Geekbrains

**Интеграция YOLOv5 и YOLOv8, машинного зрения для обнаружения и отслеживания объектов на видео**

Разработчик. Программист Python

Алексеев Павел Юрьевич

2024

1.Введение

1. Нейронные сети – общая информация
2. Машинное зрение – общая информация
3. Описание проблемы
4. Плюсы и минусы решения.
   1. Системы подсчета трафика с помощью Wi-Fi радаров:

5.2 Системы подсчета клиентов на кассах магазинов:

1. Выбор модели
   1. YOLOv8 и YOLOv5
   2. Тестирование системы, результаты
2. Определение конечного результата, определения контрольных параметров успешной реализации.
3. Оборудование и ресурсы для коммерческой реализации проекта

8.1 Код и настройка

8.2 Тест и результаты

1. Заключение
2. Список литературы

**Введение**

Компьютерное зрение – это направление искусственного интеллекта, которое обучает компьютеры анализировать и интерпретировать визуальную информацию. Путем использования цифровых изображений с камер и видео, а также моделей глубокого обучения, машины способны точно распознавать и классифицировать объекты, после чего реагировать на то, что они "видят". Компьютерное зрение находит широкое применение в различных областях, включая системы видеонаблюдения, управление автомобилями, медицинские системы анализа изображений, а также в технологиях дополненной и виртуальной реальности. Машинное зрение стремится имитировать механизм восприятия и обработки визуальной информации человеческим мозгом.

В современном мире, для получения статистических данных, необходимых для более гибкого предложения клиентам, используются различные системы. Однако информация о пешеходном трафике до сих пор собирается устаревшими методами, такими как ручные опросы, фрагментарная статистика или даже субъективные оценки. Применение мощных инструментов, таких как нейронные сети и машинное зрение, либо минимально используется, либо вовсе игнорируется. Внедрение новых технологий может решить данную проблему.

Интеграция передовых систем машинного зрения, таких как YOLOv5 и YOLOv8, для обнаружения и отслеживания объектов на видео, становится ключевым шагом в современном бизнесе. Однако, одной из основных проблем, с которой сталкиваются разработчики и предприниматели, является сложность настройки и интеграции этих систем в существующие процессы. Этот процесс требует глубоких знаний в области компьютерного зрения и искусственного интеллекта, а также значительных ресурсов как по времени, так и по квалификации.

Цель данной работы заключается в повышении эффективности выбора коммерческих помещений для всех участников бизнес-процесса до 90% путем получения качественной статистики в полуавтоматическом или полностью автоматическом режиме. Использование систем машинного зрения для анализа пешеходного трафика может значительно улучшить процессы принятия решений и помочь бизнесу стать более эффективным и конкурентоспособным. Для достижения поставленной цели необходимо выполнить следующие этапы:

1. Сформировать концепцию проекта на основе потребностей участников бизнес-процесса и определить конечные цели.
2. Проанализировать существующие решения для реализации проекта.
3. Определить конечный вид продукта (проекта) и контрольные параметры.
4. Определить необходимые ресурсы для реализации проекта.
5. Создать сеть машинного зрения.
6. Провести тестирование и сравнить результаты с контрольными параметрами.

Объектом исследования данной работы являются нейронные сети, принципы их работы, механизм функционирования систем машинного зрения, а также возможность создания собственной модели машинного зрения или использование готовой обученной модели.

**Нейронные сети – общая информация**

Изучение и применение искусственных нейронных сетей началось в середине 20 века, однако они стали широко известными лишь позднее благодаря появлению продвинутых вычислительных устройств, способных эффективно работать с нейронными сетями. В настоящее время можно легко создать нейронную сеть средней сложности на персональном компьютере.

Нейронная сеть представляет собой систему взаимосвязанных нейронов, каждый из которых вычисляет выходной сигнал на основе входных сигналов. Основные этапы работы одного нейрона включают:

1. прием сигналов от предыдущих элементов сети;
2. комбинирование входных сигналов;
3. вычисление выходного сигнала;
4. передача выходного сигнала следующим элементам нейронной сети.

Нейроны могут быть соединены между собой различными способами в зависимости от структуры конкретной сети, однако основной принцип работы нейронной сети остается неизменным. В результате комбинации входных сигналов формируется выходной сигнал или несколько выходных сигналов.

Упрощенно нейронную сеть можно представить, как черный ящик с входами и выходами, внутри которого находится множество нейронов.

Далее будут рассмотрены основные этапы работы сети, начиная с комбинирования входных сигналов.

Взвешенные связи в нейронной сети описываются тремя основными факторами:

1. Элементы связи: каждая связь имеет отправной и конечный элемент, то есть нейроны в сети, между которыми она устанавливается.
2. Вес связи: вес связи определяет важность и воздействие сигнала, передаваемого по данной связи. Этот параметр может усиливать или ослаблять входной сигнал. Например, если выходной сигнал нейрона 1 равен 5, а вес связи между этим нейроном и нейроном 2 равен 2, то входной сигнал нейрона 2 будет равен 5 \* 2 = 10.
3. Суммирование: если к нейрону поступает несколько входных сигналов через различные связи, их взвешенные значения суммируются. Это позволяет объединить информацию от различных источников и сформировать общий входной сигнал для нейрона.

Структура связей между нейронами часто представляется в виде матрицы весов W, где элемент матрицы 𝑤𝑖𝑗 определяет вес связи от элемента i к элементу j. Пользуясь весовой матрицей, можно эффективно описывать взаимодействие между нейронами и осуществлять пересчет входных сигналов в нейронной сети.

Взвешенные связи играют ключевую роль в передаче информации и расчетах внутри нейронной сети, позволяя эффективно обрабатывать данные и принимать решения на основе входных сигналов.

Фактор в описании взвешенных связей - это вес связи, который определяет важность и воздействие сигнала, передаваемого по данной связи. Этот параметр может усиливать или ослаблять входной сигнал. Каждая связь в сети имеет свой вес, определяющий, насколько сильно или слабо сигнал будет воздействовать на следующий элемент нейронной сети.

Пример: Если выходной сигнал нейрона 1 равен 5, а вес связи между этим нейроном и нейроном 2 равен 2, то входной сигнал нейрона 2 будет равен 5 \* 2 = 10.

Таким образом, вес связи является одним из ключевых параметров при моделировании и функционировании нейронных сетей, так как он определяет, насколько значимым будет воздействие каждого конкретного входного сигнала на работу нейронной сети.

Пример весовой матрицы для данной нейронной сети:

Перейдем к функции активации элемента. Для каждого элемента сети существует определенное правило, в соответствии с которым из значения комбинированного ввода элемента вычисляется его выходное значение. Это правило называется функцией активации, а само выходное значение называется активностью нейрона.

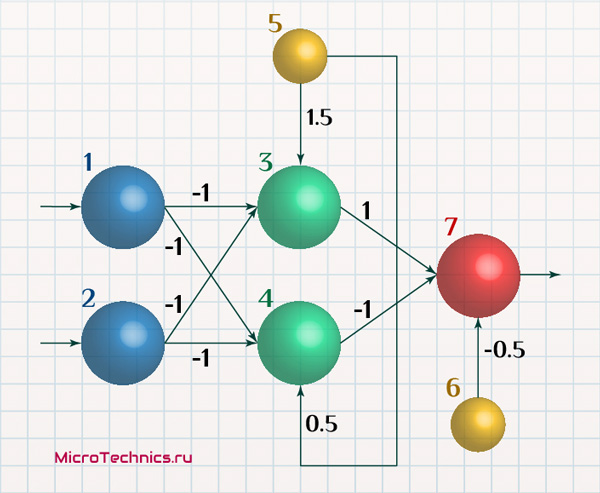
Функции активации могут представлять собой различные математические уравнения. В качестве примера рассмотрим две из самых популярных и часто используемых функций:

1. Пороговая функция: если значение комбинированного ввода ниже определенного значения (порога), то активность равна нулю, если выше – единице.
2. Логистическая функция: эта функция часто используется в нейронных сетях. Она преобразует взвешенную сумму в диапазоне от 0 до 1, что позволяет интерпретировать результат как вероятность.

Функции активации играют важную роль в определении поведения нейронов в нейронной сети, определяя, какой тип выхода будет сгенерирован в ответ на комбинированный вход.

Рассмотрим еще один небольшой пример, который очень часто используется в литературе для объяснения сути работы нейронных сетей.

Задача примера заключается в том, чтобы при помощи нейронной сети вычислить отношение XOR. То есть на вход мы будем подавать разные варианты сигналов, а на выходе должны получить результат операции XOR для поданных на вход значений:



Элементы 1 и 2 являются входными, а элемент 7 – выходным. Нейроны 5 и 6 называются скрытыми, поскольку они не связаны с внешней средой. Таким образом, мы получили три слоя – входной, скрытый и выходной. Элементы 3 и 4 называют элементами смещения. Их выходной сигнал (активность) всегда равен 1. Для вычисления комбинированного ввода в этой сети мы будем использовать правило суммирования взвешенных связей, а в качестве функции активности будет выступать пороговая функция. Если комбинированный ввод элемента меньше 0, то активность равна 0, если ввод больше 0, то активность – 1.

Подадим на вход нейрона 1 – единицу, а на вход нейрона 2 – ноль. В этом случае на выходе мы должны получить 1 (0 XOR 1 = 1). Рассчитаем выходное значение вручную для демонстрации работы сети.

1. Комбинированный ввод элемента 5:

* net5 = 1 \* (-1) + 0 \* (-1) + 1 \* 1.5 = 0.5.
* Активность элемента 5: 1 (0.5 > 0).

1. Комбинированный ввод элемента 6:

* net6 = 1 \* (-1) + 0 \* (-1) + 1 \* 0.5 = -0.5.
* Активность элемента 6: 0.

Дополнительно к примеру, с операцией XOR, стоит отметить ключевую роль скрытого слоя в нейронной сети. Скрытый слой, представленный нейронами 5 и 6, позволяет моделировать нелинейные зависимости между входными и выходными данными. Благодаря наличию скрытого слоя нейронная сеть может эффективно обучаться и решать более сложные задачи, которые линейные модели не способны адекватно обработать.

Также стоит отметить использование элементов смещения (нейроны 3 и 4) в данной сети. Эти элементы добавляют дополнительную гибкость и возможность смещения функции активации, что помогает нейронной сети лучше адаптироваться к различным входным данным.

Важно понимать, что применение подходящей архитектуры нейронной сети, выбор функций активации и весов позволяют достичь хороших результатов при решении различных задач машинного обучения. Пример с XOR иллюстрирует, как нейронные сети могут использоваться для моделирования сложных логических операций, а также важность корректного подбора параметров сети для успешного выполнения задач.

Комбинированный ввод элемента 7: net7 = 1 \* (1) + 0 \* (-1) + 1 \* (-0.5) =

0.5. Активность элемента 7, а в то же время и выходное значение сети равно 1. Что и требовалось доказать.

Можно попробовать использовать в качестве входных сигналов все возможные значения (0 и 0, 1 и 0, 0 и 1, 1 и 1), на выходе мы всегда будем видеть значение, соответствующее таблице истинности операции XOR.

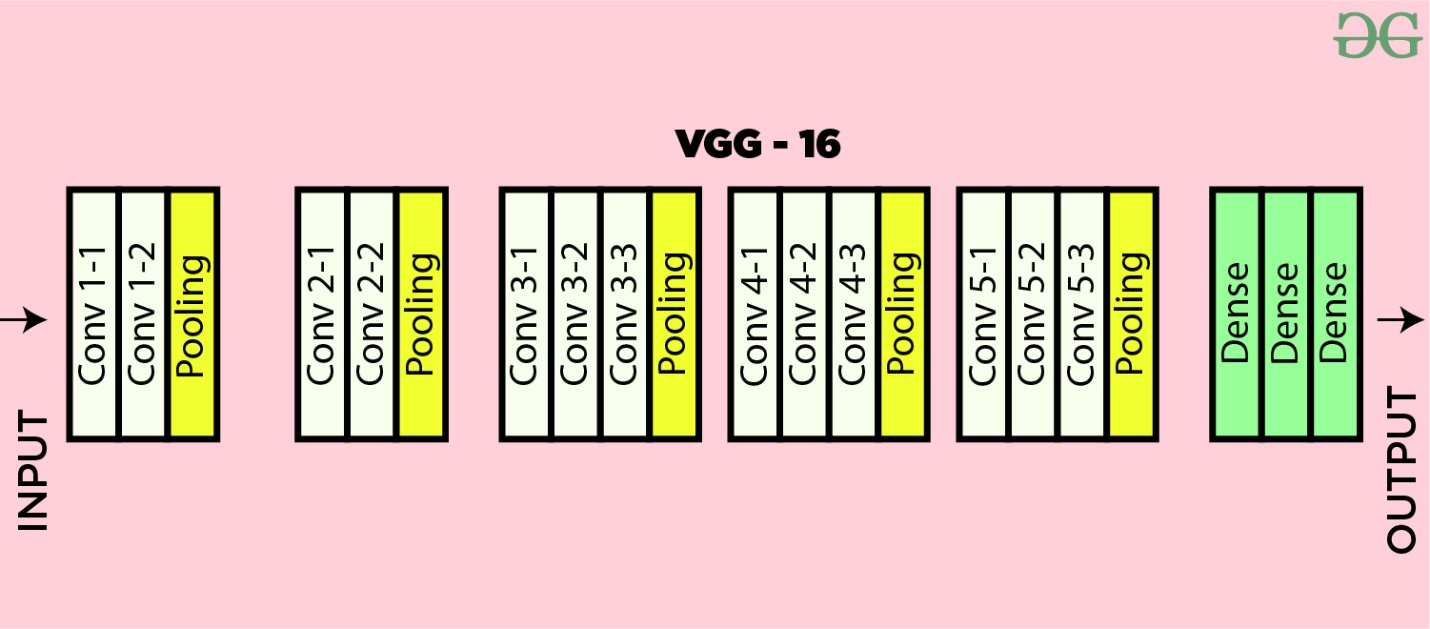
В данном случае все значения весовых коэффициентов нам были известны заранее, но главной особенностью нейронных сетей является то, что они могут сами корректировать значения веса всех связей в процессе обучения сети.

**Машинное зрение – общая информация.**

Машинное зрение — это научное направление в области искусственного интеллекта, в частности робототехники, и связанные с ним технологии получения изображений объектов реального мира, их обработки и использования, полученных данных для решения разного рода прикладных задач без участия (полного или частичного) человека.

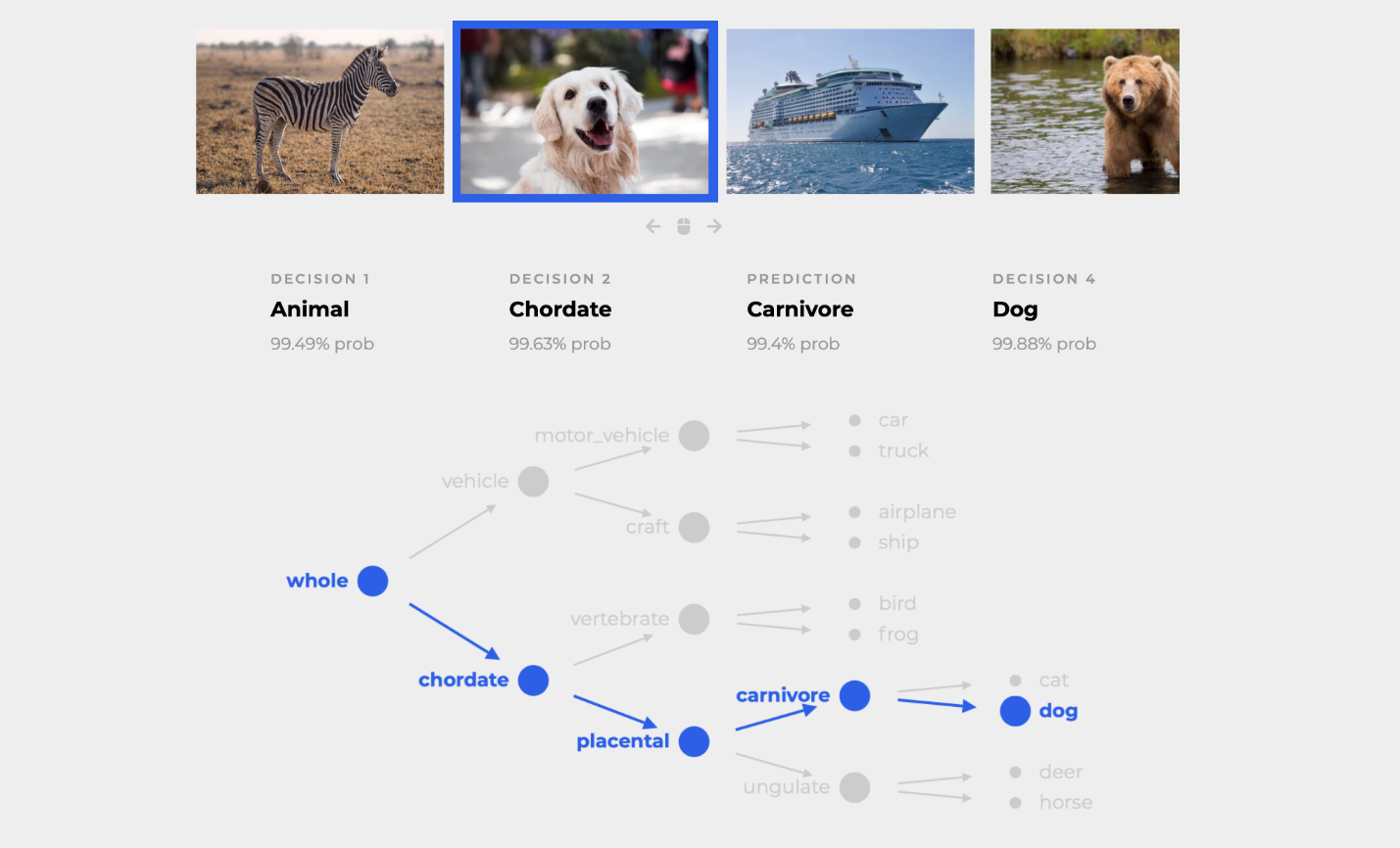
Сегодня существует большое количество моделей для работы с компьютерным зрением. Рассмотрим, каждую модель от первых сверточных моделей до современных. AlexNet – первая глубокая сверточная нейронная сеть.

VGG16 – модель сверточной нейронной сети, разработанная К. Симоняном и А. Зиссерманом из Оксфордского университета, представляет собой мощный инструмент для обработки изображений. Эта модель способна работать с трехканальными изображениями разрешением 224х224 и содержит впечатляющее количество параметров – 62,3 миллиона.



Одной из особенностей VGG16 является высокая скорость обучения благодаря применению функции активации ReLU и относительной "легкости" самой модели. VGG16 демонстрирует высокую точность в задаче распознавания объектов на изображениях, достигая показателя в 92.7% top-5 на датасете ImageNet, содержащем более 14 миллионов изображений, относящихся к 1000 классам.

SEER представляет собой самообучающуюся нейросеть с миллиардом параметров, созданную FAIR для решения задач компьютерного зрения. Эта модель обучена на изображениях из социальной сети Instagram и может быть дообучена под конкретные задачи пользователем. SEER характеризуется наличием ReLU во всех скрытых слоях, что обеспечивает определенные преимущества при обработке изображений. Однако, из-за глубокой архитектуры и большого количества полносвязных узлов, модель имеет медленную скорость обучения и высокий вес, что может повлиять на производительность при работе с данными.



NBDT (Neural-Backed Decision Trees) представляет собой нейросетевую архитектуру деревьев решений, разработанную для задач классификации. Эта модель объединяет в себе интерпретируемость классического алгоритма дерева решений с высоким качеством предсказаний современных нейронных сетей. Одной из особенностей NBDT является способность выдавать последовательные решения, которые приводят к точному предсказанию.

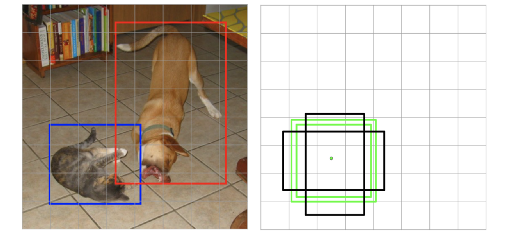
С другой стороны, YOLO (You Only Look Once) представляет собой семейство моделей, которые завоевали популярность благодаря своей легковесности и качеству предсказаний.



YOLO отличается тем, что способен делать предсказания объектов в реальном времени, выполняя одновременно обнаружение и классификацию объектов на изображении. Это позволяет быстро и эффективно обрабатывать видео и изображения с большим количеством объектов.

Каждая из упомянутых моделей обладает своими особенностями и применяется в различных задачах компьютерного зрения в зависимости от требований по скорости работы, точности предсказаний и доступным ресурсам.

SSD (Single Shot MultiBox Detector)



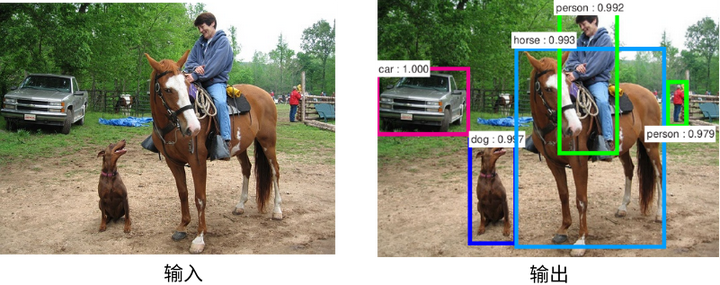
— это сеть, которая объединяет наиболее удачные черты архитектуры YOLO с собственными особенностями для повышения производительности. Одной из отличительных особенностей SSD является применение нейронной сети ко всему изображению, разделяя его на сетку и обнаруживая объекты в каждом из созданных "окон" (якорная система). Модель версии 4 достигает точности 55,8% при работе на портативных устройствах и в реальном времени.

Другой моделью, которая также позиционируется для работы в реальном времени и имеет высокую скорость работы, является SqueezeNet. Эта архитектура сверточной нейронной сети использует модуль Fire, состоящий из слоя сжатия и расширения, для уменьшения количества параметров и увеличения скорости работы нейронной сети.

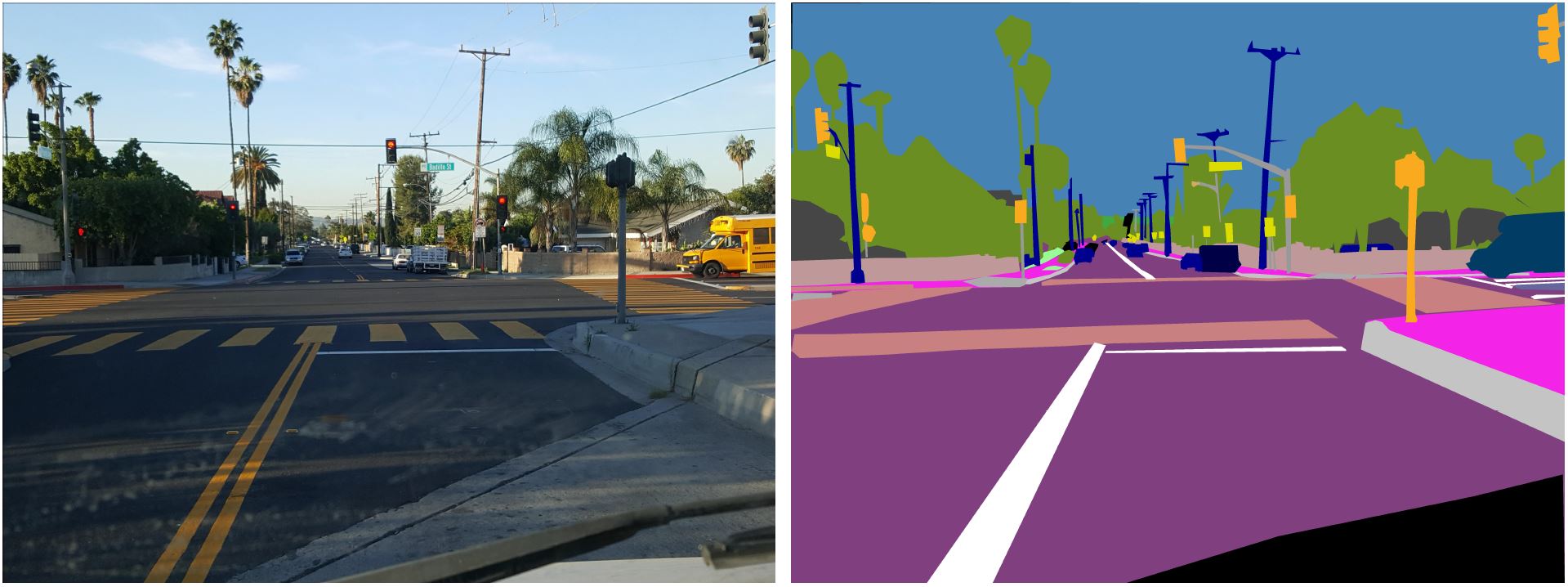
Кроме того, SSD и YOLO обладают возможностью определять, как крупные, так и мелкие объекты на изображениях за один цикл работы, что делает их эффективными инструментами для обнаружения объектов в реальном времени.

DeepLab (Semantic Image Segmentation with Deep Convolutional Nets) – модель, предназначенная для сегментации объектов на изображениях. Одной из особенностей DeepLab является наличие разряженной свертки (dilated convolution), которая позволяет сохранять пространственное разрешение и улучшать качество сегментированных объектов. Эта модель также способна убирать шумы в изображении, что положительно влияет на точность сегментации.

Faster R-CNN

 – это сеть, разработанная на основе R-CNN и следующей версии Fast R-CNN. В Faster R-CNN для локализации объектов используются Region Proposal Networks, что позволяет генерировать предложенные регионы для расположения объектов на основе последней сверточной карты признаков. Одной из отличительных особенностей Faster R-CNN является скорость работы по сравнению с Fast R-CNN, хотя локализация объектов может быть менее качественной. Также существуют предобученные модели Faster R-CNN, что делает их доступными для различных задач.

ResNet (Residual Network)

– это модель глубокой нейронной сети, разработанная Microsoft для классификации изображений. ResNet была создана для решения проблемы снижения точности предсказаний при увеличении количества слоев в нейронных сетях. Одной из ключевых особенностей ResNet является использование остаточных блоков, что позволяет эффективно обучать глубокие нейронные сети и избежать проблемы затухания градиентов.

EfficientNets  представляют собой класс моделей, разработанных с целью сохранения высокого качества решения задач и увеличения эффективности модели путем уменьшения количества параметров и снижения вычислительной сложности. Одной из особенностей EfficientNets является возможность самостоятельного конструирования сети, включая выбор глубины, разрешения и ширины, что позволяет подбирать оптимальные параметры для достижения высокой точности и эффективности. Существуют несколько версий моделей EfficientNets: базовая версия без преднастроенных параметров и несколько версий с заранее определенными весами для параметров, облегчающими поиск оптимальных настроек.

U-Net является стандартной архитектурой сверточных нейронных сетей, применяемой для задач сегментации изображений. Эта модель используется для создания маски, которая разделяет изображение на несколько классов, что позволяет определять класс изображения целиком и сегментировать его области по классам. U-Net широко применяется в медицинских областях и других сферах, где требуется точная сегментация объектов на изображениях.

 Используя глубокие сверточные нейронные сети, модель U-Net обеспечивает быструю и точную сегментацию изображений разрешением 512x512 пикселей на современных GPU. Эта модель демонстрирует высокий уровень успешности в различных областях, включая биомедицинское применение, и способна достигать высоких результатов даже при ограниченном объеме данных для обучения.

**Описание проблемы.**

В современном бизнесе, особенно в области технологий, существует потребность в интеграции передовых систем машинного зрения, таких как YOLOv5 и YOLOv8, для обнаружения и отслеживания объектов на видео. Однако, одной из основных проблем, с которой сталкиваются разработчики и предприниматели, является сложность настройки и интеграции этих систем в существующие процессы.

Интеграция YOLOv5 и YOLOv8 требует глубоких знаний в области компьютерного зрения и искусственного интеллекта. Необходимо правильно настроить модели, обучить их на соответствующих наборах данных, а затем интегрировать их в существующее программное обеспечение или инфраструктуру. Этот процесс может быть сложным и требует значительных ресурсов как по времени, так и по знаниям.

Более того, эффективное использование систем машинного зрения для обнаружения и отслеживания объектов на видео требует постоянной оптимизации и обновления моделей, чтобы обеспечить высокую точность и надежность распознавания. Это вызывает необходимость в постоянном мониторинге и поддержке системы, что также может быть вызовом для бизнеса.

Таким образом, интеграция YOLOv5 и YOLOv8 в бизнес-процессы требует не только технических знаний, но и постоянного внимания к обновлениям и оптимизации моделей для обеспечения эффективной работы системы машинного зрения.

**Плюсы и минусы решения.**

1. Статистика из поисковых сервисов:

* Плюсы: Предоставляет релевантные результаты на основе запросов пользователей.
* Минусы: При использовании геолокации может быть получена не всегда релевантная статистика, так как запросы могут быть сделаны для других целей или пользователями из других мест.

1. Статистика из картографических сервисов:

* Плюсы: Интегрирован в окружение пользователя и собирает больше данных, что может улучшить релевантность информации.
* Минусы: Подвержен аналогичным проблемам со связанными с релевантностью данных; информация о клиентах может быть запрошена навязчивым образом, что может негативно повлиять на пользователя.

1. Сбор статистики из пешеходных картографических сервисов:

* Плюсы: Может предложить более широкий массив данных для анализа и сбора статистики, включая информацию о перемещении людей.
* Минусы: Требует значительных вычислительных мощностей для сбора и анализа данных; доступ к этой информации может быть ограничен из-за сложностей в сборе и анализе данных; существует потенциальная угроза приватности и этических проблем, таких как тотальная слежка за пользователями, что может быть запрещено законодательством или привести к негативным реакциям пользователей.

Каждый из перечисленных подходов имеет свои преимущества и ограничения, и для выбора оптимального решения необходимо учитывать контекст и требования конкретной ситуации.

**Системы подсчета трафика с помощью Wi-Fi радаров:**

1. Плюсы:

* Автономность работы: Радар может функционировать автономно.
* Малое потребление ресурсов и мощности: Требуется незначительное количество ресурсов для работы системы.
* Дистанционный контроль: Возможность управления системой удаленно.

1. Минусы:

* Малый радиус действия: Система ограничена в радиусе работы.
* Неучет клиентов без современных гаджетов или выключенного Wi-Fi: Может пропустить клиентов без подключенных устройств или в случае отключенного Wi-Fi.
* Необходимость специального оборудования и настройки: Требует специализированного оборудования и его конфигурации.
* Дублирование данных: Проблема дублирования данных при наличии нескольких устройств у одного клиента (например, планшетов и смартфонов).
* Возможные помехи для других Wi-Fi сетей: Может создавать интерференцию с другими Wi-Fi сетями.
* Проблемы с законодательством и этикой: Возможные юридические и этические проблемы в связи с сбором и хранением данных пользователей.

Каждый метод имеет свои плюсы и минусы, и выбор оптимальной системы зависит от конкретных целей, требований и ограничений бизнеса.

**Системы подсчета клиентов на кассах магазинов:**

1. Плюсы:

* Нивелирует этические и юридические проблемы: Поскольку камеры расположены внутри коммерческих помещений, это ограничивает возможные проблемы с точки зрения законодательства и этики, так как магазины могут контролировать свою частную собственность.
* Основано на подсчете людей, а не устройств: В отличие от системы, основанной на Wi-Fi, данная технология позволяет точнее определять фактическое количество посетителей, а не только устройств с активированным Wi-Fi.
* Не требует специального оборудования: Большинство магазинов уже имеют видеокамеры или камеры безопасности, что делает эту систему более доступной для внедрения.

1. Минусы:

* Требование ресурсов для хранения видеоматериалов: Круглосуточная запись видеоматериалов с нескольких источников требует значительных объемов хранилища данных.
* Постоянное дообучение системы искусственного интеллекта: Различное расположение камер и ракурсы требуют постоянного корректирования и дообучения системы для точного подсчета клиентов.
* Высокие вычислительные требования: Обработка видеозаписей, поступающих с нескольких источников, требует значительных вычислительных мощностей из-за их большого объема.

Эти факторы следует учитывать при выборе оптимальной системы для конкретных целей и требований бизнеса.

Менее эффективные способы подсчета клиентов:

1. Датчики движения или пересечения определенной точки:

* Плюсы: Этот метод обладает стабильной работой и низкой стоимостью установки и обслуживания. Также, оборудование для этого метода широко доступно.
* Минусы: Датчики движения имеют узкую сферу применения и не могут быть использованы в общественных местах, что ограничивает их эффективность для подсчета клиентов в магазинах.

1. Построение модели прогнозирования на основе данных о местоположении и близлежащих местах посредством искусственного интеллекта:

* Плюсы: Этот метод не требует установки внешних датчиков, так как использует уже имеющиеся данные.
* Минусы: Однако он имеет неочевидные недостатки, такие как неточность модели, необходимость корректировки, а также высокая стоимость эксплуатации и конечной стоимости для потребителя.

1. Сбор информации от сотовых операторов:

* Плюсы: Этот метод обладает практически полным покрытием информацией о клиентах, большим количеством данных и возможностью создания оптимальной модели.
* Минусы: Однако, высокая стоимость доступа к данным из-за их собственности операторов, приблизительность данных и возможные искажения от других устройств или трафика делают его менее привлекательным в контексте стоимости и точности.

Эти методы могут быть применимы в зависимости от конкретной задачи и специфики запроса клиента, однако они имеют свои ограничения, которые необходимо учитывать при выборе оптимального решения.

Предложенная идея "Системы машинного зрения для отслеживания пешеходного трафика" с установкой камер в заранее определенных точках и выбором оптимальной точки для подсчета пешеходного трафика имеет потенциал для нивелирования недостатков предыдущих методов. Вот какие преимущества и особенности можно выделить:

1. Стандартизированные условия съемки:

* Установка камер в заранее определенных точках, обеспечивающая одинаковый угол съемки и освещение, может значительно повысить точность системы за счет анализа записей, сделанных при идентичных условиях.
* Это также позволит увеличить тайминги дообучения системы и обеспечить более стабильную работу.

1. Выбор оптимальной точки для подсчета трафика:

* Определение точки для подсчета пешеходного трафика на основе изучения наиболее релевантных мест, таких как остановки общественного транспорта, может улучшить точность подсчета и соответствие получаемых данных реальной ситуации.

1. Идентичные характеристики видеозаписей:

* Выбор оборудования с одинаковыми характеристиками видеозаписей, такими как разрешение, формат, кодек и соотношение сторон, поможет снизить нагрузку на оборудование для обработки данных и улучшить эффективность модели благодаря использованию единого формата записей.

Эти отличительные особенности могут действительно помочь повысить точность и эффективность системы для отслеживания пешеходного трафика, обеспечивая лучшие результаты по сравнению с предыдущими методами.

**Выбор модели.**

В контексте поставленной задачи, где требуется отслеживание пешеходного трафика с использованием системы машинного зрения, предложенный подход к выбору модели вполне разумен. Учитывая ограничения по времени и ресурсам, а также наличие уже обученных моделей, создание собственной модели с нуля может быть неоправданно.

Вместо этого, целесообразно использовать уже обученные модели машинного зрения, которые можно адаптировать к конкретным потребностям задачи. Это позволит сократить время разработки, опробования и оптимизации модели, а также уменьшить нагрузку на вычислительные ресурсы и время обучения за счет использования существующих датасетов.

Такой подход также обеспечит доступ к широкому спектру уже обученных моделей, которые могут быть адаптированы для работы в различных условиях и с различным оборудованием. Это позволит избежать необходимости разработки модели "с нуля" и сосредоточить усилия на финетюнах и адаптации выбранной модели под конкретные требования проекта.

Таким образом, использование уже обученных моделей для выполнения поставленной задачи представляется более эффективным и практичным в данном контексте.

Исходя из ограничений по ресурсам и времени, а также наличия частично реализованного функционала другими разработчиками, использование предобученных моделей машинного зрения для решения задачи отслеживания пешеходного трафика кажется целесообразным подходом.

Рассмотренные модели машинного зрения, такие как AlexNet и VGG16, хоть и имеют свои преимущества, но не оптимальны для данной задачи из-за ориентированности на статичные изображения, устаревшей архитектуры и тяжеловесности, что может привести к медленной работе и недостаточной эффективности при обработке видеопотока с отслеживанием пешеходов.

Более современные модели, такие как Overfeat, RCNN, RPN и их модификации, которые используют двухфазные методы обнаружения объектов, представляются более перспективными. Эти модели способны находить "подозрительные" зоны на изображениях и затем классифицировать их, что повышает точность обнаружения объектов. Однако, следует учитывать, что данные модели могут обладать относительно невысокой скоростью работы, что также является важным фактором при выборе оптимальной модели для решения задачи отслеживания пешеходного трафика.

Таким образом, выбор современных моделей машинного зрения, способных обнаруживать объекты с высокой точностью, при условии оптимизации и ускорения их работы, может быть ключевым для успешной реализации проекта по отслеживанию пешеходного трафика в рамках имеющихся ограничений.

Исходя из упомянутых особенностей и результатов сравнения моделей Faster R-CNN, SSD и YOLOv3 для решения задачи отслеживания пешеходного трафика в реальном времени при ограниченных ресурсах и времени, оптимальным выбором будет использование модели SSD (Single Shot MultiBox Detector).

Хотя модель Faster R-CNN продемонстрировала высокий процент mAP (87,7%), ее скорость обработки данных оказалась недостаточной для анализа видеопотока в реальном времени, что делает ее менее подходящей для данного случая.

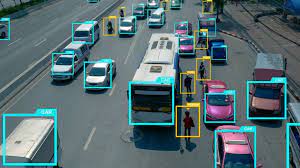
С другой стороны, модель SSD показала хороший баланс между точностью (mAP 82,4%) и скоростью обработки данных, увеличив скорость на 360% по сравнению с Faster R-CNN. Это позволит эффективно обрабатывать видеопоток в реальном времени при достаточно высоком уровне точности.

Модель YOLOv3, хотя обладает высокой скоростью обработки данных (рост на 63% по сравнению с SSD), ее точность была наихудшей среди рассмотренных моделей (mAP 80,2%), что может отразиться на качестве обнаружения объектов в условиях отслеживания пешеходного трафика.

Таким образом, модель SSD представляется наилучшим выбором для решения поставленной задачи, обеспечивая достаточную точность детектирования объектов при приемлемой скорости работы, что является критическим фактором для анализа видеопотока в реальном времени.

При рассмотрении возможности расширения проекта на автоматический сбор и анализ данных в режиме реального времени, выбор модели YOLOv5 обоснован следующим образом:

1. Производительность и эффективность: Модель YOLOv5 из семейства YOLO обладает хорошей скоростью обработки данных (FPS) при использовании графического процессора NVIDIA GeForce RTX 4090 с поддержкой CUDA. Это позволяет обрабатывать видеопотоки быстро и эффективно, что важно для работы в реальном времени и с большим объемом данных.
2. Соотношение качества и производительности: Результаты тестирования показали, что модели YOLOv5 демонстрируют лучшее сочетание качества работы (показатель mAP) и скорости обработки данных по сравнению с другими версиями YOLO и с другими моделями, такими как SSD и Faster R-CNN. Это означает, что модель способна обеспечить высокую точность обнаружения объектов при достаточно высокой скорости работы.
3. Применимость к обработке видеоданных: Учитывая, что в проекте требуется обработка видео с различных источников в реальном времени, модель YOLOv5 может эффективно справляться с этой задачей благодаря своей производительности и возможности работать с видеопотоками.

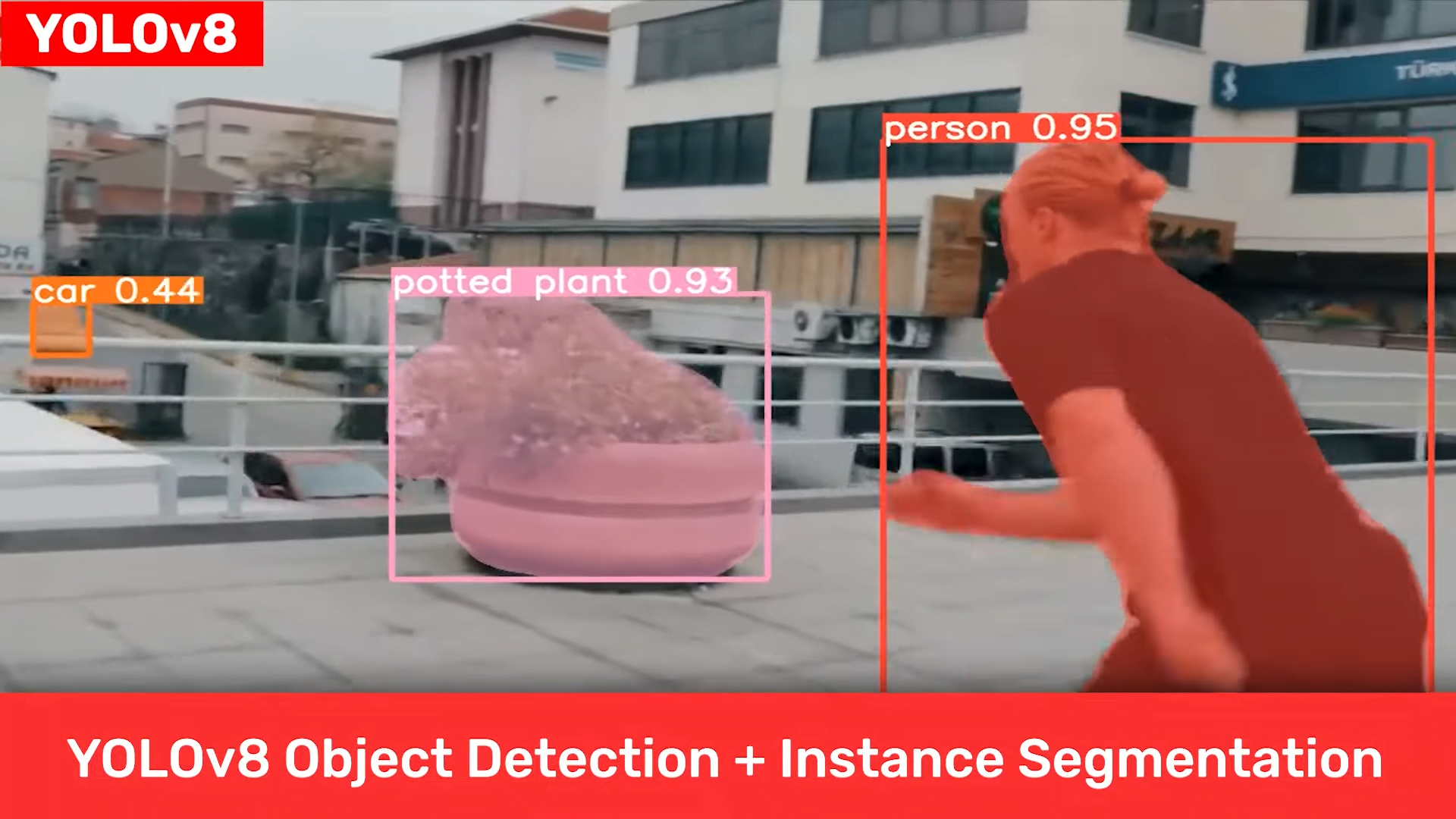


1. Лучшее сочетание качества и производительности: Часто уменьшение размера модели приводит к увеличению скорости обработки данных, но снижению точности. Однако результаты показали, что модели YOLOv5 сохраняют приемлемый уровень точности при высокой скорости обработки данных, что делает их оптимальным выбором для решения поставленной задачи.

Таким образом, модель YOLOv5 является предпочтительным выбором для расширения проекта на автоматический сбор и анализ данных в реальном времени благодаря своей высокой производительности, хорошему соотношению качества и скорости работы, а также способности эффективно работать с видеоданными из различных источников.

**YOLOv8 и YOLOv5**

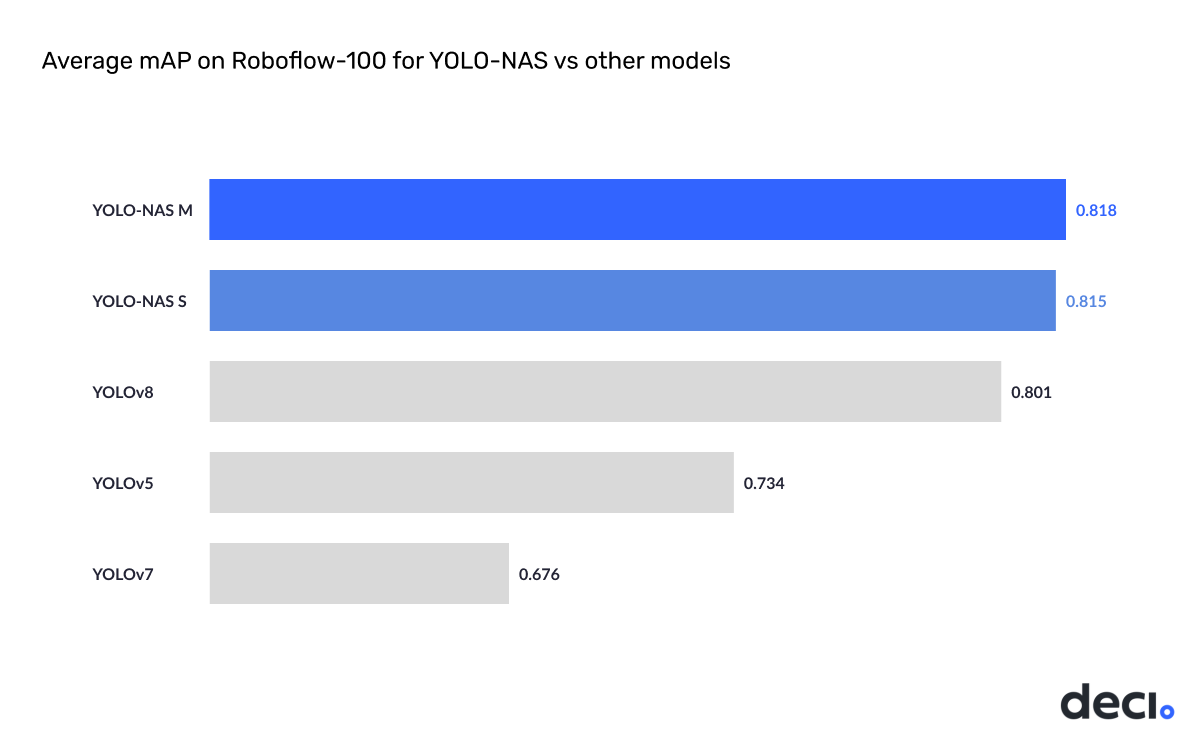
Из предоставленной информации видно, что модель YOLOv8 включает несколько улучшений и нововведений по сравнению с предыдущими версиями, что делает ее более быстрой, точной и эффективной.



Новые возможности YOLOv8 включают:

1. Увеличение скорости и повышение точности: Модель YOLOv8 улучшила свою производительность по сравнению с предыдущими версиями, обеспечивая более высокую скорость обработки данных и точность обнаружения объектов. Это является важным преимуществом при работе с видеопотоками и большим объемом данных.
2. Новая основная сеть (backbone): Использование новой основной сети для извлечения признаков позволяет улучшить качество обнаружения объектов, так как хорошо выбранная backbone сеть способствует более эффективному извлечению признаков из изображений.
3. Anchor-Free head: Внедрение нового алгоритма для определения областей нахождения объектов около якорных точек позволяет улучшить точность детектирования объектов и упростить процесс обнаружения.
4. Новая функция потерь: Использование новой функции потерь помогает оптимизировать обучение модели, что в конечном итоге может привести к улучшению ее обобщающей способности и точности.
5. Разные модели для разных целей: Категория модели YOLOv8 включает в себя несколько вариантов (YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, YOLOv8x), отличающихся по скорости и точности. Это позволяет выбрать подходящую модель в зависимости от конкретных требований проекта.

Сравнение моделей YOLOv5 и YOLOv8 позволит определить, какая из них лучше подходит для задачи обработки изображений с разрешением 224x224 и 640x640.



При этом следует учитывать новые возможности и улучшения, внедренные в модель YOLOv8, которые могут значительно повлиять на ее производительность и точность по сравнению с YOLOv5.

Понятно, что модель YOLOv5 обладает некоторыми преимуществами в контексте большей базы мануалов и инструкций, стабильности и доступности информации об ошибках и багах благодаря своему более длительному использованию. Это важные факторы, особенно при работе в больших нейронных сетях и в условиях, когда возможны проблемы и требуется обширная поддержка со стороны сообщества.

Кроме того, учитывая, что модель YOLOv5 была выпущена в 2020 году и уже получила множество исправлений ошибок и багов, это подтверждает ее относительную стабильность и зрелость как продукта. Наличие обширной информации об ошибках и багах также является важным плюсом для разработчиков и исследователей, поскольку это может значительно упростить процесс разработки и отладки.

В целом, несмотря на то, что более современные модели YOLOv8 могут обладать лучшей точностью и производительностью, у YOLOv5 есть свои преимущества в виде большой базы знаний, стабильности и доступности информации об ошибках и багах. Поэтому выбор между этими моделями должен учитывать, как технические характеристики, так и практические аспекты, связанные с поддержкой и доступностью ресурсов для разработки и внедрения модели в конкретном проекте.

YOLOv8 в качестве детектора объектов:

1. Скорость и точность: Модель YOLOv8 обладает превосходной скоростью и точностью по сравнению с предыдущими версиями, что делает ее отличным выбором для обнаружения объектов в реальном времени.
2. Обширная база данных и сообщество: С момента релиза модели прошло достаточно времени для исправления ошибок и создания обширной базы знаний, что обеспечивает надежную поддержку и доступность информации в случае возникновения проблем.
3. Возможность замены: В случае возникновения критических ошибок, которые невозможно исправить, замена модели может быть выполнена в кратчайшие сроки благодаря ее современности и распространенности.

DeepSORT в качестве трекера объектов:

1. Работа в режиме реального времени: DeepSORT является легковесным решением, позволяющим эффективно работать в режиме реального времени и обеспечивать отслеживание объектов с высокой скоростью.
2. Простая интеграция с YOLO: DeepSORT имеет удобный интерфейс интеграции с YOLO, что облегчает их совместное использование для выполнения задачи обнаружения и трекинга объектов.
3. Поддержка сообщества и наличие руководств: Наличие активных форумов и комьюнити упрощает поиск решений и помогает быстро разрешать возможные проблемы при использовании DeepSORT.
4. Специализация на трекинге людей: DeepSORT изначально обучен для трекинга людей в кадрах, что делает его хорошим выбором для вашей конкретной задачи.

В целом, выбор моделей YOLOv8 и DeepSORT обоснован и учитывает не только технические характеристики, но и практические аспекты и потребности проекта, такие как скорость, точность, готовность к использованию в режиме реального времени и поддержка сообщества.

**Определение конечного результата, определения контрольных параметров успешной реализации.**

Цель проекта:

Создание системы на основе искусственного интеллекта и машинного зрения для контроля пешеходного трафика посредством видеозаписей или видео в реальном времени в автоматическом режиме с точностью не менее 30%.

Ключевые показатели (KPI):

1. Анализ видеофайлов в автономном режиме:

* Контрольный параметр: Способность системы проводить анализ видеофайлов без участия оператора.
* Количественный показатель: Доля успешно обработанных видеофайлов без ошибок, например, 95% обработанных файлов без ошибок.

1. Ведение базы данных статистики:

* Контрольный параметр: Создание и поддержание базы данных собранной статистики о пешеходном трафике.
* Количественный показатель: Количество записей в базе данных за определенный период времени (например, в день/неделю/месяц).

1. Увеличение качества подбора коммерческой недвижимости в 2 раза:

* Контрольный параметр: Изменение в качестве подбора коммерческой недвижимости после внедрения системы.
* Количественный показатель: Увеличение числа успешных сделок или повышение удовлетворенности клиентов в результате улучшенного контроля пешеходного трафика.

Резюме:

Ключевые показатели (KPI) для данного проекта включают в себя анализ видеофайлов в автономном режиме, ведение базы данных статистики и увеличение качества подбора коммерческой недвижимости. Для каждого из этих параметров указаны количественные показатели, которые помогут определить достижение промежуточных контрольных точек и конечного результата.

Для полноценной работы проекта, включая обработку видеозаписей машинным зрением на базе искусственного интеллекта, требуется выбрать оптимальное оборудование. В данном случае, потребуются высокопроизводительный персональный компьютер (рабочая станция) и сервер для хранения видеозаписей и статистики.

Высокопроизводительный компьютер (Рабочая станция):

1. Процессор: Рекомендуется выбрать процессор с высокой мощностью и подходящим количеством ядер для параллельной обработки данных машинным зрением. Процессоры серии Intel Core i9 или AMD Ryzen Threadripper могут быть подходящими вариантами.
2. Графический процессор (GPU): Для ускорения работы с машинным зрением необходимо иметь мощный GPU. GPU серии NVIDIA RTX или NVIDIA Quadro обеспечивают высокую производительность для обучения и выполнения моделей глубокого обучения.
3. Оперативная память: Рекомендуется установить не менее 32 ГБ оперативной памяти для эффективной обработки данных и запуска моделей машинного зрения.
4. Хранение данных: SSD-накопители обеспечат высокую скорость загрузки и обработки данных.

Сервер для хранения данных:

1. Хранение видеозаписей и статистики: Необходимо учесть объем хранения видеозаписей и статистики для выбора соответствующего объема хранилища (HDD или SSD).
2. Вычислительные ресурсы: В зависимости от задач обработки и анализа, возможно также потребуется оборудование с высокопроизводительными процессорами и достаточным объемом оперативной памяти для эффективной обработки данных.

При выборе оборудования также важно учесть возможность масштабирования системы в будущем, а также обеспечить необходимые меры безопасности и резервного копирования данных.

**Оборудование и ресурсы для коммерческой реализации проекта**

1. Для формирования бухгалтерской отчетности и другой документации:

* Компьютер с высокой производительностью и программами для работы с документами (офисные пакеты, программное обеспечение для бухгалтерской отчетности).

1. Видеокамеры и аксессуары:

* Для постоянного размещения: антивандальные видеокамеры.
* Для временного размещения: видеорегистраторы в автомобилях или IP-видеокамеры.

1. Сопутствующее оборудование:

* Маршрутизаторы, карт-ридеры, источники бесперебойного питания, принтеры и МФУ для офисных нужд.

1. Программное обеспечение:

* Операционные системы, офисные пакеты, программное обеспечение для бухгалтерской отчетности, антивирусы, редакторы видеофайлов, СУБД, редакторы исходного кода.

1. Мебель, расходные материалы и канцтовары:

* Столы, стулья, шкафы, расходные материалы для принтеров, канцелярские товары.

1. Аренда помещения и заработная плата:

* Помещение для работы, оплата за аренду, заработная плата сотрудникам.

1. Настройка оборудования и программного обеспечения:

* Затраты на первичную настройку всего оборудования и программного обеспечения.

Для коммерческой реализации проекта необходимо учитывать все вышеперечисленные аспекты, чтобы обеспечить эффективную работу системы и бизнес-процессов. Также важно провести анализ затрат и бюджетирование для правильного планирования финансовых ресурсов. Если требуется дополнительная информация или консультация по каким-либо конкретным аспектам, не стесняйтесь уточнить.

При выборе оборудования для реализации проекта, включая обработку видеозаписей машинным зрением и бухгалтерскую отчетность, важно учитывать следующие рекомендации:

Высокопроизводительный персональный компьютер (рабочая станция):

1. Процессор: Желательно выбрать процессор, совместимый с выбранной видеокартой NVIDIA Geforce RTX 4080 и выше, например Intel Core i9 или AMD Ryzen Threadripper.
2. Видеокарта: NVIDIA Geforce RTX 4080 или аналогичная модель, предназначенная для работы с искусственным интеллектом и машинным зрением.
3. Оперативная память: Рекомендуется установить 32 ГБ оперативной памяти для эффективной обработки данных и запуска моделей машинного зрения.
4. Хранение данных: Обеспечьте быстрый твердотельный накопитель (SSD) для быстрой загрузки и обработки данных.
5. Периферийные устройства: Учтите необходимость монитора, клавиатуры и мыши.

Сервер для хранения данных:

1. Процессор: Выберите средний сегмент процессоров со встроенным видеоускорителем, который соответствует потребностям проекта.
2. Оперативная память: Рекомендуется установить не менее 8 ГБ оперативной памяти для обработки данных и функционирования сервера.
3. Хранение данных: Подберите жесткие диски общим объемом 10 ТБ с необходимой скоростью чтения/записи данных.
4. Материнская плата: Удостоверьтесь, что материнская плата имеет достаточное количество разъемов для подключения выбранных жестких дисков.

Стоимость:

* Для рабочей станции: примерно 280 000 руб., включая периферийные устройства.
* Для сервера: примерно 80 000 руб., включая все необходимое оборудование.

Исходя из предоставленной информации, стоимость комплекта оборудования, программного обеспечения и прочих расходов для начального этапа работы проекта определяется следующим образом:

1. Оборудование:

* Видеокамеры:
* Ip-видеокамера
* Регистратор
* Антивандальная камера
* Комплектующие (кабеля, крепления, карты памяти)
* Суммарная стоимость всех трех комплектов: 30 тыс. руб

1. Сопутствующее оборудование:

* Два источника бесперебойного питания: 10 тыс. руб
* Карт-ридер: 4 тыс. руб
* Черно-белое МФУ: 16 тыс. руб
* Прочее сопутствующее оборудование: 30 тыс. руб

1. Настройка оборудования: 100 тыс. руб

Программное обеспечение:

1. Бесплатное программное обеспечение:

* Операционная система на базе Linux для основного компьютера
* Видеоредакторы, офисные пакеты и редакторы исходного кода

1. Платное программное обеспечение:

* Операционная система Windows: 4 тыс. руб
* Подписка на антивирус: 5 тыс. руб
* Программное обеспечение для бухгалтерской отчетности: 6 тыс. руб
* Суммарная стоимость платного программного обеспечения: 15 тыс. руб

Расходы на офис, мебель и зарплату:

* Расходы на первые три месяца работы: 480 тыс. руб

Итоговая сумма для начального этапа проекта:

Оборудование + программное обеспечение + прочие расходы + настройка =

30 тыс. руб + 15 тыс. руб + 480 тыс. руб + 100 тыс. руб = 625 тыс. руб

Таким образом, исходя из предоставленной информации, стоимость необходимого оборудования и программного обеспечения для начального этапа проекта составляет 625 тыс. руб, что немного отличается от указанной вами суммы 1 млн. 15 тыс. руб.

**Код и настройка**

Установка необходимых библиотек:

python

pip install tensorflow

pip install nympy

pip install os

pip install random

pip install cv2

pip install ultralytics

pip install scipy

Здесь происходит установка нескольких библиотек, но следует отметить, что "os" и "random" являются стандартными библиотеками Python и не требуют отдельной установки с помощью pip.

Создание файла трекера tracker.py:

python

from scipy.optimize import linear\_sum\_assignment as linear\_assignment

from deep\_sort.deep\_sort.tracker import Tracker as DeepSortTracker

from deep\_sort.tools import generate\_detections as gdet

from deep\_sort.deep\_sort import nn\_matching

from deep\_sort.deep\_sort.detection import Detection

import numpy as np

В этой части кода импортируются необходимые модули и классы из библиотек. Например, linear\_sum\_assignment используется для решения задачи о назначении в алгоритме трекинга, а DeepSortTracker, generate\_detections, nn\_matching и Detection являются частями библиотеки DeepSORT.

python

class Tracker:

tracker = None

encoder = None

tracks = None

def \_\_init\_\_(self):

# Инициализация параметров трекера, включая загрузку модели энкодера

...

def update(self, frame, detections):

# Метод обновления трекера на основе переданных кадра и детекций

...

def update\_tracks(self):

# Метод обновления списка треков на основе трекера

...

Этот фрагмент содержит определение класса Tracker с методами \_\_init\_\_, update и update\_tracks, которые позволяют инициализировать трекер, обновлять его состояние и получать информацию о текущих треках соответственно.

python

class Track:

track\_id = None

bbox = None

def \_\_init\_\_(self, id, bbox):

# Класс для хранения информации о треках

...

Вышеописанный класс Track представляет собой простую структуру данных для хранения информации о треке, такой как его идентификатор и ограничивающий прямоугольник (bbox).

Это базовая структура кода для реализации трекера на основе DeepSORT. Все функции для обновления трекера и обработки детекций описаны внутри класса Tracker.

Шаги для подготовки работы трекера на основе DeepSORT:

1. Скачивание модели mars-small128:

* Необходимо найти и скачать модель mars-small128 для работы трекера. Сохраните файл в папку model\_data вашего проекта.
* Примерный путь к файлу: model\_data/mars-small128.pb

1. Клонирование официальной сборки DeepSORT:

* Найти ссылку на официальный репозиторий DeepSORT на GitHub.
* Клонировать репозиторий в локальное хранилище с помощью команды: git clone https://github.com/computervisioneng/deep\_sort.git

1. Изменение строки в файле linear\_assignment.py:

Если установлены последние версии библиотек, необходимо заменить одну строку в файле linear\_assignment.py (в папке deep\_sort/deep\_sort) клонированного репозитория.

* Находим и заменяем следующую строку:

from sklearn.utils.linear\_assignment\_ import linear\_assignment

на: from scipy.optimize import linear\_sum\_assignment as linear\_assignment

* Это необходимо из-за различий в библиотеках между старыми и новыми версиями, чтобы обеспечить корректную работу трекера.

После выполнения этих шагов трекер на основе DeepSORT должен быть готов к использованию с вашей моделью и библиотеками.

Шаги для создания основного инициализатора системы машинного зрения ([main.py](http://main.py/)) следующие: Проверка видео и инициализация трекера:

1. Инициализация видеоанализа:

* Используем библиотеку OpenCV для чтения видео и отображения его содержимого.

import cv2

# Открываем видео файл

video\_path = 'path\_to\_your\_video.mp4'

cap = cv2.VideoCapture(video\_path)

# Проверяем, открыто ли видео

if not cap.isOpened():

print("Error opening video file")

exit()

1. Добавление кода для детекции и инициализации трекера:

* Импортируем ранее созданный файл трекера и инициализируем трекер.
* Обработаны полученные координаты и классы объектов с помощью трекера.

from tracker import Tracker

# Инициализация трекера

tracker = Tracker()

while cap.isOpened():

ret, frame = cap.read()

if not ret:

break

# ... Код для обнаружения объектов на кадре

# Обновление трекера

tracker.update(frame, detections)

# Обновление списка треков

tracker.update\_tracks()

# ... Другие операции с трекером

1. Сохранение видео с результатом:

* Записать обработанные кадры в новое видео с использованием библиотеки OpenCV.

# ... (код для обработки и отображения окончательных результатов)

# Создание объекта для записи видео

output\_video = 'output\_video.mp4'

fourcc = cv2.VideoWriter\_fourcc(\*'mp4v')

out = cv2.VideoWriter(output\_video, fourcc, fps, (width, height))

# Запись обработанных кадров

out.write(processed\_frame)

# Освобождение ресурсов

out.release()

1. Установка параметров фильтрации объектов:

* Добавьте условия для игнорирования объектов с низкой вероятностью и учета только объектов с определенным классом.

# Пример фильтрации по вероятности и классу

for detection in detections:

if detection.probability > 0.5 and detection.class == 0:

# Обработка объекта

# ...

1. Вывод количества людей на видео:

* В конце цикла нужно добавить вывод количества уникальных id для определения числа людей на видео.

# Вывод количества уникальных id

print("Количество людей на видео:", len(tracker.tracks))

Это лишь примерная структура кода для основного инициализатора системы машинного зрения.

import os

import random

import cv2

from ultralytics import YOLO

from tracker import Tracker

video\_path = os.path.join('.', 'data', 'test.mp4')

video\_out\_path = os.path.join('.', 'test\_out.mp4')

cap = cv2.VideoCapture(video\_path)

ret, frame = cap.read()

cap\_out = cv2.VideoWriter(video\_out\_path, cv2.Video\_fourcc(\*'MP4V'), cap.get(cv2.CAP\_PROP\_FPS),

(int(frame.shape[1]), int(frame.shape[0])))

model = YOLO("yolov8n.pt")

tracker = Tracker()

colors = [(random.randint(0, 255), random.randint(0, 255), random.randint(0, 255)) for \_ in range(10)]

detection\_threshold = 0.5

while ret:

results = model(frame)

for result in results:

detections = []

for r in result.boxes.data.tolist():

x1, y1, x2, y2, score, class\_id = r

x1 = int(x1)

x2 = int(x2)

y1 = int(y1)

y2 = int(y2)

class\_id = int(class\_id)

if score > detection\_threshold and class\_id == 0:

detections.append([x1, y1, x2, y2, score])

tracker.update(frame, detections)

for track in tracker.tracks:

bbox = track.bbox

x1, y1, x2, y2 = bbox

track\_id = track.track\_id

cv2.rectangle(frame, (int(x1), int(y1)), (int(x2), int(y2)), colors[track\_id % len(colors)], 3)

cap\_out.write(frame)

ret, frame = cap.read()

cap.release()

cap\_out.release()

cv2.destroyAllWindows()

print(len(tracker.tracks))

**Тест и результаты**

Результаты и анализ, которые проведены, крайне полезны для оценки эффективности проекта. Рассмотрим ключевые моменты:

Успехи системы:

1. Точность определения объектов: Ваша система уже показывает точность определения объектов выше целевого показателя, что свидетельствует о хорошем начале проекта. Это может быть ключевым преимуществом при маркетинге системы.
2. Автоматический анализ видеозаписей: Вы успешно организовали автоматический анализ видеозаписей, что является важным достижением. Это дает возможность использовать систему для обработки больших объемов видеоданных без необходимости ручного участия.

Неудачи и ограничения:

1. База данных статистики: Организация базы данных статистики по видеозаписям в условиях сжатых сроков не удалась. Это может быть областью для дальнейшего улучшения. Рассмотрите возможность использования более эффективных методов сбора и анализа данных для более оперативной работы системы.
2. Коммерческая эксплуатация: Проект пока не запущен в коммерческую эксплуатацию из-за определенных ограничений. Важно продолжить работу над устранением этих ограничений для последующего коммерческого запуска.

Анализ экономической эффективности:

Расчеты экономической эффективности проекта указывают на следующие ключевые факторы:

1. Окупаемость проекта: При средней окупаемости в 3 года проект может быть рассмотрен как приемлемый с точки зрения инвестиций.
2. Расходы на текущую эксплуатацию: Постоянные расходы на интернет, электричество и другие ресурсы в размере 10 тыс. рублей должны быть учтены в общем бизнес-плане.
3. Потенциальный доход: Если средний доход составляет 188 тыс. рублей в месяц, это может быть положительным индикатором для инвесторов.

Дальнейшие шаги:

1. Улучшение базы данных: Попробуйте найти способы улучшения организации базы данных статистики для более эффективного использования данных. Рассмотрите внедрение автоматизированных систем сбора и анализа данных.
2. Подготовка к коммерческому запуску: Вам следует продолжить подготовку к запуску проекта в коммерческую эксплуатацию, учитывая текущие ограничения. Обратитесь к разработке плана действий для устранения ограничений и запуска проекта.

Экономический анализ: расчеты прибыли и окупаемости проекта предоставляют важную информацию для будущих инвестиций. Следует также продолжить изучать дополнительные параметры, которые могут повлиять на экономическую эффективность проекта, такие как потенциальные изменения в расходах и доходах.

**Заключение**

В настоящее время исследователи активно занимаются разработкой методов объяснения принятия решений нейронных сетей, которые позволяют интерпретировать результаты и выделять ключевые факторы, влияющие на прогнозы или классификации. Такие методы включают в себя генерацию heatmaps для визуализации важности признаков, использование методов "линий влияния", а также различные подходы к анализу влияния входных данных на выход модели.

Разработка более прозрачных и объяснимых моделей машинного обучения поможет повысить доверие к этим технологиям и улучшить их применимость в широком спектре областей. Это также способствует эффективной коммуникации результатов между специалистами в области ИИ, заказчиками и конечными пользователями.

Обеспечение прозрачности и объяснимости результатов работы нейронных сетей представляет собой серьезный вызов для сообщества, особенно в критических областях. Разработка методов объяснения принятия решений нейронными сетями имеет важное значение для повышения доверия к технологиям искусственного интеллекта и их успешной интеграции в различные отрасли.

**Список литературы**

1. Python и наука данных для чайников, 2-е издание - год выпуска 2020, издание, авторы Джон Пол Мюллер, Лука Массарон.
2. Comparison of YOLO v3, Faster R-CNN, and SSD for Real-Time Pill Identication - <https://assets.researchsquare.com/files/rs-668895/v1_covered.pdf?c=1631875157>
3. Ultralytics YOLOv8 - <https://habr.com/ru/articles/710016/>
4. Сравнение YOLO v5 и Faster R-CNN для обнаружения людей на изображении в потоковом режиме - <https://cyberleninka.ru/article/n/sravnenie-yolo-v5-i-faster-r-cnn-dlya-obnaruzheniya-lyudey-na-izobrazhenii-v-potokovom-rezhime>
5. Компьютерное зрение: технологии, компании, тренды - <https://rdc.grfc.ru/2021/04/analytics-computer-vision/>
6. Interpretable Machine Learning: A Guide for Making Black Box Models Explainable - <https://christophm.github.io/interpretable-ml-book/>
7. Как работает Object Tracking на YOLO и DeepSort - <https://habr.com/ru/articles/514450/>
8. Practical Statistics for Data Scientists: 50+ Essential Concepts Using R and Python – <https://www.amazon.com/Practical-Statistics-Data-Scientists-Essential/dp/149207294X>
9. Сравнение производительности моделей обнаружения объектов YOLO v5, v6 и v7 - <https://cyberleninka.ru/article/n/sravnenie-yolo-v5-i-faster-r-cnn-dlya-obnaruzheniya-lyudey-na-izobrazhenii-v-potokovom-rezhime>
10. NBDT: интерпретируемые нейронные деревья решений - <https://neurohive.io/ru/vidy-nejrosetej/nbdt-interpretiruemye-nejronnye-derevya-reshenij/>