis", способной автоматически обнаруживать объекты, распознавать текст и предоставлять структурированные данные.

**Задачи проекта**:

1. Интеграция модулей обнаружения объектов и OCR в единую систему. Разработка механизма для объединения результатов обнаружения объектов и распознавания текста для создания более полной и структурированной картины содержимого изображения.

2. Оптимизация производительности системы для обработки больших объемов данных. Реализация методов для повышения скорости обработки изображений и масштабирования системы для работы с большими объемами данных.

3. Разработка API для интеграции с другими системами. Создание API, позволяющего интегрировать систему VISTA с другими приложениями и сервисами.

4. Оценка производительности и точности системы на тестовых наборах данных. Проведение всестороннего тестирования системы для оценки ее производительности и точности на различных типах изображений и с использованием различных метрик оценки.

5. Создание документации. Подготовка подробной документации, описывающей установку, настройку и использование системы VISTA.

Планируемые результаты:

• Разработана функционирующая система VISTA, способная автоматически анализировать изображения, обнаруживать объекты и распознавать текст.

• Реализован высокоточный модуль обнаружения объектов с точностью не менее 0.8 на тестовом наборе данных.

• Реализован модуль распознавания текста (OCR) с уровнем ошибок не более 0.1 на тестовом наборе данных.

• Разработан API для интеграции системы с другими приложениями.

• Создана подробная документация.

• Проведено тестирование системы на тестовых наборах данных.

**АРХИТЕКТУРА СИСТЕМЫ**

Архитектура проекта представляет собой веб-приложение на Flask, которое позволяет пользователям загружать изображения, обнаруживать объекты и текст на них, сохранять метаданные и результаты в базу данных, а также выполнять поиск по этим данным

**Основные компоненты**:

1. Flask. Веб-фреймворк, используемый для создания веб-приложения.
2. SQLAlchemy. для работы с базой данных SQLite.
3. XML. Используется для хранения метаданных изображений.

Модули для обработки изображений включают функции для обнаружения объектов и распознавания текста (такие как OpenCV).

**Структура проекта**:

1. app.py представляет собой основной файл приложения, содержащий маршруты и логику обработки запросов.
2. config.py представляет собой файл конфигурации, содержащий настройки приложения, такие как путь к папке загрузки, разрешенные расширения файлов и секретный ключ.
3. database.py представляет собой модуль для работы с базой данных, включая создание таблиц, добавление и поиск данных.
4. metadata.py представляет собой модуль для работы с метаданными изображений, включая их сохранение в формате XML.
5. models/ представляет собой папку, содержащую модули для обработки изображений (обнаружение объектов и распознавание текста).
6. utils/ представляет собой папку, содержащую вспомогательные модули для работы с метаданными и базой данных.

**Работа с базой данных**:

1. ImageDB представляет собой класс, отвечающий за взаимодействие с базой данных. Включает методы для добавления изображений и поиска по тексту или объектам.
2. Image представляет собой модель данных, представляющая таблицу images в базе данных. Содержит поля для имени файла, объектов и текста.

**Работа с метаданными**:

save\_metadata представляет собой функцию, которая сохраняет метаданные изображения (объекты и текст) в XML-файл. Метаданные хранятся в папке static/uploads/metadata.

**Обработка изображений**:

1. detect\_objects представляет собой функцию для обнаружения объектов на изображении.
2. extract\_and\_correct\_text представляет собой функцию для распознавания и коррекции текста на изображении.

Шаблоны:

1. Index.html представляет собой главную страницу для загрузки изображений.
2. results.html представляет собой страницу для отображения результатов обработки изображения

**ИСПОЛЬЗУЕМЫЕ ТЕХНОЛОГИИ**

Python является основным **языком программирования**, используемым для разработки системы VISTA. Он выбран благодаря своей широкой распространенности в области машинного обучения и компьютерного зрения, наличию множества библиотек и фреймворков, а также простоте и удобству разработки. Python используется для объединения модулей обнаружения объектов, распознавания текста (OCR), интеграции, API и логики обработки данных.

**Библиотеки и фреймворки**:

1. YOLOv8. Для обнаружения объектов на изображениях используется модель YOLOv8. YOLOv8 представляет собой современную модель обнаружения объектов, отличающуюся высокой скоростью и точностью. Она позволяет эффективно обнаруживать различные объекты на изображениях в режиме реального времени или близком к нему.
2. EasyOCR. Для оптического распознавания символов (OCR) используется библиотека EasyOCR. EasyOCR - это простой в использовании, но мощный инструмент OCR, поддерживающий множество языков и обеспечивающий высокую точность распознавания текста на изображениях. EasyOCR позволяет извлекать текстовую информацию из различных источников, включая фотографии и сканированные документы.
3. Yandex.Speller. Для проверки орфографии и исправления ошибок в распознанном тексте используется API Yandex.Speller. Yandex.Speller позволяет повысить качество распознанного текста, исправляя опечатки и грамматические ошибки. Использование этого инструмента особенно важно для приложений, требующих высокой точности и надежности текстовой информации.
4. Flask. Фреймворк Flask (версия 3) используется для разработки API, обеспечивающего доступ к функциональности системы VISTA. Flask позволяет быстро и легко создавать веб-приложения и API на Python, обеспечивая гибкость и масштабируемость.

**СТРУКТУРА КОДА, КЛАССОВ И МЕТОДОВ**

1. app.py - основной файл приложения Flask, содержащий маршруты и логику обработки запросов.

Классы и методы.

Маршруты:

1. /: обрабатывает загрузку изображений, вызывает функции для обнаружения объектов и распознавания текста, сохраняет метаданные и результаты в базу данных.
2. /search обрабатывает поиск по базе данных по тексту или объектам.

Методы:

1. index() обрабатывает главную страницу и загрузку изображений.
2. search() обрабатывает поисковые запросы.
3. download() отправляет файл для скачивания.
4. get\_metadata() возвращает XML-файл с метаданными.

2. config.py - файл конфигурации, содержащий настройки приложения.

3. database.py - модуль для работы с базой данных (SQLite через SQLAlchemy).

Классы и методы:

1. Класс Image - модель таблицы images в базе данных.
2. Класс ImageDB - управляет взаимодействием с базой данных.

Методы:

1. init() инициализирует подключение к базе данных и создает таблицы.
2. add\_image() добавляет запись о изображении в базу данных.
3. search\_images() выполняет поиск по тексту или объектам.

4. metadata.py - модуль для работы с метаданными изображений в формате XML. Метод: save\_metadata() сохраняет метаданные (объекты и текст) в XML-файл.

5. models/ - папка с модулями для обработки изображений (предположительно).

Функции:

1. detect\_objects() обнаружение объектов на изображении.
2. extract\_and\_correct\_text() распознавание и коррекция текста на изображении.

6. utils/ - папка с вспомогательными модулями.

Модули:

1. metadata.py обеспечивает работу с метаданными.
2. database.py обеспечивает работу с базой данных.

**Описание взаимодействия**:

1. Пользователь загружает изображение через главную страницу.
2. Изображение обрабатывается.
   1. Обнаружение объектов (detect\_objects).
   2. Распознавание текста (extract\_and\_correct\_text).
3. Результаты сохраняются.
   1. В базу данных (ImageDB.add\_image).
   2. В XML-файл (save\_metadata).

**ОСОБЕННОСТИ РЕАЛИЗАЦИИ**

Используется предобученная модель YOLOv8x, которая обеспечивает высокую точность обнаружения объектов. Модель поддерживает множество классов объектов и работает быстро даже на больших изображениях.

Результаты обнаружения объектов возвращаются в виде строки, содержащей метку объекта и время обработки изображения. Это упрощает дальнейшую обработку и хранение данных.

EasyOCR настроен на распознавание текста на русском языке, что делает его подходящим для русскоязычных пользователей.

После распознавания текста используется Yandex.Speller для исправления опечаток. Это улучшает качество распознанного текста, особенно в случаях с плохим качеством изображения или сложным шрифтом.

Flask обрабатывает загруженные изображения синхронно, что может быть ограничением для больших файлов или сложных задач.

Реализованы маршруты для загрузки, скачивания и получения метаданных изображений, что делает приложение удобным для пользователей.

Используется SQLite для хранения информации о загруженных изображениях. Метаданные (объекты и текст) сохраняются в XML-файлах. Это позволяет легко экспортировать и анализировать данные, но может быть неэффективным для больших объемов

Код разделен на модули (Flask, база данных, обработка изображений), что упрощает добавление новых функций и поддержку проекта.

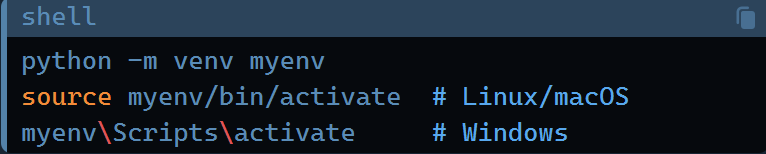
OpenCV используется для работы с изображениями, что обеспечивает высокую производительность и поддержку различных форматов файлов.

Для удобной обратной связи и технической поддержки сайта был реализован телеграмм-бот с перенаправлением сообщений пользователя администратору.

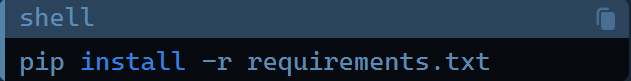
**ИНСТРУКЦИЯ ПО ЗАПУСКУ**

В папку куда вы выгрузили все файлы скачайте: https://docs.ultralytics.com/ru/models/yolo12/#detection-performance-coco-val2017

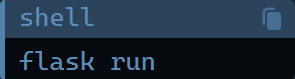
Для запуска требуется перейти в консоль. Перед запуском кода нужно создать папку и перейти в директорию. Далее выполняются следующие команды:



Далее нужно перетащить в папку проекта файл requirements.txt со всеми вспомогательными библиотеками



После установки всех пакетов, нужно перетащить содержимое архива туда же, куда и requirements.txt. После нужно открыть консоль и написать



После можно приступать к работе с системой.

**ЗАКЛЮЧЕНИЕ**

Настоящая документация предоставляет всестороннее описание системы VISTA – Visual and Textual Analysis, разработанной для автоматического анализа изображений с целью обнаружения объектов и распознавания текста. Этот документ охватывает ключевые аспекты системы, начиная с постановки проблемы и определения целей проекта, и заканчивая архитектурой системы, используемыми технологиями, особенностями реализации, инструкциями по использованию.

В процессе разработки системы VISTA были успешно решены поставленные задачи, в частности реализация системы для обнаружения объектов и распознавания текстов на изображении, разработка удобного пользовательского интерфейса для взаимодействия с системой, а также бот для технической поддержки.

Результаты проведенного тестирования подтверждают работоспособность и эффективность системы VISTA, а также ее соответствие требованиям, предъявляемым к системам автоматического анализа изображений. Разработанная система VISTA представляет собой перспективное решение для автоматизации процессов, связанных с анализом визуальных данных, и обладает значительным потенциалом для дальнейшего развития и совершенствования. В будущем планируется расширение списка распознаваемых объектов, улучшение алгоритмов OCR, добавление новых функций и возможностей, а также интеграция с другими платформами и сервисами.