

Реферат

На - страниц, - рисунков, - таблицы, - приложение

КЛЮЧЕВЫЕ СЛОВА: КОМПЬЮТЕРНОЕ ЗРЕНИЕ, НЕЙРОННЫЕ СЕТИ, ОБНАРУЖЕНИЕ ОБЪЕКТОВ, ОТСЛЕЖИВАНИЕ НЕСКОЛЬКИХ ОБЪЕКТОВ, ПЕРЕИДЕНТИФИКАЦИЯ ПЕШЕХОДОВ, YOLO, Улучшенный DeepSORT алгоритм

Выпускная квалификационная работа на тему: «Применение нейронных сетей для оценки трафика магазина».

В данной работе рассматривается реализации алгоритмов нейронных сетей для оценки трафика в магазине. Для решения поставленных задач были использованы методы глубокого обучения и компьютерного зрения.

Задачи, решаемые в ВКР:

1. Для обнаружения пешеходов был использован алгоритм YOLOv5
2. Для установления связи между персонажами при работе с несколькими камерами был реализован алгоритм повторного распознавания (reid).
3. Для отслеживания локальных зон был реализован алгоритм DeepSORT он основе повторного распознавания.
4. На основе методики, описанных в данной статье, была разработана система оценки трафика посетителей для супермаркета.

В результате исследования была создана система оценки трафика посетителей для супермаркетов, которая может идентифицировать входящих и выходящих людей с точностью до 95,5% и повторную идентификацию пешеходов с точностью до 85,0%. В конце, на других больших открытых наборах данных мы сравнили реализации DeepSORT, FastReID с другими популярными алгоритмами и добились неплохих результатов.

Abstract

Оглавление

| | |
|---|----|
| Ведение..... | 5 |
| ГЛАВА 1. ОБЗОР ЛИТЕРАТУРЫ | 6 |
| Обноружевание пешеходов..... | 6 |
| Алгоритм распознавания объектов YOLO..... | 9 |
| Отслеживание людей в виде последовательности | 10 |
| Алгоритм трекинга множества объектов DeepSORT..... | 11 |
| Набор данных MOT-16..... | 12 |
| Повторная идентификация людей..... | 13 |
| Алгоритм FastReID | 17 |
| Набор данных MSMT17 | 19 |
| Метрики оценки эффективности модели | 20 |
| Подсчета количества людей в видеопоследовательности | 22 |
| ГЛАВА 2. ОБУЧЕНИЕ И УСТРАИВАНИЕ МОДЕЛЕЙ | 22 |
| Обучение модели FastReID | 22 |
| Принцип обучения | 22 |
| Построить модель | 23 |
| Функции потери | 24 |
| Ходы обучения | 25 |
| Результаты обучения | 27 |
| Устраивание модели DeepSORT | 28 |
| ГЛАВА 3. СИСТЕМА ОЦЕНКИ ТРАФИКА МАГАЗИНА | 29 |
| Цели разработки системы | 29 |
| Описание системы | 30 |

| | |
|---|----|
| Общая архитектура системы | 30 |
| Модуль обнаружения пешеходов..... | 30 |
| Модуль отслеживания пешеходами..... | 31 |
| Модуль подсчета и распознавания пешеходного | 31 |
| Модуль хранения | 34 |
| Среда разработки | 34 |
| Демонстрация и тестирование системы | 35 |
| Общая принципиальная схема системы | 35 |
| Функционалы сисетмы | 36 |
| ГЛАВА 4. АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ | 42 |
| Экспериментальная оценка модели FastReID на открытом наборе данных | 42 |
| Сравнительное тестирование улучшенного алгоритма DeepSORT с исходном алгоритмом DeepSORT | 43 |
| Сравнение улучшенного алгоритма DeepSORT с несколькими основными алгоритмами | 45 |
| Экспериментальная оценка улучшенной модели DeepSORT на открытом наборе данных | 46 |
| Экспериментальные оценки системы оценки трафика магазина | 48 |
| ЗАКЛЮЧЕНИЕ | 55 |
| Список Литературы..... | 57 |

ВЕДЕНИЕ

С развитием розничной торговли и усилением конкуренции, многие отечественные торговые центры осознали важность данных о потоках посетителей для принятия эффективных бизнес-решений. Изучение потока покупателей является ключевым элементом для научного управления торговыми центрами во всех аспектах и повышения комфорта торговой среды.

Традиционный метод искусственного подсчета, который представляет собой непрерывное визуальное подсчитывание посетителей на входе и выходе, не требует больших капиталовложений, однако его использование ограничено трудовыми и материальными ресурсами и подвержено ошибкам.

С появлением механического сенсорного оборудования, инфракрасного сенсорного оборудования и оборудования для камер наблюдения, методы подсчета посетителей прошли две стадии: статистику физического прикосновения и статистику инфракрасного излучения. Однако, данные методы имеют ряд очевидных недостатков, таких как высокие требования к установке оборудования, низкую скорость и точность подсчета.

Для эффективного управления и поддержки принятия решений для супермаркетов необходим более точный, автоматизированный и эффективный метод. В данном исследовании был предложен алгоритм мониторинга и расчета пассажиропотока супермаркетов с использованием метода нейронных сетей.

Основная цель данной работы заключается в разработке метода, который сочетает в себе алгоритмы обнаружения, отслеживания и повторной идентификации пешеходов, чтобы получить точное время посещения магазина. Для этого были использованы зрелые технологии глубокого обучения, такие как алгоритм YOLOv5, алгоритм отслеживания DeepSORT и алгоритм повторной идентификации пешеходов FastReID.

Предложенный метод демонстрирует высокую выполнимость, основываясь на экспериментах, проведенных в реальных условиях супермаркета. Более того, полученные данные могут быть использованы для эффективного управления и принятия решений в супермаркетах, что повышает их конкурентоспособность на рынке.

Для достижения цели данной работы были выполнены следующие задачи:

- 1) Собрать видеонаблюдения супермаркета и выполнить предварительную обработку;
- 2) Разработать и реализовать модуль обнаружения пешеходов на основе алгоритма YOLOv5;
- 3) Разработать и реализовать алгоритм отслеживания DeepSORT на основе алгоритма повторной идентификации пешеходов FastReID, который применяется для отслеживания пешеходов в видеопотоке;
- 4) Разработать и реализовать алгоритм повторной идентификации пешеходов FastReID для ассоциации персонажей с назначенными посетителями под несколькими камерами;
- 5) Хранить и записывать фотографии и времени входа и выхода посетителей, количество посетителей за определенный период времени.

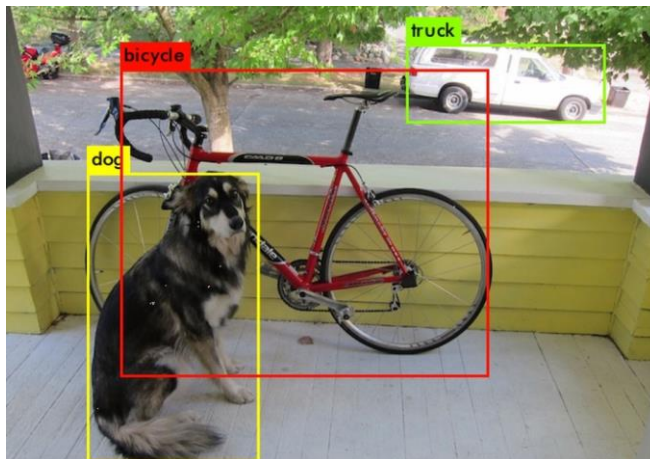
Таким образом, разработанный метод мониторинга и расчета пассажиропотока супермаркетов может эффективно улучшить управление трафиком внутри супермаркета, повысить качество обслуживания клиентов и увеличить прибыльность бизнеса.

ГЛАВА 1. ОБЗОР ЛИТЕРАТУРЫ

Обнаружение пешеходов

Суть обнаружения пешеходов заключается в обнаружении объектов. Обнаружение объектов лежит в основе задач многоцелевого сопровождения, основная цель которого — отметить объект, подлежащий обнаружению, прямоугольной рамкой (bounding box) из статической картинке каждого кадра

и получить конкретное местоположение объекта на картинке. Если при обнаружении объекта происходит пропущенное обнаружение, цель не будет отслеживаться позже, поэтому обнаружение объекта играет очень важную роль в задачах отслеживания нескольких целей.



Появление сверточных нейронных сетей значительно повысило эффективность методов обнаружения объектов. Такие алгоритмы можно условно разделить на следующие две категории. Одна из них представляет собой двухэтапный алгоритм обнаружения, который сначала выбирает области-кандидаты на изображении, а затем выполняет задачи классификации этих областей. Этот тип алгоритма характеризуется высокой точностью, но медленной скоростью. Например, [R-CNN](#), предложенная Россом [1], сначала предварительно определяет области, которые должны быть обнаружены на изображении, с помощью операции выборочного поиска, а затем извлекает признаки этих областей с помощью сверточной нейронной сети и, в конце, классифицирует их. Хотя алгоритм R-CNN значительно повышает точность обнаружения объектов, он имеет проблему вычислительной избыточности. [Алгоритм sppnet](#), предложенный Хе [2], устраняет влияние повторяющегося масштабирования изображений R-CNN путем добавления слоя объединения пространственных пирамид. Кроме того, [Гиршик](#) [3] предложили алгоритм [Fast R-CNN](#), который дополнительно оптимизировал алгоритмы R-CNN и sppnet, добавив слой объединения регионов. [Рен](#) [4] предложили алгоритм

[Faster R-CNN](#), который заменил выборочный поиск сетью предложений регионов и обучил сеть от начала до конца, что значительно повысило скорость детектора. Другой представляет собой одноэтапный алгоритм обнаружения, который выполняет регрессию для вывода положения и категории объекта обнаружения при создании области-кандидата. Репрезентативным алгоритмом является алгоритм серии [YOLO \[5\]](#).

Серия алгоритмов YOLO была впервые предложена Редмоном в 2016. Он отказался от двухэтапной парадигмы обнаружения «обнаружение предложения + проверка» и напрямую предсказал вероятность категории и положение каждой сетки, разделив изображение на несколько сеток, информация.

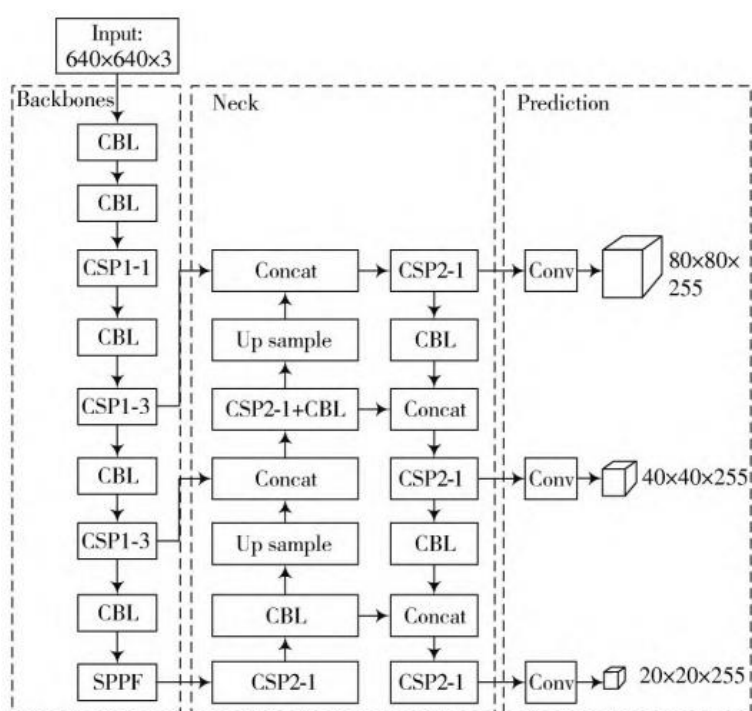
Из приведенного выше введения мы делаем вывод, что традиционные подходы к обнаружению объектов обычно требуют ручного проектирования функций и классификаторов, которые плохо работают в сложных сценах и требуют большого опыта и времени для настройки параметров. Напротив, технология обнаружения объектов на основе глубокого обучения использует глубокие нейронные сети для автоматического изучения функций и классификаторов, что обеспечивает более высокую точность и надежность и подходит для различных сложных сцен. Обнаружение объектов является предпосылкой отслеживания объектов. Мы выбрали YOLOv5 в качестве детектора объектов для исследовательского проекта. [По словам авторов статьи \[6\]](#), на момент своего появления, модель YOLO явилась самой передовой в области обнаружения объектов на изображениях. Она была протестирована на широко используемых наборах данных, таких как PASCAL VOC и COCO, и продемонстрировала высокую точность, превосходящую альтернативные подходы Faster R-CNN и SSD, при этом затрачивая меньше времени на обработку.

Исходя из вышеперечисленных преимуществ, YOLOv5 больше подходит для наших сценариев использования.

Алгоритм распознавания объектов YOLO

YOLOv5 был предложен автором Glenn Jocher. Существует четыре размера моделей, а именно YOLOv5s, YOLOv5m, YOLOv5l и YOLOv5x. Среди них YOLOv5s имеет наименьшее количество сверточных слоев, самую быструю скорость обнаружения и самую низкую точность обнаружения, для остальных трех моделей количество сверточных слоев расположено в порядке возрастания. По мере увеличения сложности модели обнаружение Скорость постепенно снижается, а точность обнаружения постепенно увеличивается.

Структура YOLOv5 состоит из четырех частей: Input, Backbones, Neck и Prediction, как показано на рисунке.



Сторона ввода включает в себя улучшение Mosaic, динамическую опорную рамку, адаптивную обработку изображений и т. Д.; Магистраль включает в себя кросс-этапную локальную сеть CSP, пространственный пирамидный пул SPPF, первый помогает сократить объем вычислений, а второй повышает точность обнаружения; Neck использует FPN+ Структура PAN, понижающая дискретизация улучшает семантическую информацию, повышающая

дискретизация улучшает информацию о местоположении; Прогноз является конечным результатом обнаружения.

Чтобы добиться более высокой скорости, мы выбираем предварительно обученную модель YOLOv5s, выпущенную [веб-сайтом с открытым исходным кодом YOLOv5](#) [7].

Отслеживание людей в виде последовательности

Задача Отслеживание (Трекинг) объектов в видеопотоке заключается в сопоставлении детекция объекта на последовательности кадров видеопотока треку объекта. Задача трекинга множества объектов (Multi-Object Tracking, MOT) заключается в трекинге нескольких различных объектов. [8] В частности, алгоритм трекинга множества объектов в основном решает проблему обнаружения объектов, которые мы хотим отслеживать, в каждом кадре видео, получения позиции на кадре и присвоения идентификатора каждому объекту. В том числе, в процедуру трекинга идентификатор каждого объекта должен оставаться неизменным.



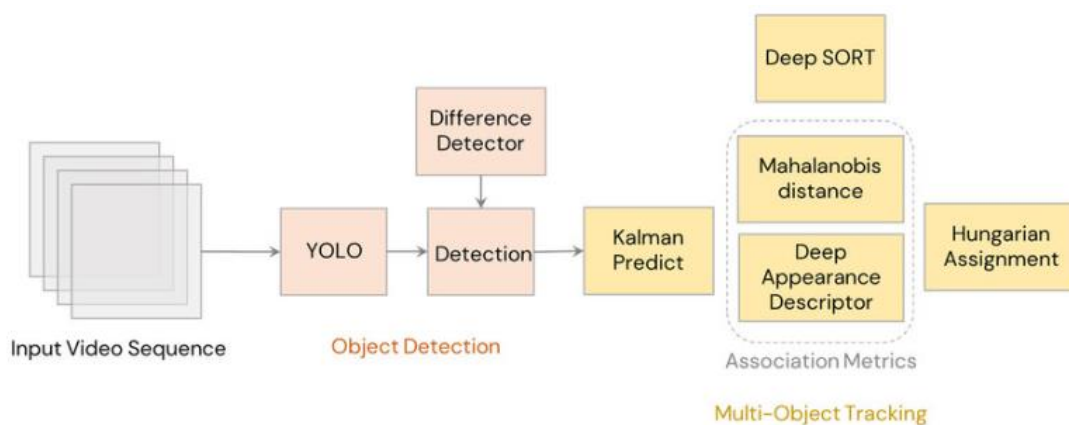
Как показано на рисунке, после ввода последовательности кадров, собранных камерой, в алгоритм MOT выводятся результаты отслеживания нескольких целей в кадре, включая местоположение объекта на изображении и соответствующую идентификационную информацию.

По сравнению с отслеживанием одной цели задачи отслеживания нескольких целей должны не только сталкиваться с такими проблемами, как изменение освещения, окклюзия и размытие движения, но также решать такие проблемы, как неопределенное количество целей и прерывание траекторий целей во время процесса отслеживания. С непрерывным ростом исследований

многоцелевого отслеживания в последние годы, связанные с ними методы появляются бесконечно. Далее рассматривается популярный алгоритм MOT – DeepSORT

Алгоритм трекинга множества объектов DeepSORT

DeepSORT[29] является усовершенствованием алгоритма отслеживания SORT. Основная цель DeepSORT — генерировать непрерывные траектории движения множества персонажей в видеопотоке, а общий рабочий процесс показан на рисунке ниже. Чтобы уменьшить количество переключателей идентификатора цели отслеживания (переключение идентификатора, ID switch), DeepSORT вводит функции внешнего вида и каскадное сопоставление. Алгоритм DeepSORT принимает координаты кадра обнаружения и достоверность результатов обнаружения сети обнаружения объектов в качестве входных данных, предсказывает положение пешеходов в следующем кадре с помощью алгоритма фильтра Калмана, выполняет каскадное сопоставление, а затем использует венгерский алгоритм для ассоциации данных. В конце, фильтр Калмана обновлен.



DeepSORT вычисляет сходство, используя информацию о движении и внешнем виде цели (человека). Для информации о движении расстояние Махаланобиса используется для оценки корреляции между прогнозируемой целью и обнаруженной целью. Выражение расстояния Махаланобиса:

$$d^{(1)}(i, j) = (d_j - y_i)^T S_i^{-1} (d_j - y_i)$$

Где d_j – позиция bounding box j , y_i - позиция, предсказанное фильтром Калмана, S_i -ковариационная матрица обнаруженных и предсказанных позиций.

Когда цель закрыта в течение длительного времени или угол обзора колеблется, необходимо ввести информацию о внешнем виде и использовать косинусное расстояние для решения проблемы переключения идентичности, вызванного окклюзией.

Выражение косинусного расстояния:

$$d^{(2)}(i, j) = \text{Min} \{ 1 - r_j^T r_k^{(i)} \mid r_k^{(i)} \in R_i \}$$

Где r_j — собственный вектор bounding box d_j ; $r_k^{(i)}$ — набор собственных векторов, соответствующих последним 100 кадрам трека i ; R_i — набор векторов признаков внешнего вида.

Чтобы в полной мере использовать два вида информации, для суммирования используется линейно-взвешенный метод:

$$c_{i,j} = \lambda d^{(1)}(i, j) + (1 - \lambda) d^{(2)}(i, j)$$

В формуле: λ является гиперпараметром тогда и только тогда, когда значение измерения $c_{i,j}$ Существует между $d^{(1)}(i, j)$ И $d^{(2)}(i, j)$, только в этом случае он считается относящимся к цели.

Стоит заметить, что, поскольку алгоритм фильтра Калмана основан на линейном равномерном движении, алгоритм DeepSORT может предсказывать состояние цели только в линейной среде и может быть не в состоянии хорошо предсказать, когда траектория пешехода нелинейна.

Набор данных MOT-16

Этот набор данных является общим набором данных, используемым для оценки производительности алгоритмов отслеживания нескольких целей. Существуют различные пешеходные сцены, в том числе 14 видеопоследовательностей, а 7 видеопоследовательностей помечены подробно с идентификаторами пешеходов и позициями кадров, которые

используются для обучать многоцелевых пешеходов Алгоритм слежения, еще 7 видеофрагментов служат тестовой выборкой.

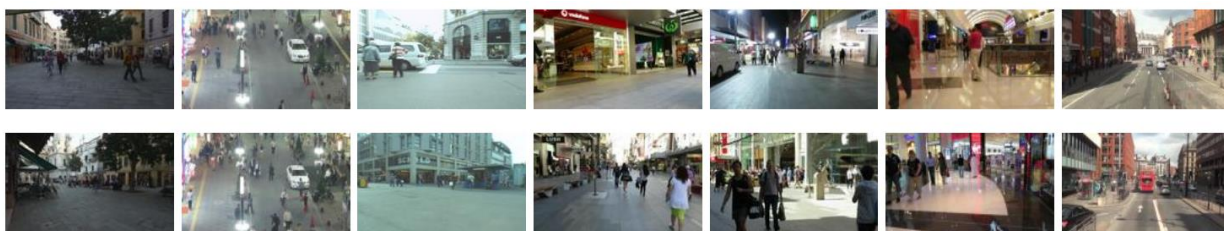


Figure 1 An overview of the MOT16 dataset. Top: Training sequences; bottom: test sequences.

Критерии оценки, использованные в эксперименте: точность сопровождения (MOTA), точность сопровождения (MOTP), количество правильно сопровождаемых траекторий объекта выше 80% (MT), количество правильно сопровождаемых траекторий объекта ниже 20% (ML), смена идентификатора (IDsw, т.е. ID switch), точность отслеживания и точность отслеживания вычисляются следующим образом:

$$\text{MOTA} = 1 - \frac{\sum_t (m_t + n_t + s_t)}{\sum_t g_t}$$

$$\text{MOTP} = \frac{\sum_{i,t} d_i^t}{\sum_t c_t}$$

Где t представляет t -й кадр, m_t представляет количество пропущенных целей обнаружения в t -кадре; n_t представляет количество ложных целей обнаружения в t -кадре; s_t представляет количество переключений идентичности в t -кадре; g_t представляет общее количество целей в кадре t ; расстояние между прогнозируемой позицией объекта кадра i и реальной позицией; c_t представляет количество успешно сопоставленных целей в кадре t .

Повторная идентификация людей

Повторная идентификация множества людей (Person Re-identification, также известная как Ре-идентификация человека или межкамерное сопровождение) — это новая технология, появившаяся в области

интеллектуального анализа видео в последние годы. Ре-идентификация множества людей может быть определена как задача присвоения одного и того же имени или индекса всем образам одного и того же человека, получаемым с пространственно-разнесенных камер, области видимости которых не пересекаются друг с другом, на основе выделения и анализа признаков его изображений.[9] В видеонаблюдении из-за разрешения камеры и угла съемки обычно невозможно получить очень качественные изображения лиц. Когда распознавание лиц не работает, reid становится очень важной замещающей технологией. Очень важной особенностью reid является кросскамерность, поэтому при оценке производительности в научных работах необходимо получать одни и те же изображения пешеходов с разных камер.

Одной из основных проблем является выбор дескриптора, описывающего человека[10]. Для решения указанных задач требуется выявить отличительные признаки объектов, в том числе людей, на изображениях. Это можно выполнить путем сравнения их между собой или с галереей для ре-идентификации. Однако поиск таких признаков является неформализованным и требует эмпирического подхода, что может быть долгим и трудоемким процессом. В связи с этим, повторная идентификация людей может потребовать большого количества времени из-за неоднозначности их внешнего вида под разными ракурсами, освещением и другими факторами. Недавние достижения в области глубокого обучения и развитие сверточных нейронных сетей (СНС) позволили автоматизировать процесс извлечения признаков и значительно увеличить точность повторной идентификации.

Наиболее часто используемая концепция повторной идентификации пешеходов:

При выполнении задачи повторной идентификации пешеходов мы обычно разделяем все изображения пешеходов на две группы: набор запросов (Query) и набор галерей (Gallery).

Набор запросов (Query): коллекция изображений, которые мы хотим использовать для поиска подобных изображений в наборе галерей.

Набор галерей (Gallery): коллекция всех доступных изображений для поиска.

Запрос (probe или query image): одно изображение или группа связанных изображений для поиска. В нашем случае это изображение человека.

Обычно мы используем запрос("probe") для представления каждого изображения в наборе запросов, которое является целью нашего поиска, а "галерея" представляет собой набор изображений, в которых мы ищем соответствующие изображения. Цель повторной идентификации пешеходов заключается в том, чтобы найти все изображения пешеходов в наборе галереи, которые похожи или совпадают с изображениями в наборе запроса, и могут быть точно аутентифицированы или идентифицированы.



Задача повторной идентификации пешеходов в основном состоит из двух процесса: процесса извлечения признаков и процесса измерения сходства. **Извлечение признаков** - это обучение особенностей, способных справиться с изменениями пешеходов под разными камерами. **Обучение измерения** - это отображение обученных признаков в пространство более высокой размерности, таким образом, чтобы одинаковые люди находились ближе друг к другу, а разные люди - дальше.

Традиционный метод заключается в ручном извлечении признаков изображения, например, цвета, **HOG** (гистограмма ориентированных градиентов)[11], **SIFT** (масштабно-инвариантное преобразование

функций)[12], LOMO (локальный максимальный возникновение). Затем используется XQDA (кросс-видовой квадратичный дискриминантный анализ)[13] или KISSME (простое и прямолинейное изучение метрики)[14] для изучения лучшего измерения сходства. Однако традиционные методы ручного описания особенностей имеют ограниченные возможности и трудно применяются к задачам с большим объемом данных в сложных сценах. Кроме того, при обработке больших объемов данных традиционные методы обучения измерений также становятся очень сложными.

В последние годы глубокое обучение, представленное сверточными нейронными сетями, достигло больших успехов в области компьютерного зрения, победив традиционные методы в многих задачах и даже в какой-то мере [превзойдя уровень человека \[15-16\]](#). В случае повторной идентификации пешеходов методы на основе глубокого обучения могут автоматически извлекать сложные описания признаков и использовать простые евклидовы расстояния для измерения сходства, что дает хорошие результаты. Другими словами, глубокое обучение может реализовать задачу повторной идентификации пешеходов полностью, что делает задачу более простой. В настоящее время методы повторной идентификации пешеходов на основе глубокого обучения значительно превосходят традиционные методы по производительности. Эти преимущества делают глубокое обучение популярным в области повторной идентификации пешеходов.

Принципы повторной идентификации

Повторная идентификация пешеходов обычно используется в сочетании с отслеживанием пешеходов. Эти две технологии составляют интеллектуальную видеосистему с [отслеживанием и повторной идентификацией людей](#). На рис. показана общая схема этой видеосистемы.[9]

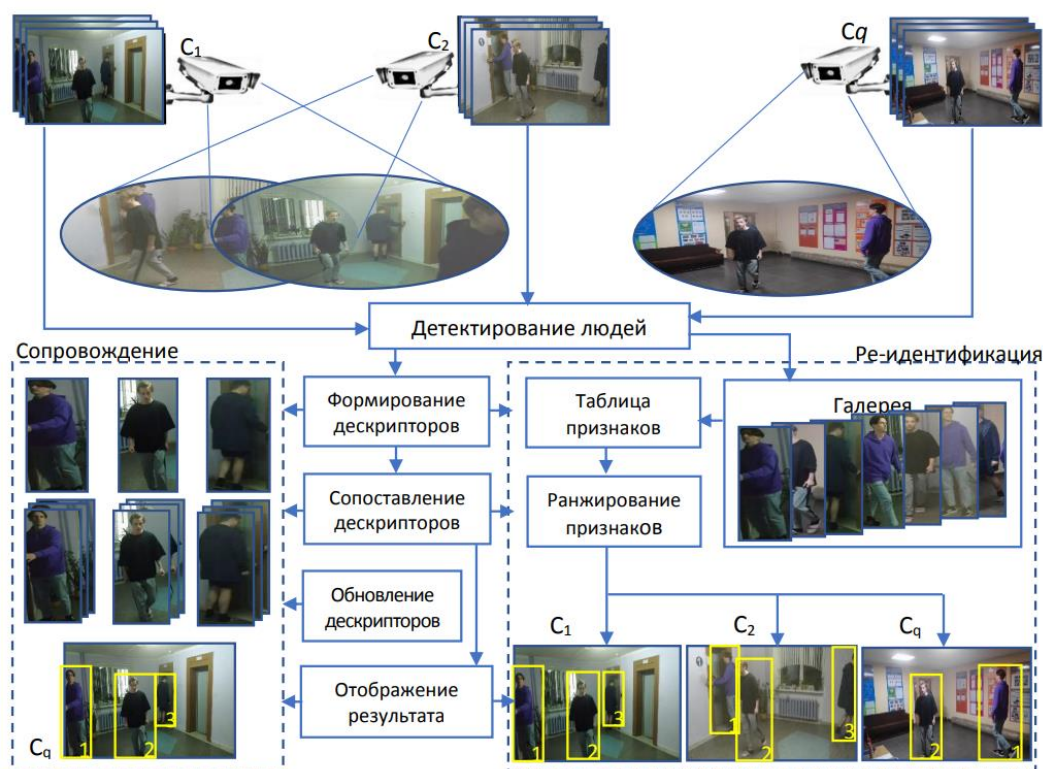


Figure 2 Общая схема интеллектуальной видеосистемы с сопровождением

Имеются разнообразные модели повторной идентификации. Среди них мы выбрали алгоритм FastReID [17], основанный на метрическом обучении в качестве метода повторной идентификации пешеходов исследовательского проекта. Были проведены экспериментальная оценка и его сравнение с другими алгоритмами. Результаты показали, что она сыграла очень хорошую роль в нашем проекте. Ниже мы представим его подробно.

Алгоритм FastReID

Благодаря построению нейронной сети повторной идентификации WRN, DeepSORT способен эффективно решать проблему перекрытия. WRN состоит главным образом из остаточных блоков, имеет много параметров и требует больших вычислительных затрат. Чтобы улучшить скорость и способность идентификации повторной идентификации сети, мы вводим модель повторной идентификации человека FastReID взамен WRN. Это позволяет снизить вычислительную сложность, увеличить экспрессивность и достичь более быстрой скорости и лучшего эффекта идентификации. FastReID в целом состоит из четырех частей: предварительная обработка изображения, основная

сеть, интеграция функций и головные сети. Общая архитектура показана на следующей рисунке:

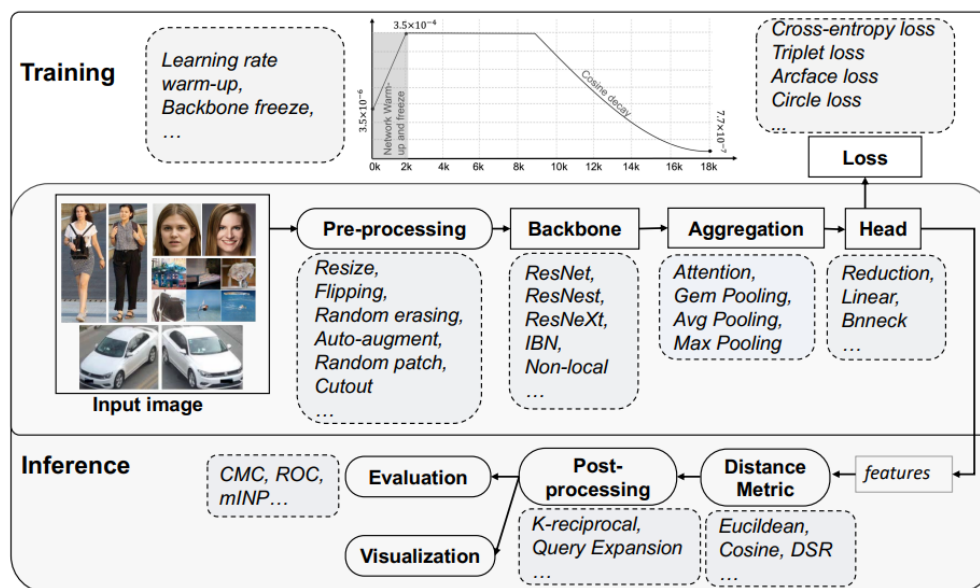


Figure 1. The Pipeline of FastReID library.

При обучении модели FastReID сначала используйте модуль предварительной обработки данных для улучшения и предварительной обработки изображения, затем используйте модуль магистральной сети для извлечения функций реснета и повысьте стабильность модели с помощью модуля нормализации пакетов экземпляров (IBN). Затем используйте модуль интеграции функций, чтобы объединить эти функции в глобальный вектор функций и использовать функции потерь (включая кросс-энтропийную потерю, тройную потерю, потерю дуги и круговую потерю) для обучения классификатора. Классификатор состоит из трех частей: головка BN, линейная головка и головка редукции, а также слои Conv, ReLU, Dropout и Reduction. После обучения модель можно использовать для вывода.

При рассуждении модели FastReID изображение запроса сначала соответствующим образом предварительно обрабатывается, и 512-мерный глобальный вектор признаков генерируется через магистральную сеть, интеграцию признаков и классификатор. Затем модуль метрики расстояния используется для сравнения сходства изображения запроса с изображениями базы данных и возврата набора наиболее похожих результатов. Наконец,

используйте модуль постобработки для обработки результатов, включая такие методы, как k-обратное кодирование и расширение запроса, чтобы повысить точность поиска. Производительность модели обычно оценивается стандартными метриками (такими как CMC, mAP, ROC и minp и т. д.). Для лучшего просмотра результатов поиска FastReID также предоставляет инструменты визуализации для представления результатов поиска.

Набор данных MSMT17

Для выполнения процедуры обучения нейронной сети возникла необходимость найти такой набор данных, который бы покрывал различные условия съёмки и содержал большое количество кадров с людьми.

На конференции CVPR2018 был предложен **новый крупномасштабный набор данных MSMT17 [18]**, который ближе к реальной сцене, а именно Multi-Scene Multi-Time, охватывающий несколько сцен и несколько периодов времени. Набор данных MSMT17 использует сеть из 15 камер видеонаблюдения в кампусе, включая 12 наружных камер и 3 внутренние камеры. Для сбора необработанного видео наблюдения было выбрано 4 дня в месяц с разными погодными условиями. Каждый день собирается 3 часа видео, охватывающих три временных периода: утро, полдень и день. Таким образом, общая продолжительность необработанного видео составляет 180 часов.

Три комментатора-человека потратили два месяца на просмотр обнаруженных ограничивающих прямоугольников и пометку пешеходов. В конце, получается 126 441 ограничительная рамка 4 101 пешехода.

Протокол оценки

Набор данных случайным образом делится в соотношении обучение-тестирование 1:3, а не поровну, как другие наборы данных. Целью этого является поощрение эффективных стратегий обучения, поскольку маркировка данных очень дорога в реальных приложениях.

В конце, обучающий набор содержит 1041 пешехода с 32621 ограничивающей рамкой, а тестовый набор содержит 3060 пешеходов с 93820

ограничивающими рамками. Для тестового набора 11659 ограничивающих рамок выбираются случайным образом в качестве запроса, а остальные 82161 ограничивающие рамки используются в качестве галереи.

Показатели теста - кривая СМС и mAP. Для каждого запроса может быть несколько положительных совпадений.

Метрики оценки эффективности модели

В исследованиях повторной идентификации пешеходов для оценки эффективности модели в основном используются два показателя оценки: точность k-ой совпадения Rank-k и средняя средняя точность (Mean Average Precision, mAP).

Точность k-ой совпадения (Ранг-k)

Ранг-k конкретно относится к вероятности нахождения одного и того же изображения пешехода на k изображениях, наиболее похожих на изображение, которое необходимо получить *probe* в галерее G. Вычисление Ранг-k выглядит следующим образом:

$$Rank(k) = \frac{1}{N} \sum_{i=1}^N \begin{cases} 1, & r_i \leq k \\ 0, & r_i > k \end{cases}$$

Где $r_i (i = 1, 2, \dots, N)$ указывает индекс в отсортированном списке изображения с правильным совпадением идентификатора; N указывает количество изображений в галерее.

Вообще говоря, Ранг-1 является наиболее важным показателем для оценки эффективности модели повторной идентификации человека, который может быть выражен как вероятность того, что первое изображение в полученном отсортированном списке является тем же пешеходом, что и изображение, которое нужно получить. В реальных ситуациях, помимо использования ранга-1, другие часто используемые ранг-k включают ранг-5, ранг-10, ранг-20 и т. д.

Средняя средняя точность (mAP)

Изображение *probe* обычно можно найти несколько совпадающих изображений с одинаковым идентификатором в библиотеке изображений-кандидатов G , и трудно оценить влияние труднодоступных образцов на производительность модели, используя только оценки Rank- k . Следовательно, для более полной оценки эффективности модели повторной идентификации человека к индексу оценки модели повторной идентификации человека добавляется средняя точность. Средняя точность — это индекс, который может оценить результаты ранжирования всех положительных образцов. Только когда все изображения извлеченного человека ранжируются вверху в библиотеке кандидатов, индекс mAP будет высоким, поэтому он может более полно отражать пешеходов. При расчете mAP сначала вычисляется средняя точность (Average Precision, AP), соответствующая каждому извлекаемому изображению *probe*, которая используется для измерения точности распознавания модели на одном образце запроса. Процесс расчета показан в формуле:

$$AP(q) = \frac{\sum_{k \in \{k_1, k_2, \dots, k_S\}} \frac{k_r}{k}}{S}$$

Где S указывает количество положительных образцов, соответствующих полученному изображению *probe* в библиотеке кандидатов G , $\{k_1, k_2, \dots, k_S\}$ — позиции индекса S положительных образцов в результатах сортировки, а k_r указывает количество положительных образцов в первых k результатах. В конце, после вычисления средней точности всех изображений пешеходов в библиотеке Q , содержащей m изображений, можно использовать среднее значение значений AP всех выборок для получения mAP. Процесс расчета показан в формуле:

$$mAP = \frac{\sum_{q_i \in Q} AP(q_i)}{m}$$

Подсчета количества людей в видеопоследовательности

Под визуальным подсчетом людей понимается возможность оценить количество людей в видео наблюдения с помощью определенной обработки. Визуальная статистика потоков людей широко используется во многих областях, в то же время широкое внедрение камер наблюдения и постоянное расширение сценариев применения статистики потоков людей привлекли большое внимание исследователей к проблеме статистики потоков людей.

Многие компании в мире провели множество исследований и экспериментов с прикладными продуктами для подсчета людей, например, серия продуктов для подсчета людей WiseCount от компании Fei Ruisi имеет статистическую точность более 95% и может работать 24 часа. В практических приложениях он имеет высокую точность в реальном времени и статистическую точность.

Что касается статистики трафика покупателей, мы изучили алгоритм, основанный на сверточной нейронной сети, который может регистрировать конкретное время входа и выхода каждого покупателя из магазина для принятия бизнес-решений и анализа.

ГЛАВА 2. ОБУЧЕНИЕ И УСТРАИВАНИЕ МОДЕЛЕЙ

Обучение модели FastReID

Принцип обучения

Принцип обучения модели FastReID заключается в следующем:

1. Предварительная обработка данных: сначала изображения в наборе данных предварительно обрабатываются с вероятностью 0,5 обрезки, масштабирования, симметрии и улучшения данных, чтобы повысить устойчивость модели к изображениям людей в разных позах и условиях освещения.

2.Извлечение признаков: FastReID использует обученную сверточную нейронную сеть серии resnet в качестве экстрактора признаков для извлечения 512-мерного вектора признаков из входного изображения. В то же время FastReID также вводит некоторые новые методы извлечения признаков, такие как выравнивание с учетом масштаба (SAA) и случайное стирание, для дальнейшего улучшения способности выражения признаков.

3.Метричное обучение: вектор признаков сопоставляется с низкоразмерным пространством посредством метрического обучения, а функции потери триплетов и перекрестной энтропийной потери используются для оптимизации модели, так что образцы с одной и той же идентичностью находятся ближе в пространстве признаков. , и разные идентичности Образцы находятся дальше друг от друга в пространстве признаков.

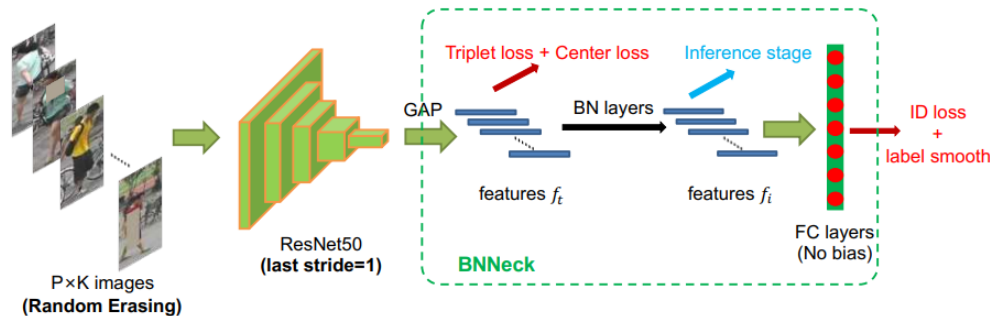
4.Оптимизация модели: использовать оптимизатор Adam для оптимизации параметров модели, чтобы повысить эффективность обучения и точность модели.

Построить модель

Чтобы ускорить сходимость моделей и повысить производительность модели, мы используем ResNet50, модель сверточной нейронной сети, предварительно обученную на наборе данных imagenet. Imagenet — это широко используемый набор данных для классификации изображений, который содержит 1,4 миллиона изображений в более чем 1000 категориях. Модель ResNet50 использует глубокую остаточную сетевую структуру, которая может эффективно справляться с крупномасштабными задачами классификации изображений. После завершения обучения веса модифицированной модели можно использовать в других задачах компьютерного зрения для повышения производительности и эффективности задачи, например, в нашей повторной идентификации пешеходов.

Размерность выходных признаков предварительно обученной модели ResNet50 составляет 2048. Мы продолжаем вставлять средний слой пула и

линейный классификатор после магистральной сети. Выходная размерность линейного классификатора равна N . N представляет собой идентификационный номер обучающих данных.



Функции потерь

Функция потерь:

Наша модель Ре-идентификация дает два выхода: признак f и предсказанные логиты p . Признак f используются для расчета триплетных потерь, а логиты p используются для расчета энтропии перекрестных потерь.

Cross Entropy Loss

Crossentropyloss — это широко используемая функция потери классификации, которая обычно используется для обучения глубоких нейронных сетей задачам классификации изображений. В частности, для входной выборки x и ее истинной метки y формула расчета перекрестной энтропийной потери выглядит следующим образом:

$$\text{Cross Entropy Loss} = - \sum_{i=1}^C y_i \text{Log}(\hat{y}_i)$$

Где C — количество категорий, y_i представляет вероятность i -й категории (то есть истинной метки), а \hat{y}_i представляет вероятность того, что модель предсказывает, что x принадлежит i -й категории.

Цель crossentropyloss — минимизировать разрыв между предсказанием и реальной меткой, чтобы модель могла более точно предсказывать категорию, к которой относится каждая выборка (каждый человек в обучающей выборке рассматривается как отдельная категория).

Triplet Loss

Triplet Loss — одна из функций потерь, широко используемых в области распознавания лиц, назначение которой — приблизить векторы встраивания изображений (embedding) одного и того же человека и отдалить векторы встраивания изображений разных людей. В частности, Triplet Loss изучит вектор встраивания для каждого образца, что может сделать изображения одного и того же человека ближе, а изображения разных людей — дальше.

Для триплета, включающего три изображения привязки, положительное и отрицательное, формула расчета tripletloss выглядит следующим образом:

$$\text{Triplet Loss} = \text{Max}(0, d(a, p) - d(a, n) + m)$$

Где $d(a, p)$ представляет собой евклидово расстояние между привязкой и положительным значением, $d(a, n)$ представляет собой евклидово расстояние между привязкой и отрицательным значением, а m представляет собой отступ, который является предустановленным гиперпараметром A , обычно положительным числом. Смысл этой формулы таков: если расстояние между текущим якорем и плюсом минус расстояние между якорем и минусом плюс маржа меньше или равно 0, это означает, что текущий вектор встраивания достаточно хорош и не нуждается в оптимизации. ; в противном случае необходимо обновить параметры модели для лучшего встраивания векторов.

Triplet Loss обычно используется в сочетании с Batch Hard Triplet Mining, то есть для обучения в каждой партии выбирается самый сложный триплет. Конкретный метод заключается в выборе изображений пешеходов, очень похожих на якорь, но разных категорий, в качестве отрицательных образцов, и выборе изображений пешеходов, которые сильно отличаются от якоря, но принадлежат к той же категории, что и положительные образцы, чтобы улучшить производительность модели.

Ходы обучения

Шаг 1: Используем ResNet50 (инициализированные веса взяты из предварительно обученной модели imagenet), а затем измените ее

полносвязный слой на N. N представляет собой идентификационный номер обучающих данных.

Шаг 2: Мы случайным образом отбираем P идентификаторов и собираем K изображений для каждого идентификатора, а размер последней партии $B = P * K$. В этой статье мы устанавливаем $P = 16$, $K = 4$.

Шаг 3: Мы изменили размер каждого изображения на 256×128 и заполнили 10 пикселей значениями 0, а затем повторно обрезали изображение размером 256×128 , используя случайное кадрирование.

Шаг 4: Каждое изображение случайным образом переворачивается по горизонтали с вероятностью 0,5.

Шаг 5: Каждое изображение декодируется как 32-битное необработанное значение пикселя с плавающей запятой в $[0,1]$, а затем мы нормализуем каналы RGB, вычитая 0,485, 0,456, 0,406 и разделяя на 0,229, 0,224, 0,225.

Шаг 6: Предсказать логиты r в соответствии с функциями $\text{reid } f$, выводимыми моделью.

Шаг 7: Функции $\text{reid } f$ используются для расчета триплетных потерь, логиты r используются для расчета энтропии перекрестных потерь, а запас m триплетных потерь устанавливается равным 0,3.

Шаг 8: Для оптимизации модели используется метод Адама. Начальная скорость обучения установлена на 0,00035 и уменьшается на 0,00010 в 40-ю и 70-ю эпохи соответственно. Всего 120 эпох.

```
Terminal - local - Command Prompt
[05/08 03:15:26 FastReid.Utils.events]: eta: 12:07:38 epoch/iter: 21/9999 total_loss: 1.474 loss_cls: 1.344 loss_triplet: 0.1238 time: 0.9364 data_time: 0.0008 lr: 3.50e-04 max_mem: 2539M
[05/08 03:18:33 FastReid.Utils.events]: eta: 12:04:28 epoch/iter: 21/10199 total_loss: 1.474 loss_cls: 1.337 loss_triplet: 0.122 time: 0.9364 data_time: 0.0012 lr: 3.50e-04 max_mem: 2539M
[05/08 03:21:29 FastReid.Utils.events]: eta: 12:01:20 epoch/iter: 21/10399 total_loss: 1.462 loss_cls: 1.331 loss_triplet: 0.1096 time: 0.9362 data_time: 0.0009 lr: 3.50e-04 max_mem: 2539M
[05/08 03:24:06 FastReid.Utils.events]: eta: 12:01:04 epoch/iter: 22/10399 total_loss: 1.461 loss_cls: 1.329 loss_triplet: 0.1125 time: 0.9363 data_time: 0.0009 lr: 3.50e-04 max_mem: 2539M
[05/08 03:26:47 FastReid.Utils.events]: eta: 11:57:53 epoch/iter: 22/10599 total_loss: 1.453 loss_cls: 1.334 loss_triplet: 0.1104 time: 0.9363 data_time: 0.0013 lr: 3.50e-04 max_mem: 2539M
[05/08 03:27:54 FastReid.Utils.events]: eta: 11:54:37 epoch/iter: 22/10799 total_loss: 1.449 loss_cls: 1.341 loss_triplet: 0.1108 time: 0.9363 data_time: 0.0010 lr: 3.50e-04 max_mem: 2539M
[05/08 03:28:46 FastReid.Utils.events]: eta: 11:53:41 epoch/iter: 22/10859 total_loss: 1.445 loss_cls: 1.332 loss_triplet: 0.1144 time: 0.9363 data_time: 0.0008 lr: 3.50e-04 max_mem: 2539M
[05/08 03:31:13 FastReid.Utils.events]: eta: 11:51:16 epoch/iter: 22/10999 total_loss: 1.453 loss_cls: 1.33 loss_triplet: 0.1201 time: 0.9362 data_time: 0.0013 lr: 3.50e-04 max_mem: 2539M
[05/08 03:34:08 FastReid.Utils.events]: eta: 11:47:51 epoch/iter: 23/11199 total_loss: 1.448 loss_cls: 1.34 loss_triplet: 0.1069 time: 0.9362 data_time: 0.0010 lr: 3.50e-04 max_mem: 2539M
[05/08 03:36:07 FastReid.Utils.events]: eta: 11:45:45 epoch/iter: 23/11327 total_loss: 1.432 loss_cls: 1.318 loss_triplet: 0.115 time: 0.9362 data_time: 0.0008 lr: 3.50e-04 max_mem: 2539M
[05/08 03:37:15 FastReid.Utils.events]: eta: 11:44:32 epoch/iter: 24/11399 total_loss: 1.434 loss_cls: 1.316 loss_triplet: 0.1223 time: 0.9362 data_time: 0.0014 lr: 3.50e-04 max_mem: 2539M
[05/08 03:40:11 FastReid.Utils.events]: eta: 11:41:16 epoch/iter: 24/11599 total_loss: 1.453 loss_cls: 1.342 loss_triplet: 0.1033 time: 0.9361 data_time: 0.0010 lr: 3.50e-04 max_mem: 2539M
[05/08 03:43:19 FastReid.Utils.events]: eta: 11:37:56 epoch/iter: 24/11799 total_loss: 1.433 loss_cls: 1.299 loss_triplet: 0.1203 time: 0.9361 data_time: 0.0018 lr: 3.50e-04 max_mem: 2539M
[05/08 03:43:28 FastReid.Utils.events]: eta: 11:37:56 epoch/iter: 24/11799 total_loss: 1.433 loss_cls: 1.299 loss_triplet: 0.1203 time: 0.9361 data_time: 0.0018 lr: 3.50e-04 max_mem: 2539M
[05/08 03:46:19 FastReid.Utils.events]: eta: 11:34:54 epoch/iter: 25/11999 total_loss: 1.462 loss_cls: 1.352 loss_triplet: 0.1079 time: 0.9361 data_time: 0.0012 lr: 3.50e-04 max_mem: 2539M
[05/08 03:49:42 FastReid.Utils.events]: eta: 11:31:56 epoch/iter: 25/12199 total_loss: 1.434 loss_cls: 1.299 loss_triplet: 0.1105 time: 0.9360 data_time: 0.0008 lr: 3.50e-04 max_mem: 2539M
[05/08 03:50:49 FastReid.Utils.events]: eta: 11:30:28 epoch/iter: 25/12271 total_loss: 1.438 loss_cls: 1.298 loss_triplet: 0.1117 time: 0.9360 data_time: 0.0014 lr: 3.50e-04 max_mem: 2539M
[05/08 03:52:49 FastReid.Utils.events]: eta: 11:28:20 epoch/iter: 26/12399 total_loss: 1.454 loss_cls: 1.341 loss_triplet: 0.10995 time: 0.9360 data_time: 0.0010 lr: 3.50e-04 max_mem: 2539M
[05/08 03:55:15 FastReid.Utils.events]: eta: 11:25:01 epoch/iter: 26/12599 total_loss: 1.42 loss_cls: 1.311 loss_triplet: 0.1064 time: 0.9359 data_time: 0.0009 lr: 3.50e-04 max_mem: 2539M
[05/08 03:58:09 FastReid.Utils.events]: eta: 11:22:45 epoch/iter: 26/12743 total_loss: 1.424 loss_cls: 1.313 loss_triplet: 0.1173 time: 0.9359 data_time: 0.0014 lr: 3.50e-04 max_mem: 2539M
[05/08 03:59:02 FastReid.Utils.events]: eta: 11:21:51 epoch/iter: 27/12799 total_loss: 1.434 loss_cls: 1.331 loss_triplet: 0.1093 time: 0.9359 data_time: 0.0012 lr: 3.50e-04 max_mem: 2539M
[05/08 04:02:08 FastReid.Utils.events]: eta: 11:18:22 epoch/iter: 27/12999 total_loss: 1.438 loss_cls: 1.33 loss_triplet: 0.1054 time: 0.9358 data_time: 0.0009 lr: 3.50e-04 max_mem: 2539M
[05/08 04:05:14 FastReid.Utils.events]: eta: 11:15:09 epoch/iter: 27/13199 total_loss: 1.444 loss_cls: 1.317 loss_triplet: 0.1131 time: 0.9358 data_time: 0.0014 lr: 3.50e-04 max_mem: 2539M
[05/08 04:05:19 FastReid.Utils.events]: eta: 11:14:54 epoch/iter: 27/13215 total_loss: 1.446 loss_cls: 1.331 loss_triplet: 0.1153 time: 0.9358 data_time: 0.0012 lr: 3.50e-04 max_mem: 2539M
[05/08 04:08:21 FastReid.Utils.events]: eta: 11:11:45 epoch/iter: 28/13399 total_loss: 1.434 loss_cls: 1.316 loss_triplet: 0.1008 time: 0.9357 data_time: 0.0010 lr: 3.50e-04 max_mem: 2539M
[05/08 04:11:27 FastReid.Utils.events]: eta: 11:08:35 epoch/iter: 28/13599 total_loss: 1.428 loss_cls: 1.307 loss_triplet: 0.112 time: 0.9356 data_time: 0.0013 lr: 3.50e-04 max_mem: 2539M
[05/08 04:12:09 FastReid.Utils.events]: eta: 11:07:06 epoch/iter: 28/13687 total_loss: 1.438 loss_cls: 1.331 loss_triplet: 0.1016 time: 0.9356 data_time: 0.0012 lr: 3.50e-04 max_mem: 2539M
[05/08 04:16:16 FastReid.Utils.events]: eta: 11:05:19 epoch/iter: 29/13799 total_loss: 1.416 loss_cls: 1.319 loss_triplet: 0.09246 time: 0.9356 data_time: 0.0010 lr: 3.50e-04 max_mem: 2539M
[05/08 04:17:46 FastReid.Utils.events]: eta: 11:02:15 epoch/iter: 29/13999 total_loss: 1.407 loss_cls: 1.291 loss_triplet: 0.107 time: 0.9355 data_time: 0.0014 lr: 3.50e-04 max_mem: 2539M
[05/08 04:20:09 FastReid.engine.defaults]: Prepare testing set
[05/08 04:20:10 FastReid.data.datasets.bases]: -> Loaded MSMT17 in csv format:
| hostet | # imgs | # images | # cameras |
+-----+-----+-----+-----+
| 1 | 1 | 1 | 1 |
```

```

[05/08 08:38:22 fastreid.data.datasets.bases]: <> Loaded MSMT17 in csv format:
| dataset | # imgs | # images | # cameras |
|-----|-----|-----|-----|
| query   | 3040   | 11059   | 15       |
| gallery  | 3040   | 82341   | 15       |
[05/08 08:38:22 fastreid.evaluation.evaluator]: Start inference on 9000 images
[05/08 08:38:40 fastreid.evaluation.evaluator]: Inference done 12/733. 0.6099 s / batch. ETA=0:08:04
[05/08 08:39:16 fastreid.evaluation.evaluator]: Inference done 56/733. 0.6716 s / batch. ETA=0:07:35
[05/08 08:39:47 fastreid.evaluation.evaluator]: Inference done 101/733. 0.6745 s / batch. ETA=0:07:07
[05/08 08:40:18 fastreid.evaluation.evaluator]: Inference done 146/733. 0.6764 s / batch. ETA=0:06:37
[05/08 08:40:49 fastreid.evaluation.evaluator]: Inference done 191/733. 0.6799 s / batch. ETA=0:06:06
[05/08 08:41:15 fastreid.evaluation.evaluator]: Inference done 236/733. 0.6797 s / batch. ETA=0:05:36
[05/08 08:41:49 fastreid.evaluation.evaluator]: Inference done 281/733. 0.6797 s / batch. ETA=0:05:06
[05/08 08:42:19 fastreid.evaluation.evaluator]: Inference done 325/733. 0.6766 s / batch. ETA=0:04:36
[05/08 08:42:50 fastreid.evaluation.evaluator]: Inference done 369/733. 0.6783 s / batch. ETA=0:04:07
[05/08 08:43:20 fastreid.evaluation.evaluator]: Inference done 413/733. 0.6782 s / batch. ETA=0:03:37
[05/08 08:43:50 fastreid.evaluation.evaluator]: Inference done 457/733. 0.6795 s / batch. ETA=0:03:08
[05/08 08:44:21 fastreid.evaluation.evaluator]: Inference done 501/733. 0.6808 s / batch. ETA=0:02:38
[05/08 08:44:51 fastreid.evaluation.evaluator]: Inference done 544/733. 0.6820 s / batch. ETA=0:02:09
[05/08 08:45:21 fastreid.evaluation.evaluator]: Inference done 587/733. 0.6836 s / batch. ETA=0:01:40
[05/08 08:45:52 fastreid.evaluation.evaluator]: Inference done 631/733. 0.6841 s / batch. ETA=0:01:10
[05/08 08:46:22 fastreid.evaluation.evaluator]: Inference done 675/733. 0.6845 s / batch. ETA=0:00:39
[05/08 08:46:53 fastreid.evaluation.evaluator]: Inference done 719/733. 0.6851 s / batch. ETA=0:00:09

P GUI 10000 Problems Terminal Python Packages Python Console Services
136371 CPU 117.8 4 spaces fastreid.2 master
[05/08 09:30:18 fastreid.evaluation.evaluator]: Total inference time: 0:08:45.335901 (0.721615 s / batch per device, on 1 devices)
[05/08 09:30:28 fastreid.evaluation.evaluator]: Total inference time: 0:08:45.335901 (0.721615 s / batch per device, on 1 devices)
[05/08 09:30:28 fastreid.evaluation.evaluator]: Total inference pure compute time: 0:08:44 (0.720387 s / batch per device, on 1 devices)
[05/08 09:30:28 fastreid.evaluation.evaluator]: UserWarning: Cython rank evaluation (very fast so highly recommended) is unavailable, now use python evaluation.
WARNING:rank
[05/08 10:10:04 fastreid.engine.defaults]: Evaluation results for MSMT17 in csv format:
[05/08 10:10:04 fastreid.evaluation.testing]: Evaluation results in csv format:
| Dataset | Rank-1 | Rank-5 | Rank-10 | mAP | mIMP | mATC |
|-----|-----|-----|-----|-----|-----|-----|
| MSMT17 | 84.19 | 90.95 | 95.10 | 62.89 | 14.94 | 65.84 |
P GUI 10000 Problems Terminal Python Packages Python Console Services
136371 CPU 117.8 4 spaces fastreid.2 master

```

Результаты обучения

Мы используем 1/3 данных в MSMT17 в качестве образцов для обучения модели FastReID, а затем используем оставшиеся 2/3 данных в качестве проверочного набора для проверки возможностей модели. Результаты экспериментальной оценки приведены в таблице и на рисунке.

| Dataset | Rank-1↑ | Rank-5↑ | Rank-10↑ | mAP↑ |
|---------|--------------|--------------|--------------|--------------|
| MSMT17 | 84.19 | 90.95 | 95.10 | 62.89 |

По результатам экспериментальной оценки видно, что модель FastReID добилась хороших результатов на наборе данных MSMT17. В частности, она достиг 84,19% точности на 1-м ранге, 90,95% и 95,10% на 5-м и 10-м рангах соответственно. При этом средняя точность (mAP) также достигла 62,89%, что означает, что производительность модели на всем наборе данных относительно стабильна.

В целом эти результаты показывают, что модель FastReID хорошо работает с набором данных MSMT17.

Были проведены сравнение с другими алгоритмами на том же наборе:

| Methods | MSMT17 | |
|--------------------------|---------|------|
| | Rank-1↑ | MAP↑ |
| Ianet(TVPR'19) [19] | 75.7 | 45.8 |
| Auto-reid(ICC'V'19) [20] | 78.2 | 52.5 |
| Osnet(ICC'V'19) [21] | 78.7 | 52.9 |

| | | |
|------------------------------|--------------|--------------|
| Abdnet(ICCV'19) [22] | 82.3 | 60.8 |
| Circle Loss(CVPR'20) [23] | 76.9 | 52.1 |
| Ours | 84.19 | 62.89 |

Через сравнение мы знаем, что точность нашей модели намного выше, чем у других моделей, выпущенных в то же время, и показатели точности являются лучшими.

Устраивание модели DeepSORT

Шаг 1. Используем DeepSORT в качестве сети отслеживания, чтобы отслеживать каждого человека и сопоставлять его с предыдущими траекториями.

Шаг 2. Используем YOLOv5s в качестве части обнаружения DeepSORT для обнаружения людей в каждом кадре изображения и вывода кадра обнаружения.

Шаг 3. Используем FastReID в качестве экстрактора признаков для извлечения признаков внешнего вида пешеходов, чтобы DeepSORT мог различать разных людей.

Шаг 4. Определим гиперпараметры модели. Параметры модели улучшенного алгоритма DeepSORT следующие:

```
DEEPSORT:
REID_CKPT: "./fast-reid/checkpoint/model-final.pth"
MAX_DIST: 0.2
MIN_CONFIDENCE: 0.3
NMS_MAX_OVERLAP: 0.5
MAX_IOU_DISTANCE: 0.7
MAX_AGE: 140
N_INIT: 3
NN_BUDGET: 100
```

Мы используем общедоступный набор данных о пешеходах MOT-16 для тестирования модели отслеживания множества объектов (DeepSORT после улучшения модуля повторной идентификации пешеходов). Чтобы лучше подчеркнуть производительность алгоритма в этой статье, для сравнения были проведены два разных эксперимента.

Эксперимент 1: Сравнительное тестирование нашего улучшенного алгоритма DeepSORT (YOLOv5+FastReID+DeepSORT) с оригинальным алгоритмом DeepSORT. Протестировать оба варианта на тестовом видео, чтобы проанализировать плюсы и минусы улучшенного алгоритма DeepSORT.

Эксперимент 2. Анализ надежности улучшенного алгоритма в различных сценариях. Сравнить улучшенный алгоритм с несколькими основными алгоритмами и проанализировать преимущества и недостатки улучшенного алгоритма.

ГЛАВА 3. СИСТЕМА ОЦЕНКИ ТРАФИКА МАГАЗИНА

Цели разработки системы

Оценки трафика посетителей на основе видеонаблюдения является основой для сбора бизнес-информации и выполнения задач интеллектуального управления, а также важным функциональным компонентом современного интеллектуального видеонаблюдения. В этом проекте будут использоваться передовые технологии компьютерного зрения и глубокого обучения, такие как YOLOv5, DeepSORT и FastReID. В частности, система будет получать видеопотоки в магазине из несколько камеров, использовать YOLOv5 для автоматического обнаружения, появляющихся на видео, затем использовать DeepSORT для отслеживания покупателя и, в конце, использовать FastReID для идентификации и анализа поведение покупателя, чтобы реализовать автоматическую статистику и анализ посещаемости магазина.

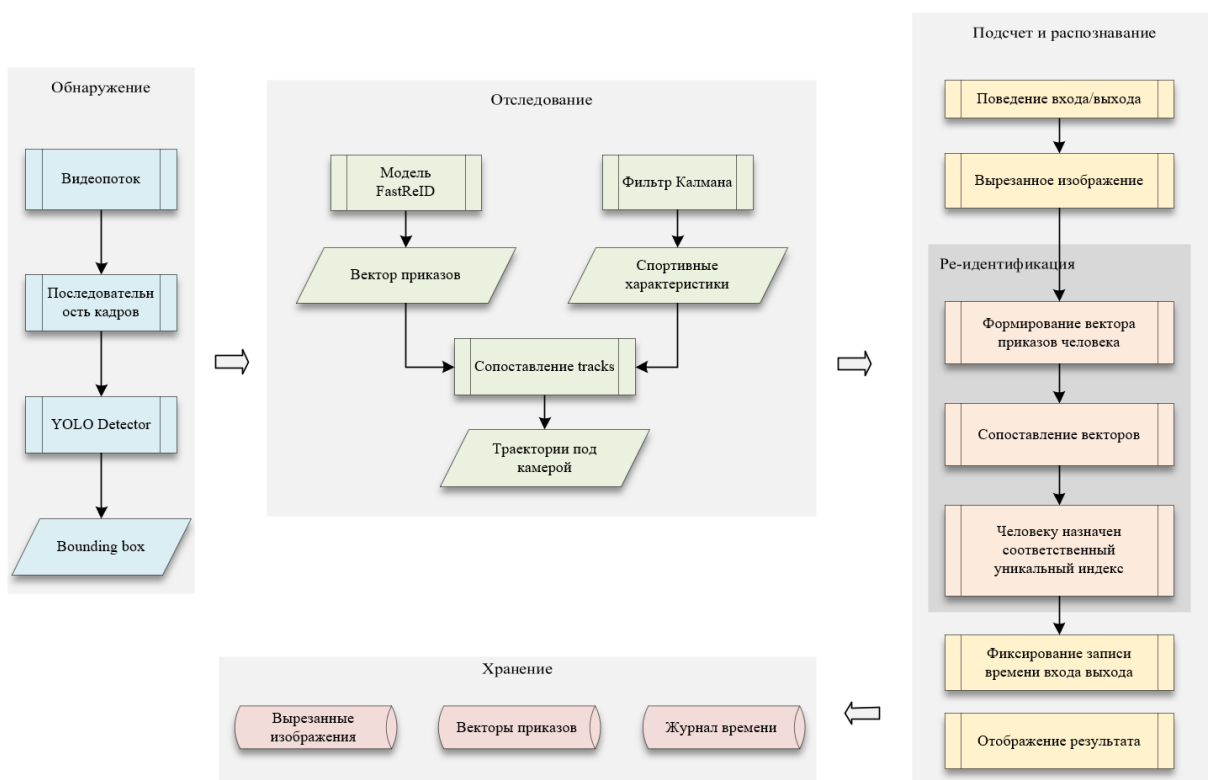
Благодаря исследованиям и практике этого проекта он может помочь продавцам более точно понять привычки и предпочтения поведения посетителей , сформулировать более научные и эффективные стратегии продаж, а также повысить прибыльность и рыночную конкурентоспособность магазинов. В то же время система также может предоставлять продавцам функции мониторинга и прогнозирования потоков посетителей в режиме реального времени, помогая продавцам еще больше оптимизировать

управление операциями, повышать уровень обслуживания и удобство работы пользователей.

Описание системы

Общая архитектура системы

На основе FastReID и DeepSORT разрабатывается система оценки трафика покупателей, ориентированная на небольшие магазины. Система **состоит из четырех модулей**: 1)модуль обнаружения пешеходов, 2)модуль отслеживания пешеходов, 3)модуль подсчета и распознавания пешеходов и 4)модуль хранения. Общая архитектура системы показана на рисунке.



Модуль обнаружения пешеходов

На вход модуля обнаружения пешеходов подается видеопоток с одной камеры, в процессе реализации в данной работе используется алгоритм обнаружения объектов YOLO. В частности, YOLO используется для выделения пешеходов в каждом кадре видео, и пешеходы помечаются

прямоугольным кадром. Конкретное положение пешехода в возвращается картинка, определение которой приведено в формуле:

$$b = [x, y, w, h]$$

Где (x, y) — пиксельные координаты верхнего левого угла кадра обнаружения, а (w, h) — ширина и высота кадра обнаружения.

Модуль отслеживания пешеходами

Модуль отслеживания пешеходов использует модель DeepSORT, и его входными данными является кадр и координаты всех людей. Цель этой модули состоит в том, чтобы связать обнаруженные пешеходы в текущем кадре с существующими траекториями в предыдущих кадрах, впоследствии обновляет их. После получения кадра обнаружения и координатов, предоставленных модулем обнаружения пешеходов, сначала использовать модуль FastReID и алгоритм Калмана, чтобы извлечь признаки внешнего вида и признаки движения пешеходных в текущем кадре, а затем вычислить матрицу расстояний между ними, и, в конце, использовать венгерский алгоритм для сопоставления. Алгоритм связывает обнаруженные пешеходные в текущем кадре с существующими траекториями движения в предыдущем кадре, чтобы построить траектории всех пешеходов в видео.

Модуль подсчета и распознавания пешеходного

Вход данной модуля — траектория движения всех людей под одной камерой, и цель — точно зафиксировать в текущий момент количество людей, входящих и выходящих из магазина, и для передвигавшихся вдоль указанных направлений выполнить Ре-идентификацию, чтобы собирать статистику о времени и количестве уникальных вхождений человека на магазине, покрываемом несколькими камерами видеонаблюдения. В качестве выходных данных вырезаны их изображения, для которых формированы вектора признаков и, зафиксированы в журнал времени входа/выхода.

Подсчет людей

После рассмотрения реалистичных требований к задаче в этом исследовании используется «метод двух линий» для осуществления подсчёта людей. Конкретный принцип двухлинейного метода заключается в следующем:



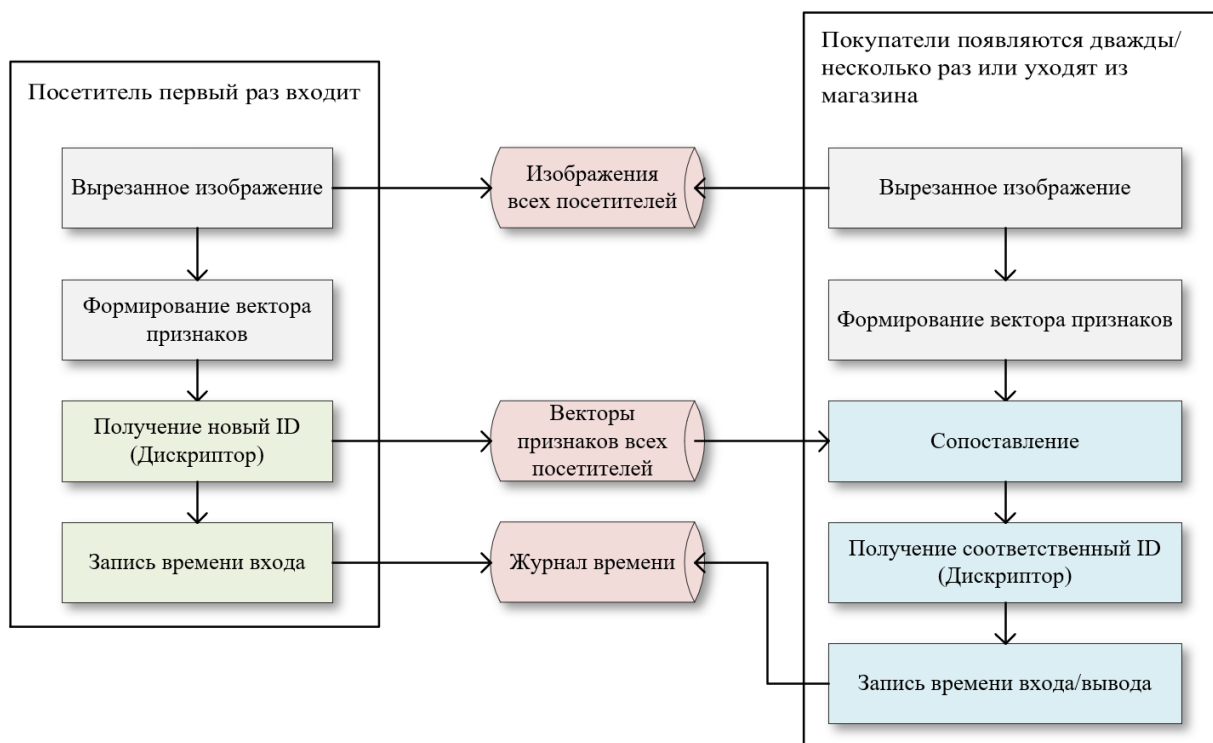
В магазине устанавливается стационарная камера и получается видеопоток наблюдения. Причем в видеопотоке отмечаются две счетные линии, которые записываются как А и В. На основе траектории движения тела на пересечении траектории и линии указывается направления (вход/выход) проходящих людей.

В качестве примера предположим, что магазин находится слева от линии В, как показано в рисунке. Видно, что траектория входа посетителя будет сначала соприкоснуться с линией А, а затем с контактом с В, это называется поведением входа. И наоборот, траектория ухода пешехода должна сначала коснуться линии В, а затем линии А, это называется поведением выхода. Нам нужно только записать, касается ли траектория пешехода сначала линии А или линии

В, а затем мы уже можем определить, входит ли пешеход в магазин или выходит из него. Перебирая весь видеопоток, мы можем получить точное количество людей, входящих и выходящих.

Ре-идентификация

Краткая иллюстрация Ре-идентификации представлена на рисунке.



Когда происходит поведение входа/выхода, будут выделять обнаруженного пешехода, чтобы получить вырезанное изображение и сформировать вектор признаков, а также зафиксировать время в журнал.

Для описания человека при ре-идентификации дескриптор может быть представлен как:

$$P_{ID} = (p^{ID}, f^{gen})$$

Где p^{ID} – идентификатор (метка) человека; f^{gen} – вектор признаков для изображения человека. Создается таблица, содержащая изображения всех посетителей и их дескрипторы, которая называется галереей.

При выполнении запроса происходит вычисление вектора признаков f , который используется для определения расстояния d между данным запросом

и дескрипторами изображений в галерее. Расстояния d затем используются для ранжирования таблицы от d_{min} до d_{max} . Если значение расстояния d превышает пороговое значение, то считается, что запрос не соответствует ни одному дескриптору в галерее, и соответствующий дескриптор добавляется в конец таблицы с новым уникальным идентификатором человека. После исключения всех неподходящих кандидатов из таблицы, для повторной идентификации выбирается человек с наиболее схожим вектором признаков f^{gen} , который находится вверху списка ранжированной таблицы. Это человек, который будет принят за результат повторной идентификации. После этого будет возвращаться уникальный идентификатор человека.

Таким образом, все посетители получают уникальные идентификаторы, и на этой основе фиксируются времени входы/выхода. В качестве выходных данных создается журнал, где записаны ID камер, идентификатор человека и времени появления.

Модуль хранения

Модуль хранения отвечает за регистрацию времени, когда покупатели заходят в магазин, появляются под камерой и выходят из магазина.

Каждый раз, когда прохожий входит в указанные зоны, будет фиксировать время появления человека и в то же время перехватывать изображение соответствующего человека и извлекать вектор признаков на основе изображения. Если камера является камерой в магазине или на выходе, личность пешехода также учитывается в соответствии с моделью повторной идентификации. В конце, личность человека, в каком отделе он появляется, время появления и изображение человека сохраняются на диске для последующего статистического анализа.

Среда разработки

Аппаратные среды, использованные для разработки, показаны в Таблице 1:

Таблица 1

| Предмет | Технические характеристики | Комментарий |
|---------|---|-------------------------|
| Чип | AMD Ryzen 5 5600H with Radeon Graphics 3.30 GHZ | - |
| Памяти | 16.0 GB | ПО использует до 2.1 GB |
| Камер | 1080P | 720P минимум |
| ОС | Windows 10 Professional | - |

Программные среды, использованные в разработке, показаны в Таблице 2:

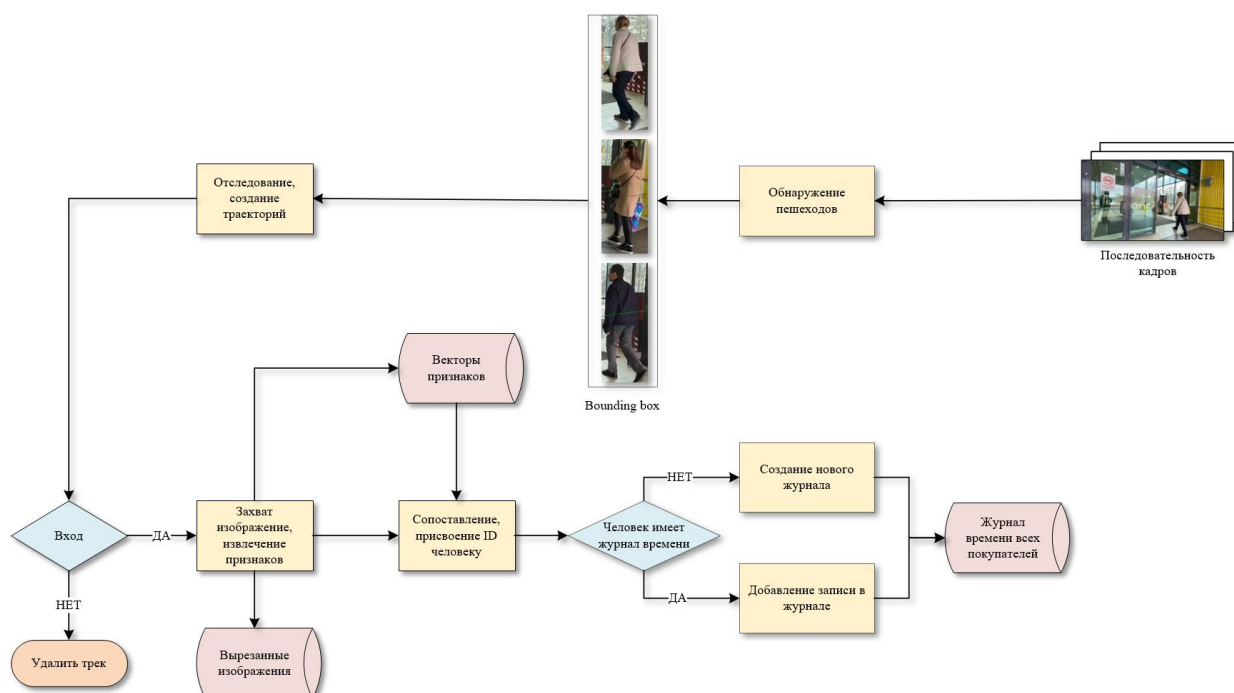
Таблица 2

| Предмет | Версия | Комментарий |
|---------|--------------------------------|-----------------|
| Python | 3.8.16 | Данная или выше |
| Pycharm | 2022.1.2 (Educational Edition) | Данная или выше |
| Opencv | 4.5.3 | Данная или выше |
| Pytorch | 1.13.0 | Данная или выше |

Демонстрация и тестирование системы

Общая принципиальная схема системы

Общая принципиальная схема системы показана ниже на рисунке:



Функционалы системы

Отслеживание людей под одной камерой

Следующие три изображения выбираются соответственно из изображений 300-го, 330-го, 360-го и 390-го кадра под камерой 3, а временной интервал выделения составляет около 1,5 секунды. На 3 рисунках ниже показано отслеживание человека 881.





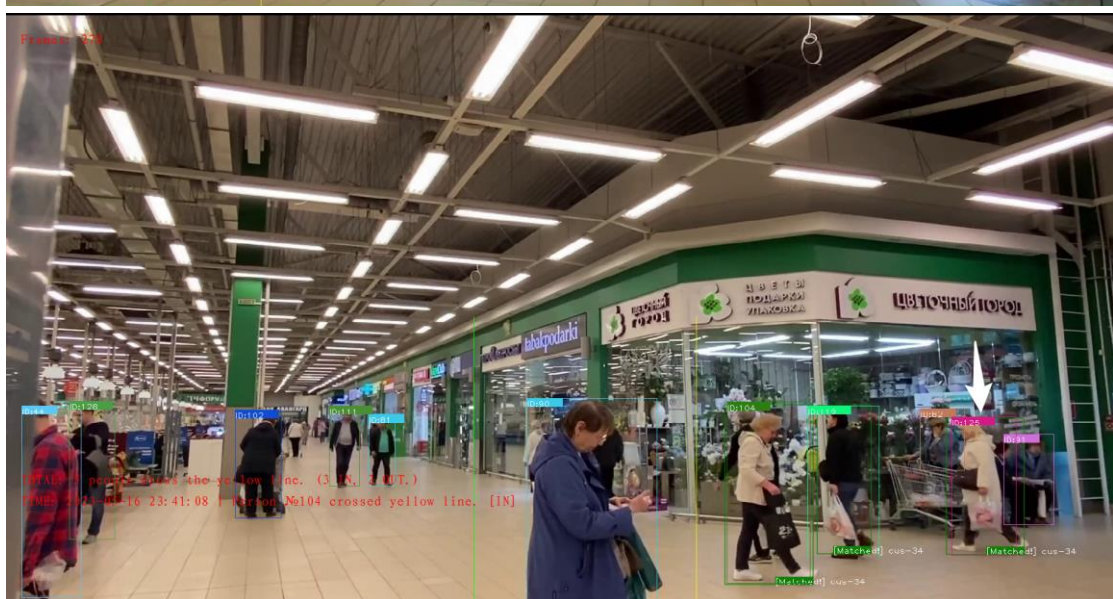


Видно, что отслеживание человека у № 881 хорошее в пределах 90 кадров, потому что идентификатор у него всегда сохраняется без переключения. Даже если перекрывание между людьми возникает в третьем кадре, идентификатор пешехода остается 881 в четвертом кадре, когда окклюзия исчезает в четвертом кадре. Это роль ассоциации признаков внешнего вида, предоставляемая FastReID. Кроме того, другие пешехода, отличные от 881, такие как 803, 819, 869, также хорошо отслеживаются.

Результаты испытаний доказывают, что система может непрерывно отслеживать нескольких пешеходов под одной камерой.

Отслеживание людей под несколькими камерами

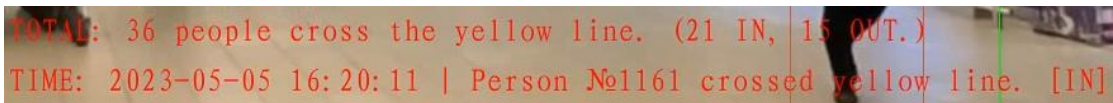
На трех нижеприведенных снимках показаны результаты отслеживания пешехода cus-43, проходящего через три камеры в хронологическом порядке.



Из результатов теста видно, что при первом появлении пешехода в зоне наблюдения камеры 1 система выдает ей идентификатор: sus-43, который является глобально уникальным. Когда пешеход sus-43 исчез из поля зрения камеры 1 и вошел в камеру 2, система извлекла черты его внешности через FastReID для сравнения, узнала идентификационную информацию пешехода sus-43 и восстановила отслеживание и повторно идентифицировала личность. Отображается информация в правом нижнем углу окна обнаружения. Точно так же, когда она вышла из камеры 2 и вошла в камеру 3, система также успешно идентифицировала ее в зоне наблюдения камеры 3.

На кадрах существуют желтая и зеленая виртуальные линии, они не реалины. Эти линии используются только для подсчета качества людей.

Когда пешеходы проходят через линии, в левом нижнем углу экрана печатаются подсказки времени прохождения, личности входящего персонала и общего количества входящих и выходящих.



TOTAL: 36 people cross the yellow line. (21 IN, 15 OUT.)
TIME: 2023-05-05 16:20:11 | Person №1161 crossed yellow line. [IN]

Запись времени входа и выхода покупателей

Мы используем структуру данных в Python для записи входных и выходных записей всех посетителей за определенный период времени. Распечатаем его, как показано ниже на рисунке.

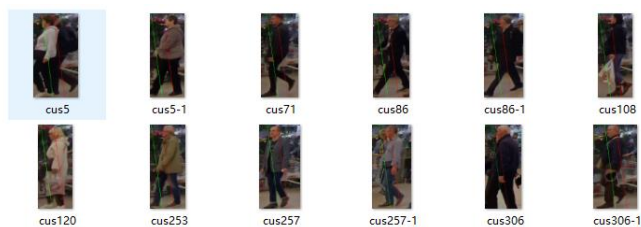
| PERSON-ID | CAMERA-ID | TIME | CAMERA-ID | TIME | CAMERA-ID | TIME |
|-----------|-----------|-----------------------|-----------|-----------------------|-----------|-----------------------|
| cus-1 | cam-1 | 2023-05-10 03: 31: 27 | cam-2 | 2023-05-10 03: 37: 59 | cam-3 | 2023-05-10 03: 46: 04 |
| cus-5 | cam-1 | 2023-05-10 03: 31: 37 | cam-2 | 2023-05-10 03: 50: 03 | cam-3 | |
| cus-35 | cam-1 | 2023-05-10 03: 32: 03 | cam-2 | 2023-05-10 03: 38: 10 | cam-3 | 2023-05-10 03: 46: 19 |
| cus-71 | cam-1 | 2023-05-10 03: 32: 11 | cam-2 | 2023-05-10 03: 38: 15 | cam-3 | 2023-05-10 03: 46: 23 |
| cus-82 | cam-1 | 2023-05-10 03: 32: 37 | cam-2 | 2023-05-10 03: 38: 26 | cam-3 | 2023-05-10 03: 46: 35 |
| cus-94 | cam-1 | 2023-05-10 03: 32: 48 | cam-2 | 2023-05-10 03: 38: 32 | cam-3 | 2023-05-10 03: 46: 42 |
| cus-101 | cam-1 | 2023-05-10 03: 32: 53 | cam-2 | 2023-05-10 03: 41: 28 | cam-3 | |
| cus-103 | cam-1 | 2023-05-10 03: 32: 53 | cam-2 | 2023-05-10 03: 41: 39 | cam-3 | 2023-05-10 03: 50: 03 |
| cus-108 | cam-1 | 2023-05-10 03: 33: 00 | cam-2 | 2023-05-10 03: 41: 02 | cam-3 | 2023-05-10 03: 49: 20 |
| cus-120 | cam-1 | 2023-05-10 03: 33: 10 | cam-2 | 2023-05-10 03: 42: 06 | cam-3 | 2023-05-10 03: 50: 23 |
| cus-134 | cam-1 | 2023-05-10 03: 33: 14 | cam-2 | 2023-05-10 03: 42: 30 | cam-3 | |
| cus-147 | cam-1 | 2023-05-10 03: 33: 21 | cam-2 | 2023-05-10 03: 42: 57 | cam-3 | 2023-05-10 03: 51: 18 |
| cus-203 | cam-1 | 2023-05-10 03: 34: 15 | cam-2 | 2023-05-10 03: 42: 59 | cam-3 | 2023-05-10 03: 51: 16 |
| cus-253 | cam-1 | 2023-05-10 03: 35: 21 | cam-2 | | cam-3 | |
| cus-248 | cam-1 | 2023-05-10 03: 35: 24 | cam-2 | | cam-3 | |
| cus-257 | cam-1 | 2023-05-10 03: 35: 28 | cam-2 | | cam-3 | |
| cus-306 | cam-1 | 2023-05-10 03: 36: 17 | cam-2 | | cam-3 | |
| cus-314 | cam-1 | 2023-05-10 03: 36: 30 | cam-2 | | cam-3 | |
| cus-331 | cam-1 | 2023-05-10 03: 36: 53 | cam-2 | | cam-3 | |
| cus-335 | cam-1 | 2023-05-10 03: 36: 56 | cam-2 | | cam-3 | |

Вырезанные изображения покупателей

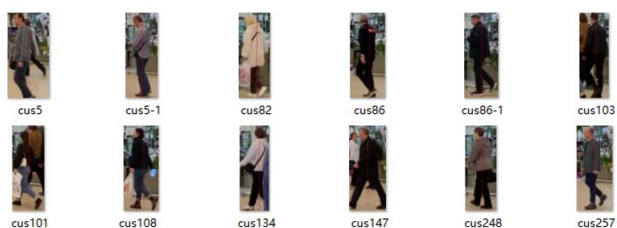
Вырезанные изображения покупателей из 1-ой камеры (частично):



Вырезанные изображения покупателей из 2-ой камеры (частично):



Вырезанные изображения покупателей из 3-ой камеры (частично):



ГЛАВА 4. АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ

Экспериментальная оценка модели FastReID на открытом наборе данных

После завершения обучения модели мы решили протестировать производительность нашей модели на другом наборе данных Market1501[26] и DukeMTMC[27] и сравнили ее с другими моделями ReID, появившимися в тот же период:

| Methods | Market1501 | | DukeMTMC | |
|---------------------------|-------------|-------------|-------------|-------------|
| | Rank-1 | mAP | Rank-1 | mAP |
| IANet(IVPR'19) [19] | 94.4 | 83.1 | 87.1 | 73.4 |
| Auto-ReID(ICCV'19) [20] | 94.5 | 85.1 | - | - |
| OSNet(ICCV'19) [21] | 94.8 | 84.9 | 88.6 | 73.5 |
| ABDNet(ICCV'19) [22] | 95.6 | 88.3 | 89.0 | 78.6 |
| Circle Loss(CVPR'20) [23] | 96.1 | 87.4 | 89.0 | 79.6 |
| ours | 95.7 | 88.4 | 90.1 | 81.3 |

Согласно приведенным выше данным, мы видим, что модель FastReID хорошо работает как с наборами данных Market1501, так и с наборами данных DukeMTMC. Его точность Rank-1 в наборе данных Market1501 составляет 95,7%, mAP — 88,4%, а его точность Rank-1 в наборе данных DukeMTMC — 90,1%, mAP — 81,3%.

По сравнению с другими алгоритмами FastReID показал очень хорошую производительность. Например, в наборе данных Market1501 точность Rank-1 FastReID и mAP превзошли такие алгоритмы, как IANet, Auto-ReID и OSNet соответственно; даже в наборе данных DukeMTMC его точность Rank-1 и mAP превзошли все другие алгоритмы. Это показывает, что FastReID обладает высокой надежностью и способностью к обобщению в задачах повторной идентификации личности.

В заключение, основываясь на этих оценочных метриках, мы можем считать FastReID эффективной моделью повторной идентификации личности, которую предполагается применять в практических сценариях.

Сравнительное тестирование улучшенного алгоритма DeepSORT с исходным алгоритмом DeepSORT

Процедура эксперимента

Мы протестируем все видеопоследовательности в наборе данных MOT16, в которых идентификаторы пешеходов и положения границ отмечены вручную (gt.txt), всего 7. Это: MOT16-02, MOT16-04, MOT16-05, MOT16-09, MOT16-10, MOT16-11, MOT16-13. Здесь в качестве примера взят MOT16-13, а операции для других видео аналогичны.

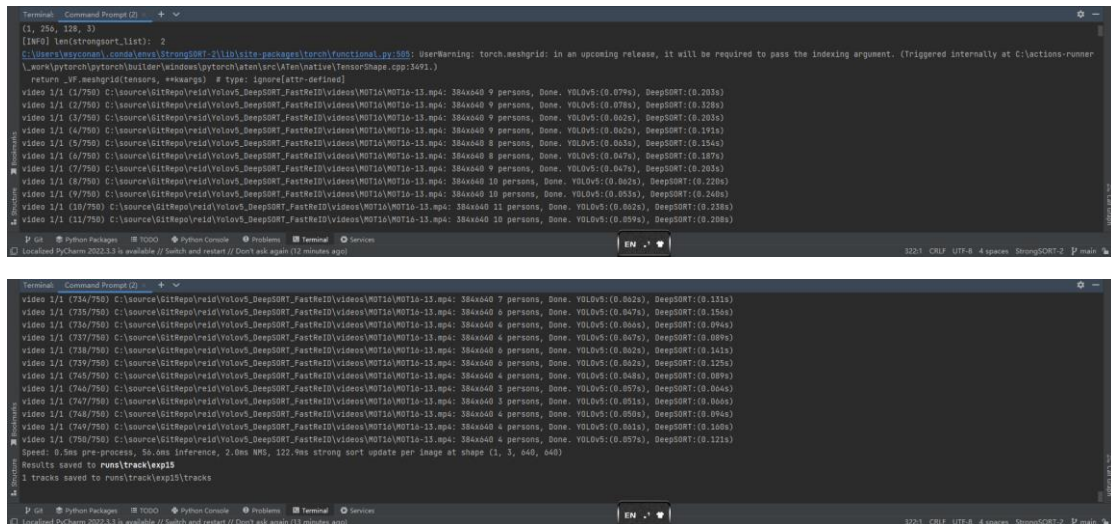
1. Загрузить набор данных MOT16, получить видеофрагменты и соответствующие им файлы gt.txt. Файл gt.txt в MOT16-13 выглядит следующим образом (частично), а параметры разделены запятыми:

```
1,1,1376,485,37,28,0,11,1
2,1,1379,486,37,28,0,11,1
3,1,1382,487,38,29,0,11,1
4,1,1386,488,38,29,0,11,1
5,1,1389,490,38,29,0,11,1
```

Файл gt.txt представляет собой текстовый файл CSV, каждая строка содержит объект, описывающий отслеживаемый объект в одном из фреймов, с 9 значениями, разделенными запятыми. Trackeval использует только первые 6, номер кадра, идентификатор объекта и 4 координаты кадра отслеживания, а последние 3 (надежность классификации, категория объекта, видимость) не участвуют в расчетах и могут быть проигнорированы следующим образом:

```
<frame>, <id>, <bb_left>, <bb_top>, <bb_width>, <bb_height>,  
<conf>, <class>, <visibility>
```

2. Запустить программу track.py в созданной нами модели DeepSORT.



```
Terminal Command Prompt (2)
(1, 256, 128, 3)
[INFO] testStrongSORT.List: 0
C:\Users\user\anaconda\envs\StrongSORT-1110\site-packages\torch\functional.py:589: UserWarning: torch.meshgrid: in an upcoming release, it will be required to pass the indexing argument. (Triggered internally at C:\actions-runner\_work\pytorch\pytorch\builder\windows\pytorch\aten\src\ATen\native\TensorShape.cpp:3691)
return _VF.meshgrid(tensors, **kwargs) # type: ignore[attr-defined]
video 1/1 (1/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 9 persons, Done. VOLov5:(0.079%), DeepSORT:(0.203%)
video 1/1 (2/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 9 persons, Done. VOLov5:(0.078%), DeepSORT:(0.226%)
video 1/1 (3/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 9 persons, Done. VOLov5:(0.062%), DeepSORT:(0.203%)
video 1/1 (4/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 9 persons, Done. VOLov5:(0.062%), DeepSORT:(0.191%)
video 1/1 (5/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 8 persons, Done. VOLov5:(0.063%), DeepSORT:(0.154%)
video 1/1 (6/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 8 persons, Done. VOLov5:(0.047%), DeepSORT:(0.187%)
video 1/1 (7/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 9 persons, Done. VOLov5:(0.047%), DeepSORT:(0.203%)
video 1/1 (8/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 10 persons, Done. VOLov5:(0.062%), DeepSORT:(0.226%)
video 1/1 (9/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 10 persons, Done. VOLov5:(0.051%), DeepSORT:(0.246%)
video 1/1 (10/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 11 persons, Done. VOLov5:(0.062%), DeepSORT:(0.226%)
video 1/1 (11/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 10 persons, Done. VOLov5:(0.059%), DeepSORT:(0.208%)

video 1/1 (724/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 7 persons, Done. VOLov5:(0.062%), DeepSORT:(0.111%)
video 1/1 (725/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 8 persons, Done. VOLov5:(0.047%), DeepSORT:(0.166%)
video 1/1 (726/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 4 persons, Done. VOLov5:(0.066%), DeepSORT:(0.096%)
video 1/1 (727/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 4 persons, Done. VOLov5:(0.047%), DeepSORT:(0.089%)
video 1/1 (728/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 6 persons, Done. VOLov5:(0.062%), DeepSORT:(0.141%)
video 1/1 (729/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 6 persons, Done. VOLov5:(0.062%), DeepSORT:(0.125%)
video 1/1 (730/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 4 persons, Done. VOLov5:(0.066%), DeepSORT:(0.089%)
video 1/1 (731/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 3 persons, Done. VOLov5:(0.057%), DeepSORT:(0.066%)
video 1/1 (732/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 3 persons, Done. VOLov5:(0.051%), DeepSORT:(0.066%)
video 1/1 (733/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 4 persons, Done. VOLov5:(0.050%), DeepSORT:(0.094%)
video 1/1 (734/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 4 persons, Done. VOLov5:(0.061%), DeepSORT:(0.106%)
video 1/1 (735/750) C:\source\GitRepo\reid\Volov5_DeepSORT_FastReID\videos\MOT16\MOT16-13.mp4: 384x640 4 persons, Done. VOLov5:(0.074%), DeepSORT:(0.121%)
Speed: 0.5ms pre-process, 56.0ms inference, 2.0ms NMS, 122.7ms strong sort update per image (1, 3, 640, 640)
Results saved to runs\track\exp15
1 tracks saved to runs\track\exp15\tracks
```

Получить файл данных отслеживания MOT16-13.txt, совместимый с форматом MOT16. Файл MOT16-13.txt:

```
3 7 726 247 12 40 -1 -1 -1 0
3 8 269 276 16 45 -1 -1 -1 0
3 9 744 256 13 37 -1 -1 -1 0
4 1 1 341 13 93 -1 -1 -1 0
4 2 773 279 22 57 -1 -1 -1 0
```

Формат MOT16-13.txt немного отличается от gt.txt, заявляя, что он «совместим» с форматом MOT16. Формат «совместимость» имеет всего 10 значений, а его параметры разделены пробелами. Первые 6 параметров, участвующих в расчете MOT16, такие же, как и gt.txt, а последние 4 параметра в расчете индикатора не участвуют, все они равны -1. Формат следующий:

<frame>, <id>, <bb_left>, <bb_top>, <bb_width>, <bb_height>,
<conf>, <x>, <y>, <z>

3. Запускаем программу оценки и получаем результат:

| | | | | | | | | | | | | | | |
|---|--------|---------|--------|--------|--------|--------|--------|--------|--------|---------|---------|------------|------|----|
| All sequences for deepsort finished in 0.91 seconds | | | | | | | | | | | | | | |
| HOTA: deepsort-pedestrian | HOTA | DetA | AssA | DetRe | DetPr | AssRe | AssPr | LocA | OWTA | HOTA(0) | LocA(0) | HOTALoc(0) | | |
| MOT16-13 | 57.313 | 59.474 | 56.029 | 66.611 | 73.942 | 63.201 | 75.428 | 81.584 | 60.957 | 74.316 | 76.452 | 56.816 | | |
| COMBINED | 57.313 | 59.474 | 56.029 | 66.611 | 73.942 | 63.201 | 75.428 | 81.584 | 60.957 | 74.316 | 76.452 | 56.816 | | |
| CLEAR: deepsort-pedestrian | MOTA | MOTP | MODA | CLR_Re | CLR_Pr | MTR | PTR | MLR | sMOTA | CLR_TP | CLR_FN | CLR_FP | IDSW | MT |
| MOT16-13 | 72.154 | 72.338 | 72.338 | 81.212 | 90.149 | 50 | 31.25 | 10.75 | 54.544 | 2654 | 614 | 290 | 6 | 8 |
| COMBINED | 72.154 | 72.338 | 72.338 | 81.212 | 90.149 | 50 | 31.25 | 10.75 | 54.544 | 2654 | 614 | 290 | 6 | 8 |
| Count: deepsort-pedestrian | Dets | GT Dets | IDs | GT IDs | | | | | | | | | | |
| MOT16-13 | 2044 | 3268 | 18 | 16 | | | | | | | | | | |
| COMBINED | 2044 | 3268 | 18 | 16 | | | | | | | | | | |

4. Выполнить ту же операцию для MOT16-02, MOT16-04, MOT16-05, MOT16-09, MOT16-10, MOT16-11 соответственно и просмотрите весь набор данных MOT16. Получить окончательный результат на этом наборе данных MOT16.

Результаты эксперимента

Результат сравнения улучшенного алгоритма DeepSORT с исходным показан на таблице:

| Algorithm | MOTA ↑ | MOTP ↑ | MT / % ↑ | ML / % ↓ | Ids / % ↓ |
|-------------------|-------------|-------------|-------------|-------------|------------|
| Original DeepSORT | 61.4 | 79.1 | 32.8 | 18.2 | 781 |
| Ours | 66.2 | 80.8 | 35.3 | 17.6 | 760 |

По результатам видно, что после улучшения алгоритма DeepSORT все показатели немного улучшаются. Точность сопровождения (MOTA) увеличилась на 5,2%, точность сопровождения (MOTP) увеличилась на 1,7%, количество правильно сопровождаемых траекторий целей более 80% (MT) увеличилось на 3,5%, а количество правильно сопровождаемых траекторий целей ниже 20%. (ML) сократилось на 0,6%, а общее количество целевых переключателей идентификации (ID) сократилось в 21 раз. Это значит, что улучшенный DeepSORT может уменьшить количество пешеходных переключений.

На основании экспериментальной оценки можно утверждать, что внедрение FastReID и YOLOv5 действительно повышает точность отслеживания алгоритма DeepSORT.

Сравнение улучшенного алгоритма DeepSORT с несколькими основными алгоритмами

Результаты эксперимента 2 представлены в таблице.

| Algorithm | MOTA ↑ | MOTP ↑ | MT / % ↑ | ML / % ↓ | Ids / % ↓ | FPS / Hz ↑ |
|---------------------------|--------|--------|----------|----------|-----------|------------|
| SORT [24] | 59.8 | 79.6 | 25.4 | 22.7 | 1423 | 8.6 |

| | | | | | | |
|------------------------|------|------|------|------|------|------|
| Original DeepSORT [18] | 61.4 | 79.1 | 32.8 | 18.2 | 781 | 6.4 |
| JDE [25] | 64.4 | - | 35.4 | 20 | 1544 | 18.5 |
| Ours | 66.2 | 80.8 | 35.3 | 17.6 | 760 | 9.8 |

Мы усовершенствовали алгоритм повторной идентификации пешеходов (FastReID) на основе DeepSORT, благодаря чему была повышена точность, точность и возможность сохранения идентификатора пешехода модели. По сравнению с другими моделями наша модель имеет комплексные преимущества: точность сопровождения (MOTA) и точность сопровождения (MOTP) самые высокие, а количество правильно сопровождаемых траекторий целей более чем на 80% (MT) практически равно JDE и делят первое место. Количество правильно отслеженных траекторий целей ниже 20% (ML) и общее количество переключений идентификации целей (ID) являются самыми низкими.

Кроме того, замечено, что производительность в реальном времени немного хуже, чем у алгоритма JDE, потому что JDE является одноэтапным алгоритмом отслеживания, а отслеживание в реальном времени относительно высоко, но переключение идентификаторов относительно частые, что обусловлено взаимным перекрытием мишеней. В розничной среде часто бывают сцены с большим количеством плотных пешеходов, поэтому наш алгоритм является подходящим выбором в это время.

На основании экспериментальной оценки можно утверждать, что данный алгоритм превосходит аналоги по точности сопровождения, количеству правильно сопровождаемых траекторий и стабильности.

Экспериментальная оценка улучшенной модели DeepSORT на открытом наборе данных

Мы провели сравнительный эксперимент на более сложном новом наборе данных MOT17.



Мы протестировали алгоритм многоцелевого отслеживания на четырех подмножествах набора данных MOT17 и сравнили результаты с результатами в конкурсе MOT Challenge. В сравнении [использовались 4 видеопоследовательностей, для которых в \[28\] опубликованы результаты.](#)

Измерение проводилось с использованием 5 показателей точности – MOTA, IDF1, MT, ML и IDs. Среди них MOTA является основной оценочной метрикой для отслеживания нескольких объектов, которая сочетает в себе точность и полноту обнаружения, сопоставления и отслеживания. IDF1 - показатель, уделяющий больше внимания точности следящего устройства. MT и ML соответственно представляют собой долю количества правильно сопровождаемых целей и долю количества пропущенных сопровождаемых целей в общем количестве обработанных кадров.

Результат показан на таблице:

| Traker | MOTA ↑ | IDF1↑ | MT/% ↑ | ML/% ↓ | IDs ↓ |
|------------|-----------|-------------|-----------|-----------|-------------|
| MFI | 60.1 | 58.8 | 26.0 | 29.7 | 2065 |
| ISE_MOT17R | 60.1 | 56.4 | 28.5 | 28.1 | 2556 |
| SLA | 59.7 | 63.4 | 24.0 | 31.1 | 1647 |
| LPC_MOT | 59.0 | 66.8 | 29.9 | 33.9 | 1122 |

| | | | | | |
|-------------|-------------|-------------|-------------|-------------|-------------|
| MPNTrack | 58.8 | 61.7 | 28.8 | 33.5 | 1185 |
| ours | 60.4 | 64.9 | 30.2 | 27.9 | 1192 |

По таблице видно, что наш многоцелевой алгоритм отслеживания работает очень хорошо по всем показателям. Ниже представлен анализ каждой метрики:

1. MOTA: Наш трекер имеет оценку MOTA 60,4, что немного выше, чем у большинства других трекеров.

2. IDF1: Наш трекер имеет более высокий показатель IDF1, чем большинство других трекеров, что означает, что он лучше избегает дублирующихся обнаружений.

3. MT/%: Наш трекер имеет показатель MT 30,2%, что выше, чем у большинства других трекеров.

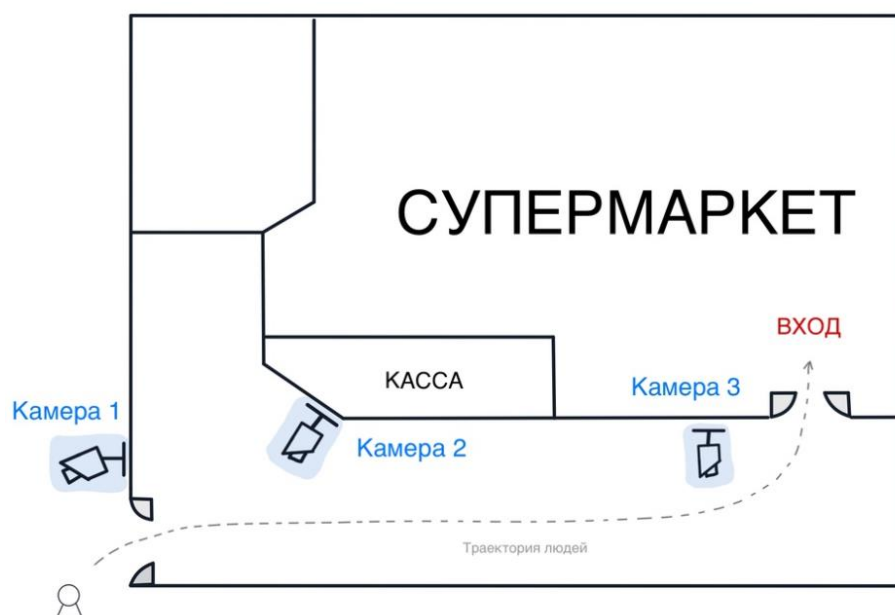
4. ML/%: Наш трекер имеет показатель ML 27,9%, что ниже, чем у большинства других трекеров.

5. IDs: IDs нашего трекера (1192) выше, чем у некоторых конкурентов (LPC_MOT и MPNTrack), но отклонение не велико.

На основании экспериментальной оценки можно утверждать, что данный улучшенный алгоритм является более универсальным, чем другими.

Экспериментальные оценки системы оценки трафика магазина

Тестовая среда



Для проверки работоспособности различных функций в этой системе тест проведен в крупном супермаркете ОКЕЙ. Мы развернули 3 фиксированные камеры по траектории пешеходов, входящих в супермаркет, для получения видео. План этажа супермаркета и расположение 3-х камер показаны на рисунке выше. Продолжительность видеозаписи около 2 минут.

Результаты статистического теста представлены в таблице ниже:

Модуль подсчета входов и выходов пешеходов

| Камера | Ground Truth | | Count | | Точность |
|--------|--------------|------------------------|-------------|------------------------|---------------|
| | Направление | Количество посетителей | Направление | Количество посетителей | |
| Cam1 | Вход | 30 | Вход | 30 | 100.00% |
| | Выход | 16 | Выход | 15 | 93.75% |
| | Всего | 46 | Всего | 45 | 97.83% |
| Cam2 | Вход | 30 | Вход | 28 | 93.33% |
| | Выход | 23 | Выход | 20 | 86.96% |
| | Всего | 53 | Всего | 48 | 90.57% |
| Cam3 | Вход | 28 | Вход | 27 | 96.43% |
| | Выход | 25 | Выход | 25 | 100.00% |
| | Всего | 53 | Всего | 52 | 98.11% |
| Всего | | | | Вход | 96.59% |
| | | | | Выход | 93.57% |
| | | | | Всего | 95.50% |

Эти данные получают путем подсчета потока людей двухстрочным методом, включающим реальные и статистические значения трех камер.

Видно, что каждая камера имеет разную точность в направлении входа и выхода, но в целом средние точности являются 96,59% (вход), 93,57% (выход) и 95,50% (вход и выход всего).

Заметили, что точность камеры 2 в направлении выхода низкая, есть 86,96%. Это из-за того, что на определенном интервале времени появилось скопление народа. Из-за перекрытия людей, было 3 пропущенных инспекционных персонала, которые понизил среднюю точность.

Точность кулачка 1 и кулачка 3 относительно высока, выше 97%. Это связано с тем, что в этих двух сценариях почти нет больших толп людей, входящих и выходящих из магазина.

В целом, несмотря на то, что еще есть возможности для повышения точности, общая производительность относительно стабильна и точна, что может помочь руководителям объектов лучше понимать поток людей.

Модуль межкамерной Ре-идентификации пешеходов

Всего в этом тесте мы оценили 20 человек. Эти 20 человек идут от камеры 1, через камеру 2, к камере 3, то есть они имеют полную траекторию, пересекающую камеру.

В cam1 20 человек, формируют 20 скриншотов посетителей , на данный момент в таблице 20 дискрипторов.



Под камерой-2 прошло 20 человек, а скриншотов сформировалось всего 19, потому что было два пешехода, идущих параллельно и один из них скрыт, показано ниже на рисунке.



Figure 3 Срыв подсчета человека из-за перекрытия людей между собой

Из-за этого, происходило одно пропущенное обнаружение. Среди 19 изображений, 18 человек получили правильную идентификаторы с помощью повторной идентификации. 1 человек не смог получить соответствующий идентификатор из-за того, что его схожесть с исходном дискриптом было слишком низким, поэтому он был назначен новый ID. Следовательно, точность повторной идентификации по кулачку-2 составляет $18/20=90,0\%$.



Figure 4 Человек, из внешнего вида которого unsuccessfully восстановить ID



Figure 5 Правильный дескриптор входящего должна быть такой

Под камерой-3 прошли 20 человек, сформировалось 19 скриншотов, потому что было три пешехода, идущих параллельно и один из них скрыт, показано ниже на рисунке.



Figure 6 Срыв подсчета человека из-за перекрытия людей между собой

Из-за этого, происходило одно пропущенное обнаружение. Среди 19 изображений, 19 человек получили правильную идентификаторы с помощью повторной идентификации. Следовательно, точность повторной идентификации по кулачку-2 составляет $19/20=95,0\%$.

Полный результат эксперимента показан следующей таблицей:

| ID чел | Cam-1 | Cam-2 | Cam-3 |
|--------|--------------------------|-----------------------|--|
| 2 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 5 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 8 | Сформировался дескриптор | Получил правильный ID | Срыв подсчета человека из-за перекрытия людей |

| | | | между собой |
|-----|--------------------------|--|-----------------------|
| 21 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 25 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 34 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 43 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 58 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 61 | Сформировался дескриптор | Срыв подсчета человека из-за перекрытия людей между собой | Получил правильный ID |
| 77 | Сформировался дескриптор | Срыв повторной идентификации, получить неверную идентификацию | Получил правильный ID |
| 123 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 183 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 186 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 188 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 190 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 236 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 249 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 250 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 267 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |
| 270 | Сформировался дескриптор | Получил правильный ID | Получил правильный ID |

И записали времени, когда они входят в магазин (виртуальное время в экспериментальной среде):





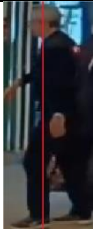
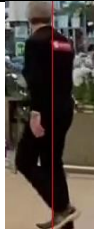


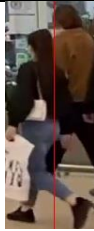



```

1  }, 'customer-2': {'cam-1': '2023-05-15 13: 26: 36', 'cam-2': '2023-05-15 13: 33: 20', 'cam-3': '2023-05-15 13: 40: 24'
2  }, 'customer-5': {'cam-1': '2023-05-15 13: 26: 45', 'cam-2': '2023-05-15 13: 33: 43', 'cam-3': '2023-05-15 13: 42: 01'
3  }, 'customer-8': {'cam-1': '2023-05-15 13: 26: 55', 'cam-2': '2023-05-15 13: 33: 31'
4  }, 'customer-21': {'cam-1': '2023-05-15 13: 26: 59', 'cam-2': '2023-05-15 13: 36: 31', 'cam-3': '2023-05-15 13: 45: 21'
5  }, 'customer-25': {'cam-1': '2023-05-15 13: 27: 00', 'cam-2': '2023-05-15 13: 33: 37', 'cam-3': '2023-05-15 13: 40: 38'
6  }, 'customer-34': {'cam-1': '2023-05-15 13: 27: 11', 'cam-2': '2023-05-15 13: 33: 47', 'cam-3': '2023-05-15 13: 40: 55'
7  }, 'customer-43': {'cam-1': '2023-05-15 13: 27: 25', 'cam-2': '2023-05-15 13: 33: 56', 'cam-3': '2023-05-15 13: 41: 01'
8  }, 'customer-58': {'cam-1': '2023-05-15 13: 27: 28', 'cam-2': '2023-05-15 13: 34: 10', 'cam-3': '2023-05-15 13: 44: 10'
9  }, 'customer-61': {'cam-1': '2023-05-15 13: 27: 30', 'cam-3': '2023-05-15 13: 41: 49'
10 }, 'customer-77': {'cam-1': '2023-05-15 13: 27: 39', 'cam-3': '2023-05-15 13: 41: 28'
11 }, 'customer-123': {'cam-1': '2023-05-15 13: 28: 39', 'cam-2': '2023-05-15 13: 39: 09', 'cam-3': '2023-05-15 13: 42: 44'
12 }, 'customer-183': {'cam-1': '2023-05-15 13: 30: 16', 'cam-2': '2023-05-15 13: 37: 35', 'cam-3': '2023-05-15 13: 45: 21'
13 }, 'customer-186': {'cam-1': '2023-05-15 13: 30: 19', 'cam-2': '2023-05-15 13: 37: 51', 'cam-3': '2023-05-15 13: 44: 04'
14 }, 'customer-188': {'cam-1': '2023-05-15 13: 30: 40', 'cam-2': '2023-05-15 13: 38: 04', 'cam-3': '2023-05-15 13: 43: 25'
15 }, 'customer-190': {'cam-1': '2023-05-15 13: 30: 26', 'cam-2': '2023-05-15 13: 37: 00', 'cam-3': '2023-05-15 13: 43: 09'
16 }, 'customer-236': {'cam-1': '2023-05-15 13: 31: 37', 'cam-2': '2023-05-15 13: 38: 30', 'cam-3': '2023-05-15 13: 44: 41'
17 }, 'customer-249': {'cam-1': '2023-05-15 13: 31: 56', 'cam-2': '2023-05-15 13: 39: 02', 'cam-3': '2023-05-15 13: 47: 47'
18 }, 'customer-267': {'cam-1': '2023-05-15 13: 32: 27', 'cam-2': '2023-05-15 13: 45: 59', 'cam-3': '2023-05-15 13: 48: 49'
19 }, 'customer-270': {'cam-1': '2023-05-15 13: 32: 32', 'cam-2': '2023-05-15 13: 45: 57', 'cam-3': '2023-05-15 13: 53: 03'
20 }
21

```

Всего было успешно повторно идентифицировано 17 человек из 20.

Частичная таблица представлена ниже:

| ID | Cam-1 | Cam-2 | Cam-3 |
|----|---|---|---|
| 2 |  |  |  |
| 5 |  |  |  |
| 21 |  |  |  |
| 25 |  |  |  |



Экспериментальный анализ: В общей сложности 20 человек прошли под три камерами. Из них 17 были полностью отслежены, что означает, что они были правильно идентифицированы и отслежены на трех камерах. Два человека не были полностью отслежены из-за пропущенных обнаружений, поэтому они не учитывались в полном отслеживании. Один человек был повторно распознан, но из-за недостаточной степени сходства не удалось восстановить его первоначальный ID, поэтому он также не учитывался в полном отслеживании.

Таким образом, всего 20 человек, из которых 17 были успешно отслежены, 2 человека не были отслежены, а один был повторно распознан, но распознавание не удалось. Мы можем вычислить точность (ассигуру) как следующее:

$$\text{Точность} = \frac{\text{число правильно отслеженных людей}}{\text{общее число людей}} = \frac{17}{20} = 0.85$$

Экспериментальный вывод: точность отслеживания нашей системы статистики трафика достигает примерно $17/20=85,0\%$.

ЗАКЛЮЧЕНИЕ

Работа данной статьи является исследованием применения технологий нейронной сети для оценки трафика магазина. Реализован алгоритм YOLOv5 для обнаружения пешеходов. Реализован алгоритм отслеживания DeepSORT для отслеживания пешеходов. Реализован алгоритм FastReID для повторно идентификации пешеходов под несколькими камерами. Реализован двухстрочный метод для подсчета количества посетителей магазина.

Хранения и записи изображений посетителей, характеристика и времени входа и выхода. В конце, разработала систему оценки трафика магазина на сцене супермаркета.

Основная работа данной статьи включает в себя следующие аспекты:

1. Повторная идентификация пешехода

Реализован алгоритм повторной идентификации пешехода FastReID, который используется для установления связи между идентификаторами пешеходов под несколькими камерами. Модель хорошо зарекомендовала себя на наборах данных Market1501 и DukeMTMC со средней точностью (mAP) 88,4% и 81,3%.

2. Отслеживание пешеходов

На основе алгоритмов YOLOv5, FastReID и DeepSORT разработан и реализован трекер пешеходов, который может выполнять многоцелевое отслеживание под одной и той же камерой, а также может идентифицировать личности между камерами. Результаты тестирования на MOT16: точность отслеживания (MOTA) 66,2% и точность отслеживания (MOTP) 80,8%.

3. Применение системы подсчета посетителей в супермаркете

В данной статье разработана система оценки потоков людей на основе видео наблюдения, которая фиксирует количество посетителей, изображения посетителей и время, когда посетители появляются в каждой зоне супермаркета на определенном интервале времени. Точность идентификации входящих и выходящих людей достигла 95,50%, а точность отслеживания + повторной идентификации пешеходов достигла 85%.

4. Сравнить с другими продвинутыми алгоритмами на других наборах данных и выясните преимущества и недостатки модели FastReID и модели DeepSORT.

В сценариях практического применения система отслеживания множества объектов имеет высокую точность. Но когда появится скопление народа, например 50 около входа, точность отслеживания будет снижена из-за

перекрытия. Точность можно повысить, улучшив алгоритмы отслеживания объектов, например, использовать обнаружение центральной точки вместо традиционного обнаружения ограничивающей рамки, использовать облегченную структуру сверточной нейронной сети, применить технологию сегментации объектов для избежания пропущенного обнаружения и т.д.

СПИСОК ЛИТЕРАТУРЫ

1. Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C]. Proceedings of the IEEE conference on computer vision and pattern recognition, 2014: 580-587.
2. He K, Zhang X, Ren S, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition[J]. IEEE transactions on pattern analysis and machine intelligence, 2015, 37(9): 1904-1916.
3. Girshick R. Fast r-cnn[C]. Proceedings of the IEEE international conference on computer vision, 2015: 1440-1448.
4. Ren S, He K, Girshick R, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[J]. Advances in neural information processing systems, 2015, 28: 91-99
5. Redmon J, Divvala S, Girshick R, et al. You only look once: Unified, real-time object detection[C]. Proceedings of the IEEE conference on computer vision and pattern recognition, 2016: 779-788.
6. Redmon J., Farhadi A. YOLO9000: Better, Faster, Stronger //arXiv preprint arXiv:1612.08242. – 2016.
7. <https://github.com/ultralytics/yolov5>
8. Захаров, Н. С. Отслеживание людей по видео последовательности / Н. С. Захаров // Актуальные проблемы авиации и космонавтики : Сборник материалов VI Международной научно-практической конференции, посвященной Дню космонавтики. В 3-х томах, Красноярск, 13–17 апреля 2020 года / Под общей редакцией Ю.Ю.

- Логинова. Том 2. – Красноярск: Федеральное государственное бюджетное образовательное учреждение высшего образования "Сибирский государственный университет науки и технологий имени академика М.Ф. Решетнева", 2020. – С. 135-137. – EDN ZKIUDD.
9. Богущ, Р. П. Сопровождение и повторная идентификация людей в интеллектуальных системах видеонаблюдения с применением сверточных нейронных сетей / Р. П. Богущ, С. А. Игнатьева, С. В. Абламейко // Первая выставка-форум IT-академграда «Искусственный интеллект в Беларуси» : сборник докладов, Минск, 13–14 октября 2022 г. – Минск : ОИПИ НАН Беларуси, 2022. – С. 46-53.
 10. Ye, S. Person Tracking and Re-Identification in Video for Indoor Multi-Camera Surveillance Systems / S. Ye, R. Bohush, C. Chen, I. Zakharava, S. Ablameyko // Pattern Recognition and Image Analysis, 2020. - Vol. 30, №4 – P. 827-837
 11. Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. San Diego, CA, USA: IEEE, 2005. 886–893
 12. Lowe D G. Object recognition from local scale-invariant features. In: Proceedings of the 7th IEEE International Conference on Computer Vision. Kerkyra, Greece: IEEE, 1999. 1150–1157
 13. K'ostinger M, Hirzer M, Wohlhart P, Both P M, Bischof H. Large scale metric learning from equivalence constraints. In: Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition. Providence, RI, USA: IEEE, 2012. 2288–2295
 14. Liao S C, Hu Y, Zhu X Y, Li S Z. Person re-identification by local maximal occurrence representation and metric learning. In: Proceedings of the 2015

- IEEE Conference on Computer Vision and Pattern Recognition. Boston, MA, USA: IEEE, 2015. 2197–2206
15. K M, Zhang X Y, Ren S Q, Sun J. Delving deep into rectifiers: surpassing human-level performance on ImageNet classification. In: Proceedings of the 2015 IEEE International Conference on Computer Vision. Santiago, Chile: IEEE, 2015. 1026–1034
 16. Lu C C, Tang X O. Surpassing human-level face verification performance on LFW with Gaussian face. In: Proceedings of the 29th AAAI Conference on Artificial Intelligence. Austin, Texas, USA: AAAI, 2015. 3811–3819
 17. Lingxiao He, Xingyu Liao FastReID: A Pytorch Toolbox for General Instance Re-identification <https://arxiv.org/abs/2006.02631>
 18. Wei L, Zhang S, Gao W, Tian Q. Person Transfer GAN to Bridge Domain Gap for Person Re-identification / L. Wei [et al.]// Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition. - 2018. - P. 79-88
 19. Ruibing Hou, Bingpeng Ma, Hong Chang, Xinqian Gu, Shiguang Shan, and Xilin Chen. Interaction-and-aggregation network for person re-identification. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019
 20. Ruijie Quan, Xuanyi Dong, Yu Wu, Linchao Zhu, and Yi Yang. Auto-reid: Searching for a part-aware convnet for person re-identification. In The IEEE International Conference on Computer Vision (ICCV), 2019.
 21. Kaiyang Zhou, Yongxin Yang, Andrea Cavallaro, and Tao Xiang. Omni-scale feature learning for person reidentification. In The IEEE International Conference on Computer Vision (ICCV), 2019. 2
 22. Binghui Chen, Weihong Deng, and Jiani Hu. Mixed highorder attention network for person re-identification. In The IEEE International Conference on Computer Vision (ICCV), 2019

23. Yifan Sun, Changmao Cheng, Yuhao Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. Circle loss: A unified perspective of pair similarity optimization. 2020.
24. Bewley A, Ge Z, Ott L, et al. Simple online and realtime tracking[C]//2016 IEEE international conference on image processing (ICIP). IEEE, 2016: 3464-3468.
25. Wang Z, Zheng L, Liu Y, et al. Towards real-time multi-object tracking[C]//Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16. Springer International Publishing, 2020: 107-122.
26. Scalable Person Re-identification: A Benchmark/ L. Zheng [et. al] // Proc of IEEE International Conference on Computer Vision (ICCV). - 2015. – P. 1116-1124.
27. Performance Measures and a Data Set for Multi-target, Multi-camera Tracking [Electronic resource]. – Mode of access: <https://arxiv.org/abs/1609.01775>. – Date of access: 03.09.2022.
28. <https://motchallenge.net/results/MOT17/>
29. Wojke N, Bewley A, Paulus D. Simple online and realtime tracking with a deep association metric[C]//2017 IEEE international conference on image processing (ICIP). IEEE, 2017: 3645-3649.