Описание технической архитектуры программного обеспечения «Система поиска GigaFlex КОЗ 1 Автономный поиск»

Термины и определения

БВС (беспилотные воздушные суда): Автономные или дистанционно управляемые воздушные суда, не предназначенные для нахождения человека на борту во время полета.

Консоль: Интерфейс командной строки, позволяющий пользователю взаимодействовать с программным обеспечением без использования графического интерфейса.

Технологии искусственного интеллекта: Технологические решения и алгоритмы, имитирующие человеческий интеллект, используемые для автоматизации различных процессов, включая распознавание образов.

JetPack: Комплексный набор программного обеспечения от NVIDIA, включающий операционную систему, библиотеки, APIs и инструменты разработки для платформ NVIDIA Jetson.

CUDA Toolkit: Набор инструментов разработки и библиотек от NVIDIA, позволяющий создавать приложения с использованием параллельных вычислений на графических процессорах NVIDIA.

cuDNN: Библиотека примитивов для глубокого обучения от NVIDIA, оптимизированная для работы с GPU и предназначенная для ускорения операций в нейронных сетях.

TensorRT: Платформа для оптимизации моделей глубокого обучения, позволяющая ускорить вывод (инференс) на GPU NVIDIA путем оптимизации вычислительного графа.

Драйверы NVIDIA: Программное обеспечение, обеспечивающее взаимодействие между операционной системой и аппаратным обеспечением NVIDIA, предоставляющее доступ к функциям GPU.

Docker: Платформа программного обеспечения, предназначенная для автоматизации развертывания и управления приложениями в средах виртуализации на уровне операционной системы. Она позволяет упаковывать приложения вместе со всеми их зависимостями в стандартизированные блоки, называемые контейнерами. Эти контейнеры могут быть легко перенесены и

запущены на любой системе, поддерживающей Docker, обеспечивая тем самым высокую степень переносимости и эффективность. Docker использует легковесную виртуализацию, обеспечивая быстрое развертывание и высокую эффективность использования ресурсов по сравнению с традиционными виртуальными машинами.

Python: Высокоуровневый язык программирования, ориентированный на повышение производительности разработчика и читаемости кода. Python поддерживает несколько парадигм программирования, включая структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное программирование.

Intersection over Union (IoU): Метрика, используемая для оценки точности алгоритмов объектного обнаружения, измеряющая степень перекрытия между предсказанным и реальным ограничивающим прямоугольником. Значения варьируются от 0 (отсутствие перекрытия) до 1 (полное совпадение).

Пороговое значение (Threshold): Значение, используемое для определения, должен ли объект быть классифицирован как интересующий нас объект или отброшен. Используется для управления уровнем уверенности алгоритма в правильности классификации объекта.

Ограничивающий прямоугольник (Bounding Box): Прямоугольник, который рисуется вокруг обнаруженного объекта на изображении, показывая расположение и размеры объекта.

YOLO (You Only Look Once): является современным алгоритмом в области компьютерного зрения, специализирующимся на задачах детекции и классификации объектов на изображениях. Этот алгоритм уникален благодаря своей способности обрабатывать изображение за один просмотр, в результате чего достигается высокая скорость обработки при сохранении достаточной точности. Алгоритм разделяет входное изображение на множество ячеек сетки и для каждой ячейки параллельно предсказывает ограничивающие рамки и вероятности классов объектов. В отличие от традиционных методов, YOLO

способен определять множество объектов различных классов на изображении, что делает его выдающимся инструментом в области машинного обучения и анализа изображений.

1. Общие сведения

Данный документ содержит описание технической архитектуры программного обеспечения «Система поиска GigaFlex КОЗ 1 Автономный поиск» (далее – ПО). Все исключительные права на ПО принадлежат Фонду НТИ (далее – Компания).

Настоящий документ подлежит размещению на официальном сайте Компании в сети Интернет по адресу: https://nti.fund/about/activity/information.php (далее – официальный сайт).

ПО не имеет пользовательского графического интерфейса, управление осуществляется с использованием консоли.

ПО разработано с целью автоматизации процесса поиска людей в условиях природной местности.

ПО способно автоматически идентифицировать человеческие фигуры на изображениях, полученных после полета беспилотных воздушных судов (БВС), используя технологии искусственного интеллекта.

Эффективность распознавания оптимизирована для работы на высотах от 45 до 70 метров. Максимальная эффективность достигается при высоте полета близкой к 45 метрам.

2. Описание архитектуры приложения

2.1. Описание функций

В данном программном обеспечении не используются классы. Ниже приведены используемые функции:

read_image: функция возвращает numpy массив и выполняет чтение изображения из файла с последующим преобразованием цветового пространства. Параметр функции image_path представляет собой путь к файлу изображения в формате Path. Функция возвращает изображение в формате RGB как numpy массив и для выполнения использует библиотеку OpenCV (cv2).

infer_image_bbox: функция возвращает List[dict] и выполняет детектирование объектов на изображении с последующим извлечением их ограничивающих рамок. Параметр функции image является numpy массивом, представляющим входное изображение. Функция возвращает список словарей, где каждый словарь содержит нормализованные координаты центра (xc, yc), размеры (w, h), метку класса (label) и уверенность детекции (score) для каждого обнаруженного объекта. Для выполнения использует предварительно загруженную модель YOLO, библиотеки OpenCV (cv2) и numpy, а также учитывает глобальные параметры конфигурации BOX_RATIO, IOU и CONF при обработке результатов.

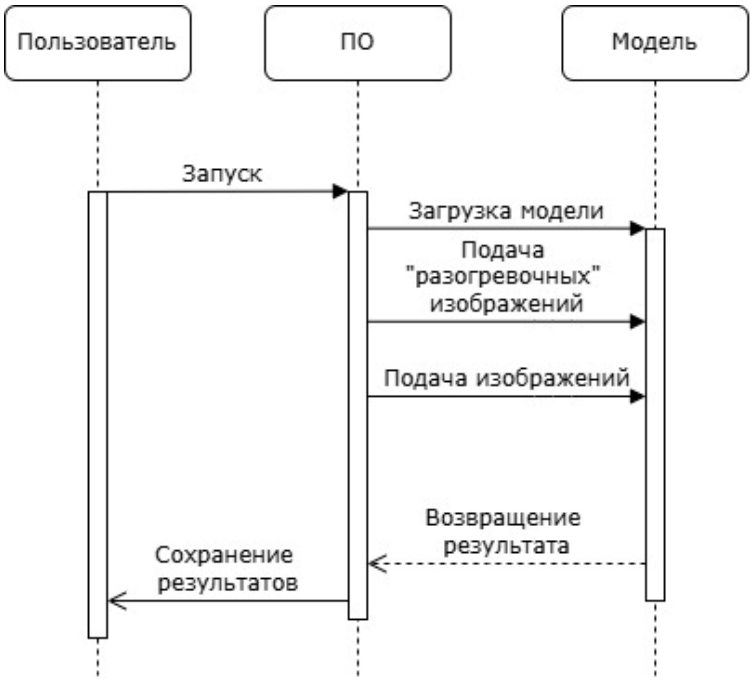
predict: функция возвращает List[List[dict]] и выполняет пакетное предсказание для одного или нескольких изображений. Параметр функции images может быть как отдельным изображением (np.ndarray), так и списком изображений (List[np.ndarray]). Функция возвращает список списков словарей, где каждый внутренний список содержит результаты детекции для соответствующего изображения в формате словарей с координатами, размерами, меткой класса и оценкой уверенности. Для выполнения использует вспомогательную функцию infer_image_bbox для обработки отдельных изображений.

create_solution: функция не возвращает значение и отвечает за создание итогового CSV-файла с результатами обработки всех изображений. Функция не принимает параметров, но использует глобальную константу SAVE_PATH для указания пути сохранения результатов. Функция обрабатывает все JPG-изображения в директории «../test_images/images», включая подпапки, создавая для каждого изображения запись в CSV с информацией о детекции объектов или пустую запись, если объекты не обнаружены. При обработке каждого изображения измеряется время выполнения, сохраняются размеры изображения и результаты детекции. Для выполнения использует вспомогательные функции read_image и predict.

main: функция выступает в качестве входной точки программы и не возвращает значение. Функция не принимает параметров и отвечает за запуск процесса обработки изображений путем вызова функции create_solution.

2.2. Диаграмма последовательностей

Диаграмма последовательности программного обеспечения «Система поиска GigaFlex КОЗ 1 Автономный поиск» приведена на рисунке:



Пользователь запускает программное обеспечение, предварительно проверив, что по указанным выше путям находятся фотографии, полученные с БВС. Происходит инициализация модели и загрузка «разогревочных»

изображений. Далее модель обрабатывает несколько таких изображений, чтобы в последствии уменьшить дисперсию времени обработки. После программное обеспечение принимает целевые фотографии в качестве входных данных, загружает из постоянной памяти в оперативную память. Модель обрабатывает изображения последовательно. Результаты обработки собираются в список, где они фильтруются и конвертируются в специфический формат.

Обработанные данные сохраняются в CSV-файл, который включает в себя следующие поля данных для каждого обработанного изображения:

«image_id»: Название обработанного файла изображения.

«xc»: Относительная координата X центра обнаруженного объекта.

«yc»: Относительная координата Y центра обнаруженного объекта.

«w»: Относительная ширина области обнаружения объекта.

«h»: Относительная высота области обнаружения объекта.

«label»: Метка класса обнаруженного объекта (в данном случае единственный класс).

«score»: Вероятность достоверности обнаружения объекта.

«time_spent»: Время, затраченное на обработку изображения.

«w_img»: Ширина исходного изображения.

«h_img»: Высота исходного изображения.

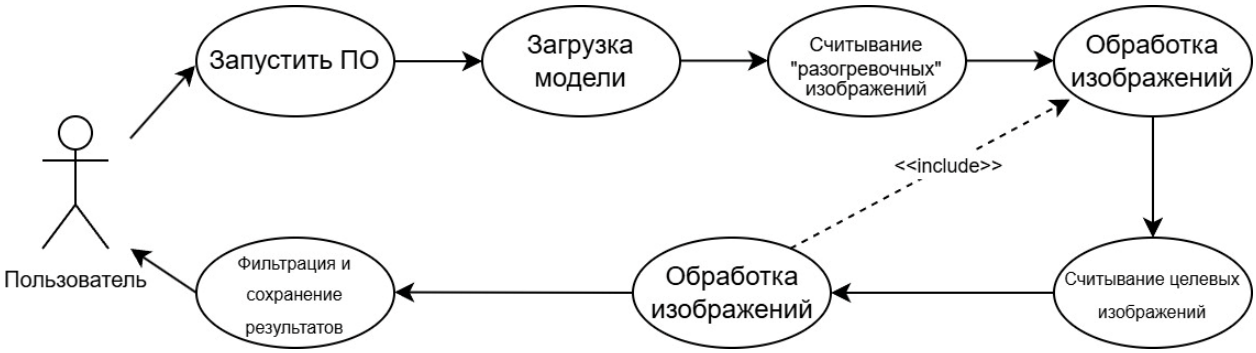
Пример записи в CSV-файле:

image_id,xc,yc,w,h,label,score,time_spent,w_img,h_img

EXAMPLE.JPG,0.12345, 0.12345, 0.2253, 0.3123,0 ,0.8345,0.156,640,480

2.3. Диаграмма прецедентов

Диаграмма прецедентов программного обеспечения «Система поиска GigaFlex КОЗ 1 Автономный поиск» приведена на рисунке:



Пользователь начинает работу с программой, предварительно проверив, что по указанным выше путям находятся фотографии. Программа автоматически обрабатывает фотографии с диска в оперативную память, подавая их в модель. Обработанные данные сохраняются в CSV-файл, который включает в себя следующие поля данных для каждого обработанного изображения:

«image_id»: Название обработанного файла изображения.

«xc»: Относительная координата X центра обнаруженного объекта.

«yc»: Относительная координата Y центра обнаруженного объекта.

«w»: Относительная ширина области обнаружения объекта.

«h»: Относительная высота области обнаружения объекта.

«label»: Метка класса обнаруженного объекта (в данном случае единственный класс).

«score»: Вероятность достоверности обнаружения объекта.

«time_spent»: Время, затраченное на обработку изображения.

«w_img»: Ширина исходного изображения.

«h_img»: Высота исходного изображения.

Пример записи в CSV-файле:

image_id,xc,yc,w,h,label,score,time_spent,w_img,h_img

EXAMPLE.JPG,0.12345, 0.12345, 0.2253, 0.3123,0 ,0.8345,0.156,640,480

3. Описание архитектурного стиля

В разработке программы был использован стиль кодирования, основанный на PEP 8, что является общепринятым стандартом написания кода на Python. Это включает в себя соблюдение отступов в четыре пробела, ограничение максимальной длины строки 79 символами, а также использование понятных имен переменных и функций, описывающих их назначение и функциональность. Комментарии и строковые документации (docstrings) использовались в некоторых функциях для облегчения понимания кода и его последующей модификации или расширения.

В отношении паттернов проектирования, основной упор сделан на функциональное программирование, учитывая, что основная часть кода организована с использованием функций. Это способствует модулярности и переиспользуемости кода, так как функции могут быть легко заменены или модифицированы без воздействия на остальные части программы.

Также стоит отметить, что некоторые функции могут быть реализованы как чистые функции, что упрощает их тестирование и отладку, а также повышает надежность программы за счет уменьшения побочных эффектов.

Для связи с разработчиками писать на почту ntifundsoft@nti.fund