ПРОГРАММИРОВАНИЕ, 2022, № 3, с. 11–21

КОМПЬЮТЕРНАЯ ГРАФИКА И ВИЗУАЛИЗАЦИЯ

ВИДЕОАНАЛИТИКА С ПРИМЕНЕНИЕМ ДЕТЕКЦИИ НА РАЗРЕЖЕННЫХ КАДРАХ

© 2022 г. Т. З. Мамедов[a,b,*] (ORCID: 0000-0001-6554-7988),
Д. А. Купляков[a,b,] (ORCID: 0000-0002-2957-3297),**
А. С. Конушин[a,c,*] (ORCID: 0000-0002-6152-0021)**

[a] Московский государственный университет им. М.В. Ломоносова, 119991 Москва, ГСП-1, Ленинские горы, д. 1-52, Россия
[b] ООО "Технологии видеоанализа", 119634 Москва, ул. Скульптора Мухиной, д. 7, этаж 1 пом. II ком. 2В, Россия
[c] Национальный исследовательский университет "Высшая школа экономики", 109028 Москва, Покровский бульвар, д. 11, Россия
**E-mail: timur.mamedov@graphics.cs.msu.ru*
***E-mail: denis.kuplyakov@graphics.cs.msu.ru*
****E-mail: anton.konushin@graphics.cs.msu.ru*

Поступила в редакцию 26.12.2021 г.
После доработки 10.01.2022 г.
Принята к публикации 20.01.2022 г.

В данной работе рассматриваются две задачи видеоаналитики, которые могут быть решены с помощью сопровождения людей в видеопотоке, – подсчет людей и оценка времени ожидания в очередях. Современные системы видеонаблюдения насчитывают несколько сотен тысяч камер, поэтому одна из важнейших проблем, с которой приходится сталкиваться при видеоаналитике, – оптимизация использования вычислительных ресурсов. Большинство текущих алгоритмов сопровождения являются неэффективными ввиду того, что они используют вычислительно затратные нейросетевые детекторы на частых кадрах видеопотока. В данной работе предлагаются методы решения упомянутых ранее задач, призванные устранить эту проблему путем применения детекций на разреженных кадрах. Проведенная экспериментальная оценка предложенных методов показала их состоятельность как с точки зрения качества работы, так и используемых вычислительных ресурсов.

DOI: 10.31857/S0132347422030074

1. ВВЕДЕНИЕ

В последние годы алгоритмы компьютерного зрения особенно активно начали использоваться в видеоаналитике. Суть последней заключается в извлечении и анализе значимой информации с камер видеонаблюдения для последующего принятия тех или иных бизнес-решений, оптимизации процессов и т.д. В данной работе рассматриваются две практически важные задачи видеоаналитики – подсчет людей и оценка времени ожидания в очередях.

Существует множество подходов для решения данных задач, например, за счет сопровождения (трекинга) людей в видеопотоке. Трекинг подразумевает построение траекторий движения людей, которые впоследствии могут быть использованы для анализа происходящих в видеопотоке событий.

Современные алгоритмы сопровождения используют нейросетевые детекторы для получения детекций людей для дальнейшего их связывания в траектории. Применение таких детекторов ведет к большим вычислительным затратам, так как для их эффективной работы необходимо задействовать видеокарты (GPU). Последние, в свою очередь, являются дорогими, особенно во времена роста популярности криптовалюты и кризиса полупроводников. Также стоит отметить, что современные алгоритмы сопровождения требуют запуска детектора с достаточно высокой частотой (то есть детектирование объектов происходит на близких по времени кадрах). Это связано с тем, что алгоритмы тем или иным образом полагаются на то, что интервал времени между обнаружениями объектов является небольшим.

Ввиду описанных выше причин и того, что в реальных сценариях приходится работать с крупномасштабными системами видеонаблюдения, масштабировать существующие методы трекинга весьма затратно. Поэтому одним из способов

снижения расходов (как денежных, так и вычислительных) для осуществления видеоаналитики является разработка алгоритмов сопровождения, способных работать на разреженных кадрах.

Способность метода трекинга работать на разреженных кадрах влечет за собой снижение частоты детекции. А последнее, в свою очередь, позволяет использовать одну видеокарту для обработки большего числа видеопотоков и тем самым сократить стоимость системы. Более того, при этом обработка видеопотоков может полностью осуществляться в специализированных центрах обработки данных (ЦОД), что способствует эффективному разделению ресурсов GPU. Также возможен вариант, при котором в ЦОД может быть вынесен только алгоритм детектирования, а остальная обработка будет осуществляется непосредственно на "узловых" устройствах. Такой подход позволяет передавать в ЦОД не видеопотоки, а лишь разреженные кадры, что влечет за собой уменьшение затрат на передачу данных.

С другой стороны, снижение частоты детекции позволяет реализовать алгоритм сопровождения без использования видеокарты в случае применения менее вычислительно затратного и менее точного алгоритма нейросетевой детекции. При условии сохранения приемлемого качества работы алгоритма подобное решение является более приоритетным для практического применения, так как в значительной степени снижает денежные и вычислительные затраты.

Современные методы сопровождения людей используют подход к сопровождению через детектирование. Основываясь на видах детектирования, алгоритмы трекинга можно условно разделить на две группы: использующие детекции тел людей [1–3] и детекции голов [4, 5].

Решения, использующие детекции голов, хорошо подходят для сопровождения людей, находящихся в толпе. Это связано с тем, что в большинстве случаев камеры видеонаблюдения расположены на высоте, большей среднего роста человека, поэтому в видеопотоке головы людей в таких случаях намного лучше видно, чем их тела. Также стоит отметить тот факт, что головы менее подвержены перекрытиям, что благоприятно сказывается на качестве трекинга и видеоаналитики в целом.

После получения детекций необходимо произвести их объединение в траектории. Существует множество способов решения данной задачи, например, жадные алгоритмы. В большинстве методов сопровождения реального времени траектории строятся от кадра к кадру – для каждого кадра вычисляется "матрица стоимостей" сопоставления новых детекций с существующими траекториями. Задача объединения детекций в траектории решается либо с помощью жадного алгоритма (ищется максимальное значение в каждой строке/столбце матрицы) [6, 7], либо решается задача о назначениях с помощью Венгерского алгоритма [1–3, 8]. Помимо этого, для решения данной задачи используются марковские цепи Монте-Карло [5].

Не стоит забывать о нейросетевых методах сопровождения людей, которые начали набирать популярность в последнее время. Так, например, в [9] авторы предлагают получать новые детекции с помощью регрессии детекций на предыдущем кадре. Однако подобный подход имеет недостаток – для его работы необходима высокая частота кадров, что ведет к увеличению частоты детектирования и вычислительных затрат.

В данной работе рассматриваются две задачи видеоаналитики – подсчет людей и оценка времени ожидания в очередях, а также предлагаются два подхода уменьшения частоты детектирования. Для решения задачи подсчета людей предлагается метод, способный эффективно сопровождать людей на коротких интервалах времени, а для задачи оценки времени ожидания в очередях – алгоритм, позволяющий эффективно оценивать время присутствия человека в кадре на длинных промежутках, используя детектирование на очень редких кадрах.

2. ЗАДАЧА ПОДСЧЕТА ЛЮДЕЙ

Задача подсчета людей, проходящих через определенные зоны социальной инфраструктуры (например, пешеходные переходы, тротуары, площади и т.д.), является практически важной. Так как при решении данной задачи приходится сталкиваться с обработкой огромных потоков данных, то возникает потребность в автоматизации процедуры подсчета людей. Существует множество алгоритмов подсчета людей, большинство из которых основаны на сопровождении людей. Ввиду того, что в трекинге с каждым человеком ассоциируется траектория, то подсчет людей в таких методах осуществляется за счет фиксации факта пересечения траекторией сигнальной линии.

В данной работе предлагается полностью автоматизированный распределенный алгоритм подсчета людей, являющийся улучшением метода [4], а также решение, способное работать на одном ядре процессора без использования видеокарт. Алгоритм получает на вход видеопоток $\{F_i\}_{i=1}$ кадров, снятых на стационарную камеру, и сигнальную линию, представленную в виде упорядоченной пары точек (L_a, L_b) на кадре. Выходом алгоритма является множество событий $\{E_i\}_{i=1}$, описываемых тройками $E_i = (k_i, r_i, d_i)$, где k_i – номер кадра, на котором произошло пересечение сигнальной линии, r_i определяет координаты обрамляющего

прямоугольника (bbox) детекции, а последнее значение d_i отвечает за направление пересечения сигнальной линии.

2.1. Базовый метод

В данной работе в качестве базового метода (baseline) используется алгоритм, представленный в [4]. Он является улучшением классического SORT-алгоритма [1]. Выбор был сделан в пользу данного метода ввиду того, что он способен работать на разреженных кадрах, что позволяет значительно снизить вычислительные затраты при его применении в крупномасштабных системах видеонаблюдения (см. раздел 1).

Выбранный baseline работает в режиме реального времени и использует Венгерский алгоритм для объединения детекций в траектории, а для улучшения результатов на низких частотах детектирования в нем задействовано визуальное сопровождение ASMS [10] для оценки скорости движения людей.

Стоит отметить, что базовый метод имеет несколько недостатков, которые усложняют его применение на практике, например:

• данный алгоритм основан на использовании детекций голов, однако люди могут иметь разный рост, что ведет к тому, что их головы могут быть расположены на разных плоскостях. Таким образом не всегда очевидно на какую высоту стоит поднять сигнальную линию, чтобы корректно фиксировать факт пересечения сигнальной линии (рис. 1);

• предыдущая проблема была решена в [4] с помощью линейной регрессии из головы в тело. Однако данную регрессию необходимо переобучать для каждой конкретной сцены, а это усложняет практическое применение алгоритма.

2.2. Предложенный метод

Выбранный базовый алгоритм состоит из следующих этапов: (1) детектирование; (2) предсказание скорости детекций с помощью визуального сопровождения; (3) предсказание позиции траектории с помощью фильтра Калмана; (4) объединение детекций в траектории; (5) экстраполяция траекторий; (6) фиксация факта пересечения сигнальной линии.

В данной работе предлагаются улучшения следующих этапов в baseline с целью его оптимизации для практического применения: детектирование, предсказание скорости детекций с помощью визуального сопровождения, фиксация факта пересечения сигнальной линии. Ниже описаны все предложенные модификации.

Рис. 1. Представьте, где здесь необходимо нарисовать сигнальную линию на уровне головы.

2.2.1. Детектирование

Так как головы менее подвержены перекрытиям и их лучше видно с камер видеонаблюдения, то было принято решение продолжить идею использования детектора голов вместо детектора тел, предложенную в [4]. Поэтому в предлагаемых алгоритмах применяется детектор голов, основанный на SSD [11]. В данной работе используются две базовые архитектуры (backbone) для детектора голов – ResNet50 [12] для усовершенствованного распределенного алгоритма с высоким качеством подсчета людей и MobileNet0.5 [13] для решения, способного работать на одном ядре процессора. Детектор обучен на публичном наборе данных CrowdHuman [14] и коллекции, собранной компанией ООО “Технологии видеоанализа”.

2.2.2. Визуальное сопровождение

Как было сказано ранее, в базовом алгоритме для оценки скорости детекций используется визуальное сопровождение ASMS. В данной работе предлагается заменить данный алгоритм визуального сопровождения на Staple [15], который позволяет значительно ускорить общее решение для подсчета людей, но при этом сохранить качество подсчета на прежнем уровне. Staple состоит из двух моделей, работающих параллельно. Первая модель является инвариантной к смене освещения, но зависит от изменения размеров объектов, вторая – напротив, инвариантна к трансформациям объектов, но зависит от освещения. Использование двух моделей вместе позволяет достичь наилучших результатов.

2.2.3. Фиксация факта пересечения сигнальной линии

Baseline использует детекции голов по причинам, описанным ранее в разделе 2.2.1. Однако такой подход обладает недостатком, описанным в разд. 2.1. В базовом алгоритме эта проблема была решена с помощью линейной регрессии, однако ее необходимо переобучать для каждой конкретной

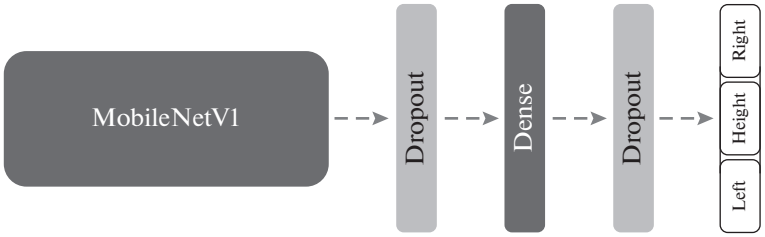

Рис. 2. Архитектура предложенной нейросетевой регрессии из головы в тело. В качестве backbone сети используется MobileNetV1 [13].

сцены, а это усложняет практическое применение метода. В данной работе предлагается новая нейросетевая регрессия из головы в тело, которая позволяет решить все упомянутые выше проблемы. Предложенный регрессор необходимо обучить единожды для применения на любой сцене.

Нейросетевой регрессор из головы в тело состоит из 2 этапов:

1). с помощью эвристики находится приблизительная позиция тела человека;

2). с помощью нейросетевой регрессии происходит уточнение приблизительной позиции тела человека.

Первый этап. В качестве эвристики используется следующий анатомический факт: ширина тела человека в среднем равна 3 ширинам его головы, а высота человека – 8 высотам его головы. Экспериментальным путем данные пропорции были немного скорректированы:

$$body_{left} = head_{left} - head_{width}, \quad (2.1)$$

$$body_{top} = head_{top}, \quad (2.2)$$

$$body_{width} = 3 \times head_{width}, \quad (2.3)$$

$$body_{height} = 8.5 \times head_{height}, \quad (2.4)$$

где $(head_{left}, head_{top}, head_{width}, head_{height})$ и $(body_{left}, body_{top}, body_{width}, body_{height})$ – координаты обрамляющих прямоугольников головы и тела, соответственно.

Второй этап. На втором этапе происходит уточнение позиции bbox тела, полученного на предыдущем шаге. Для этого была разработана нейросеть, способная регрессировать три значения: левые и правые крайние точки тела человека ($body_{regLeft}$ и $body_{regRight}$) и высоту $body_{regHeight}$ тела человека. На рис. 2 представлена архитектура предложенной сети. Данная нейросеть имеет простую архитектуру, потому что алгоритм подсчета людей должен работать в режиме реального времени. И несмотря на свою простоту, предложенный нейросетевой регрессор показывает весьма хорошее качество работы. В итоге, после двух этапов координаты срегрессированного тела имеют вид: $(body_{regLeft}, body_{top}, body_{regWidth}, body_{regHeight})$, где $body_{regWidth} = body_{regRight} - body_{regLeft}$.

В предложенном решении для сопровождения людей используются детекции голов, а для фиксации факта пересечения сигнальной линии применяются детекции тел, полученные нейросетевой регрессией. То есть каждая детекция головы связана с детекцией тела. Из-за перекрытий возможны ошибки регрессии, которые могут повлечь за собой ложные пересечения сигнальной линии, поэтому, когда происходит добавление новой детекции тела в трек, осуществляется корректировка ее высоты и ширины следующим образом:

$$bodyNew_{width} = \alpha \times bodyNew_{width} + (1-\alpha) \times bodyPrev_{width}, \quad (2.5)$$

$$bodyNew_{height} = \alpha \times bodyNew_{height} + (1-\alpha) \times bodyPrev_{height}, \quad (2.6)$$

где $bodyNew_{width}, bodyNew_{height}$ – ширина и высота добавляемого bbox тела, $bodyPrev_{width}, bodyPrev_{height}$ – ширина и высота последнего bbox тела в траектории, и α – специально выбранная константа.

2.3. Экспериментальная оценка

2.3.1. Наборы данных

Регрессия из головы в тело. Для обучения предложенной нейросетевой регрессии (см. раздел 2.2.3) использовались видеозаписи из публичных наборов данных 2DMOT2015 [16], MOT17 [17] и коллекции, собранной компанией ООО “Технологии видеоанализа”. Для получения тренировочных данных для нейронной сети из видео применялась следующая стратегия (рассматривались кадры каждые 2 секунды):

1). используя детектор голов (см. раздел 2.2.1), на каждом кадре были получены детекции голов;

2). используя линейную регрессию из [4], на каждом кадре были получены детекции тел;

3). используя эвристику (см. раздел 2.2.3), на каждом кадре были найдены приблизительные позиции тел людей. Эти данные применялись для

создания кропов обучающей выборки нейросетевого регрессора из головы в тело;

4. используя детекции тел, относящиеся к детекциям голов, для каждого кадра были найдены левые и правые крайние точки тел людей, а также их высоты. Эти данные применялись для обучения нейросетевой регрессии.

Подсчет людей. Для экспериментальной оценки предложенного алгоритма необходимы продолжительные видеозаписи, снятые на статичную камеру. По этой причине являются непригодными большинство публичных наборов данных (например, популярный набор MOTChallenge [18] содержит короткие видео, снятые на подвижные камеры). Таким образом, для тестирования методов использовались 19 видеозаписей из коллекции компании ООО "Технологии видеоанализа", а также публичный набор данных Towncentre [19]. Для каждого видео вручную были нарисованы сигнальные линии на уровне земли.

2.3.2. Метрики

В роли метрики качества в данной работе используется средняя ошибка подсчета числа пересечений сигнальной линии (событий) [20]. Результирующее множество событий, выданное алгоритмом, может включать в себя как положительные, так и отрицательные примеры. Отрицательные события не имеют пары в "эталонной" разметке. Будем считать событие E_i, выданное алгоритмом, совпадающим с событием $\widehat{E_i}$ из "эталонной" разметки, если они относятся к одному и тому же человеку, и произошли в один и тот же момент времени. Сопоставление событий происходит при помощи метода, описанного в [20].

После сопоставления событий видеозапись делится на сегменты, содержащие по 10 "эталонных" событий, и вычисляются следующие характеристики:

- GT_{seg} – число "эталонных" событий на рассматриваемом сегменте;
- FP_{seg} – число событий, выданных алгоритмом, для которых не было найдено пары в "эталонной" разметке на рассматриваемом сегменте;
- FN_{seg} – число "эталонных" событий, для которых не было найдено пары в множестве событий, выданных алгоритмом, на рассматриваемом сегменте;
- $E_{seg} = \dfrac{FP_{seg} - FN_{seg}}{GT_{seg}}$ – ошибка на сегменте.

Итоговая ошибка вычисляется следующим образом: $E = \sum_{i=1}^{N} \dfrac{E_{seg}}{N}$, где N – количество сегментов.

Таблица 1. Экспериментальная оценка предложенных алгоритмов подсчета людей в зависимости от частоты детекции (значение ошибки ↓)

Алгоритм/Частота кадров	5 Гц	3 Гц	2 Гц
SORT [1]	8.3	10.8	17.3
Старый Baseline [20]	7.5	7.5	7.5
Baseline [4]	4.1	4.1	4.3
Регрессия тела	4.8	4.7	5.1
Регрессия тела и Staple	5.0	4.6	5.1
Одноядерный алгоритм	8.9	9.3	9.1

2.3.3. Результаты экспериментов

В данном разделе представлены результаты экспериментов для предложенных алгоритмов. Были рассмотрены три типа экспериментов:

- **Регрессия тела** – в данном эксперименте использовался базовый алгоритм [4], в котором линейная регрессия из головы в тело была заменена новой нейросетевой регрессией (см. раздел 2.2.3);
- **Регрессия тела и Staple** – данный эксперимент аналогичен предыдущему за тем исключением, что визуальное сопровождение ASMS было заменено на Staple (см. раздел 2.2.2);
- **Одноядерный алгоритм** – в данном эксперименте были применены модификации, упомянутые в предыдущих экспериментах, а также был использован легковесный детектор голов с MobileNet0.5 в качестве backbone (см. раздел 2.2.1).

В таблице 1 представлены результаты проведенных экспериментов.

Экспериментальная оценка показала, что предложенный улучшенный распределенный алгоритм имеет высокую точность подсчета людей, и значительно превосходит по качеству старый baseline из [20] и незначительно уступает базовому алгоритму из [4]. Однако предложенный алгоритм может быть использован в практических целях, так как он не требует настройки для каждой новой сцены.

Результаты из таблицы 1 также показывают, что предложенный алгоритм, способный работать на одном ядре процессора и без использования GPU, имеет качество, приемлемое для практического применения. Более того, он превосходит метод подсчета людей, основанный на классическом трекинге SORT [1], как в плане качества подсчета людей, так и с точки зрения используемых ресурсов.

2.3.4. Оценка скорости алгоритма

Ввиду того, что предложенный метод состоит из нескольких этапов, общее время его работы равно сумме времен работы каждого из его компонентов: детектора, визуального сопровождения и нейросетевой регрессии из головы в тело.

Легковесному детектору голов требуется ≈250 мс для работы на Full HD кадре (см. раздел 2.2.1). Визуальное сопровождение Staple [15] (см. раздел 2.2.2) требует ≈5 мс для одной детекции (ASMS, используемый в базовом алгоритме [4], затрачивает ≈20 мс для одной детекции). Предложенный нейросетевой регрессор из головы в тело работает ≈130 мс для 10 кропов.

Если принебречь остальными незначительными вычислениями, то предложенный одноядерный метод способен работать при частоте 2 Гц и 10 людях на каждом кадре (≈500 мс). Все временные замеры были произведены в одноядерном режиме с использованием процессора Intel Core i5-9400.

3. ЗАДАЧА ОЦЕНКИ ВРЕМЕНИ ОЖИДАНИЯ В ОЧЕРЕДЯХ

Задача оценки времени ожидания в очередях является практически важной, так как алгоритмы, решающие ее, используются в коммерческих целях для улучшения качества обслуживания клиентов.

В данной работе предлагается полностью автоматизированный метод оценки времени ожидания в очередях, основанный на использовании трекинга и реидентификации (Re-ID) по верхней части тела человека, способный работать на очень редких кадрах. Входом алгоритма является видеопоток $\{F_i\}_{i=1}$ кадров, снятых на статичную камеру, координаты региона интереса (ROI), а также t – продолжительность сегмента видео в секундах. На выходе алгоритм выдает множество $\{T_i\}_{i=1}$, состоящее из оценок максимального времени ожидания для каждого сегмента длины t секунд.

3.1. Предложенный метод

Предложенный метод состоит из 5 основных этапов:

• **Детектирование** – используя детектор голов, осуществляется поиск голов всех людей на каждом разреженном кадре;

• **Регрессия верхней части тела** – на данном шаге на каждом разреженном кадре для каждой найденной головы происходит регрессия верхней части тела человека;

• **Реидентификация** – для каждой срегрессированной верхней части тела с помощью алгоритма реидентификации вычисляется “вектор внешнего вида”;

• **Трекинг** – полученные на предыдущем шаге векторы используются для связывания детекций в траектории или для создания новых траекторий;

• **Оценка времени ожидания** – результатом предыдущего этапа является множество траекторий $\{Tr_i\}_{i=1}$, $Tr_i = \{f_j, d_j\}_{j=1}$, где f_j, d_j – кадр и координаты bbox, соответственно. Используя полученное множество, имеется возможность вычислить время ожидания в очереди, как максимальное время жизни (под временем жизни $L(tr)$ траектории tr понимается продолжительность ее нахождения в видеопотоке) одной из траекторий из данного множества.

Далее будет описан каждый из первых четырех этапов предложенного алгоритма.

3.1.1. Детектирование

Как и в алгоритме, предложенном для решения задачи подсчета людей, в данном методе используется детектор голов, описанный в разделе 2.2.1. Его ключевая особенность заключается в том, что он является быстрым и обладает приемлемым качеством для практического применения.

3.1.2. Регрессия верхней части тела

Ключевая идея предложенного метода оценки времени ожидания в очередях заключается в использовании детекции верхней части тела человека, нежели всего тела, для алгоритма реидентификации. Это связано с тем, что в очередях верхняя часть человеческого тела лучше видна и при этом она содержит больше визуальной информации для Re-ID, чем, например, голова.

По этой причине предлагается произвести регрессию из головы в верхнюю часть тела по аналогии тому, как это было сделано в разделе 2.2.3. Под верхней частью человеческого тела будем понимать область тела от головы до груди.

Как и в случае с регрессией из головы в тело (см. раздел 2.2.3), сначала происходит поиск приблизительной области верхней части тела с помощью эвристики (в выражении (2.4) коэффициент 8.5 заменяется на 2), а потом ее уточнение с помощью нейросети. Последняя, в свою очередь, получая на вход найденную приблизительную область, регрессирует два значения – левую и правую крайние точки верхней части тела.

Подобное уточнение краев bbox особенно важно в случае, когда человек расположен в профиль на кадре (рис. 3).

3.1.3. Реидентификация

Нейронная сеть для реидентификации. В качестве baseline для реидентификации было выбрано решение из [21], потому что оно обладает высоким качеством решения данной задачи, а также имеет значительную скорость работы, что весьма важно для практического применения. В выбранный метод Re-ID были внесены две модификации с целью повышения его качества:

• ResNet50 был заменен на Res2Net50 [22], потому что последняя нейросеть имеет лучшее качество на задаче классификации, но при этом не сильно уступает по скорости работы ResNet50;

• Triplet Loss и Center Loss [23], используемые в baseline, были заменены на FIDI Loss [24]. Эта

Эвристика Нейросетевая регрессия

Рис. 3. Сравнение эвристики и нейросетевой регрессии из головы в верхнюю часть тела.

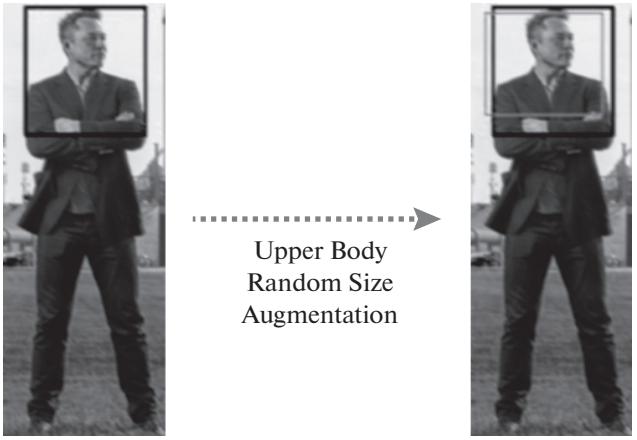

Рис. 4. Пример работы аугментации UBRSA.

функция потерь предназначена для задачи Re-ID, она сильно штрафует за незначительные различия между изображениями, что очень важно для реидентификации.

Upper Body Random Size Augmentation. Рассмотрим, например, случай реидентификации по телам людей. Чаще всего для обучения нейронной сети используются изображения, где люди представлены в полный рост. Однако в реальных сценариях, ввиду окклюзий (перекрытий) и/или ошибок детектора, на вход алгоритму реидентификации могут подаваться кропы, содержащие только часть человеческого тела (например, верхнюю часть тела). На таких примерах велика вероятность неверного решения нейросетью задачи Re-ID, так как ранее она "видела" людей только в полный рост.

Аналогичное возможно и в случае реидентификации по верхней части тела. Для решения данной проблемы предлагается новая аугментация Upper Body Random Size Augmentation (UBRSA), которая призвана улучшить качество Re-ID в реальных сценариях. Ключевая ее идея заключается в случайном изменении краев bbox верхней части тела (рис. 4). Таким образом на этапе обучения сети происходит симуляция случаев, описанных выше.

Пусть ($left_1$, top_1, $width_1$, $height_1$) – координаты bbox верхней части тела, полученные регрессором, ($left_2$, top_2, $width_2$, $height_2$) – координаты bbox верхней части тела человека после применения UBRSA, *width* и *height* – ширина и высота bbox тела, соответственно. Тогда:

$$left_2 = \max(0,\ left_1 + \alpha_1 \times \\ \times random(-width_1, width_1)), \tag{3.1}$$

$$top_2 = \max(0,\ top_1 + \\ + \alpha_2 \times random(-height_1, height_1)), \tag{3.2}$$

$$width_2 = \min(width,\ left_1 + width_1 + \alpha_3 \times \\ \times random(-width_1, width_1)) - left_2, \tag{3.3}$$

$$height_2 = \min(height, top_1 + height_1 + \alpha_4 \times \\ \times random(-height_1, height_1)) - top_2, \tag{3.4}$$

где α_1, α_2, α_3, α_4 – специально выбранные константы, в данной работе $\alpha_1 = \alpha_3 = \alpha_4 = 0.25$ и $\alpha_2 = 0.05$, а $random(a, b)$ – функция, возвращающая случайное целое число $x \in [a, b]$ из равномерного распределения.

3.1.4. Трекинг

В данной работе в качестве базового алгоритма сопровождения людей применяется [1]. Baseline работает в режиме реального времени и использует Венгерский алгоритм [8] для сопоставления детекций. В данной работе для связывания детекций в траектории и для создания новых траекторий применяются "векторы внешнего вида" (дескрипторы), полученные с помощью алгоритма реидентификации по верхней части тела. Для каждой траектории хранится 100 последних дескрипторов, которые были использованы для связывания детекций в траектории.

Пусть $\{D_i\}_{i=1}^{N}$ – множество дескрипторов, полученных с помощью алгоритма Re-ID, для детекций на новом кадре, а $\{Tr_i\}_{i=1}^{M}$ – множество списков дескрипторов для существующих траекторий. Тогда, введем "матрицу стоимостей" $CM \in \mathbb{R}^{N \times M}$, как:

$$c_{i,j} = \max_{tr_{jk} \in tr_j} CosineSimilarity(d_i, tr_{jk}), \\ \text{где} \quad tr_j \in \{Tr_i\}_{i=1}^{M}, \quad i \in [1, N], \quad j \in [1, M], \tag{3.5}$$

$$CM[i,j] = \begin{cases} c_{i,j}, & \textbf{если} \quad c_{i,j} \geq thr \\ 0, & \textbf{иначе} \end{cases}, \\ i \in [1, N], \quad j \in [1, M], \tag{3.6}$$

где thr – порог для Cosine Similarity, в данной работе $thr = 0.4$. Далее, для матрицы CM с помо-

Таблица 2. Видео, используемые для тестирования алгоритма оценки времени ожидания в очередях

Набор	Длительность	Формат
Queue/1	00:35:00	1920 × 1080, 25 FPS
Queue/2	00:32:00	704 × 576, 15 FPS
Queue/3	00:18:00	2688 × 1520, 24 FPS
Queue/4	00:30:00	1280 × 720, 60 FPS
Queue/5	01:24:00	1920 × 1080, 25 FPS
Queue/6	00:30:00	1280 × 720, 60 FPS

щью Венгерского алгоритма решается задача о назначениях для максимизации стоимости.

3.2. Экспериментальная оценка

3.2.1. Наборы данных

Регрессия верхней части тела. Для обучения нейросетевого регрессора из головы в верхнюю часть тела (см. раздел 3.1.2) использовался модифицированный набор данных CrowdHuman [14]. Модификации заключались в следующем:

1). используя дектор голов (см. раздел 3.1.1), для каждого изображения были найдены детекции голов;

2). используя Detectron2 [25], для каждого изображения были найдены детекции тел и сегментационные маски для них;

3). используя эвристику из [4], для каждого изображения были сопоставлены детекции голов и тел;

4). используя эвристику (см. раздел 3.1.2), для каждого изображения были найдены приблизительные позиции верхних частей тел. Эти данные применялись для создания кропов обучающей выборки нейросетевого регрессора из головы в верхнюю часть тела;

5). используя сегментационные маски для тел, соответствующие детекциям голов (согласно детекциям верхних частей тел), для каждого изображения были найдены левые и правые крайние точки верхних частей тел. Эти данные применялись для обучения нейросетевой регрессии.

Реидентификация. Для обучения нейронной сети для Re-ID использовался модифицированный публичный набор данных MSMT17 [26], в котором были объединены обучающая и тестовая выборки (полученный набор был назван MSMT17 Merged). Модификация данных заключалась в следующем: для каждого изображения была применена предложенная регрессия верхней части тела. Полученные детекции использовались для обучения нейросети для реиндентификации по верхней части тела.

В качестве тестовых выборок для алгоритма Re-ID выступили публичные каборы данных Market-1501 [27] и DukeMTMC-reID [28].

Алгоритм оценки времени ожидания в очередях. Для тестирования всего алгоритма оценки времени ожидания в очередях, предложенного в данной работе, необходимы продолжительные видеозаписи (не менее 10 минут), снятые на статичную камеру, и обладающие разметкой траекторий движения людей. Так как в публичном доступе таких видео нет, было принято решение использовать коллекцию компании ООО “Технологии видеоанализа”, состоящую из 6 видеозаписей, полученных с реальных камер видеонаблюдения в магазинах и прочих публичных местах, где возможны очереди. Это позволило приблизить тестирование предложенного алгоритма к реальным сценариям.

В таблице 2 представлена сводная информация о видеозаписях, используемых для тестирования предложенного алгоритма.

3.2.2. Метрики

Реидентификация. Для оценки качества реидентификации в данной работе использовались две метрики: $Rank_N$ и mAP. Их выбор был обусловлен тем, что они являются общепринятыми для данной задачи, и позволяют производить объективное сравнение со сторонними алгоритмами Re-ID.

Оценка времени ожидания в очередях. В данной работе предлагается новый метод оценки качества подобных алгоритмов. Он состоит в следующем:

1). тестовое видео делится на сегменты равной длины t секунд (за исключением, быть может, последнего сегмента). В данной работе $t = 300$ с;

2). для каждого сегмента вычисляются $GT_{seg} = \max_{tr \in Tr_{GT_{seg}}} L(tr)$ и $Pred_{seg} = \max_{tr \in Tr_{Pred_{seg}}} L(tr)$, где $Tr_{GT_{seg}}$ и $Tr_{Pred_{seg}}$ – множества “эталонных” и результирующих траекторий, выданных алгоритмом, соответственно. $L(tr)$ – время жизни траектории tr. Иными словами, для каждого сегмента из множества “эталонных” и результирующих траекторий выбираются траектории с максимальным временем жизни. **Важно:** время жизни трека является “сквозным”, то есть оно вычисляется в рамках всего видео, а не отдельно для каждого сегмента;

3). для каждого сегмента вычисляются абсолютная и относительная ошибки:

$$AE_{seg} = \left| Pred_{seg} - GT_{seg} \right|, \tag{3.7}$$

Таблица 3. Экспериментальное сравнение базового и предложенного алгоритмов реидентификации при сценарии, когда они обучены на одном наборе данных, а протестированы на другом. Условные обозначения: D – DukeMTMC-reID, M – Market-1501, MSMT17 Merged – MSMT17, в котором обучающая и тестовая выборки были объединены; "главные" заголовки столбцов заданы в формате "набор данных для обучения → набор данных для тестирования".

	MSMT17 → M		**MSMT17 → D**	
	$Rank_1(\uparrow)$, %	$mAP(\uparrow)$, %	$Rank_1(\uparrow)$, %	$mAP(\uparrow)$, %
Базовый алгоритм	58.8	30.3	58.5	38.3
Предложенный алгоритм	**62.0**	**32.7**	**63.6**	**42.6**

	MSMT17 Merged → M		**MSMT17 Merged → D**	
	$Rank_1(\uparrow)$, %	$mAP(\uparrow)$, %	$Rank_1(\uparrow)$, %	$mAP(\uparrow)$, %
Базовый алгоритм	65.7	37.7	66.2	47.7
Предложенный алгоритм	**69.7**	**41.3**	**70.7**	**52.4**

$$RE_{seg} = \frac{AE_{seg}}{GT_{seg}}; \tag{3.8}$$

4). для всего видео вычисляются средняя абсолютная и средняя относительная ошибки:

$$MAE = \frac{\sum\limits_{seg=1}^{N} AE_{seg}}{N}, \tag{3.9}$$

$$MRE = \frac{\sum\limits_{seg=1}^{N} RE_{seg}}{N} \times 100\%, \tag{3.10}$$

где N – количество сегментов.

3.2.3. Результаты экспериментов

Реидентификация. Ввиду того, что базовый алгоритм (см. раздел 3.1.3) был обучен и протестирован авторами для случая детекций тел, то для чистоты экспериментов, в таблице 3 приведена сравнительная оценка baseline с предложенным усовершенствованным методом Re-ID на изображениях, где люди изображены в полный рост.

Как можно видеть, предложенные модификации благоприятно влияют на качество реиндентификации при сценарии, когда обучение сети происходит на одном наборе данных, а тестирование – на другом (данный случай был выбран для тестирования из тех соображений, что в реальных сценариях приходится сталкиваться с изображениями из разных доменов). Также стоит отметить тот факт, что использование MSMT17 Merged во время обучения способствует значительному повышению метрик $Rank_N$ и mAP. Это связано с тем, что за счет объединения обучающей и тестовой выборок происходит увеличение числа данных, которые сеть "видит" во время тренировки, почти в 4 раза. Аналогичный результат переносится и на случай реидентификации по верхним частям тел.

Оценка времени ожидания в очередях. Для проведения экспериментального сравнения различных конфигураций предложенного алгоритма оценки времени ожидания в очередях было проведено 5 типов экспериментов:

- **Detectron2** – в данном эксперименте применялась реидентификация по телам и детекциям, полученным из Detectron2 [25];
- **Верхние части тел, эв.** – в данном эксперименте использовалась реидентификация по верхним частям тел и детекциям, полученным с помощью эвристики (см. раздел 3.1.2);
- **Верхние части тел, рег.** – в данном эксперименте применялась реидентификация по верхним частям тел и детекциям, полученным с помощью нейросетевой регрессии (см. раздел 3.1.2);
- **Верхние части тел, рег. и UBRSA** – данный эксперимент аналогичен предыдущему за тем исключением, что во время обучения Re-ID использовалась аугментация UBRSA (см. раздел 3.1.3);
- **MobileNetV2** – данный эксперимент аналогичен предыдущему за тем исключением, что Res2Net в алгоритме реидентификации был заменен на MobileNetV2 [29];

В таблице 4 представлены результаты проведенных экспериментов.

Эксперименты показали, что гипотеза о том, что верхние части тел людей лучше видно в очередях, чем полные тела, оказалась верной. Также большую ошибку в эксперименте, где использовались детекции тел из Detectron2, можно обосновать фактом, описанным в разделе 3.1.3.

Таблица 4. Экспериментальная оценка предложенного алгоритма оценки времени ожидания в очередях

	Кадры каждые 5 сек		**Кадры каждые 10 сек**	
	MAE (↓), с	*MRE* (↓), %	*MAE* (↓), с	*MRE* (↓), %
Detectron2	247.5	34.3	253.8	41.7
Верхние части тел, эв.	51.2	15.0	40.4	12.8
Верхние части тел, рег.	33.9	8.8	32.1	**10.7**
Верхние части тел, рег. и UBRSA	23.1	**6.3**	**29.4**	11.9
MobileNetV2	**16.3**	7.9	52.8	29.3

Сравнение результатов экспериментов с детекциями верхних частей тел, полученных с помощью эвристики и нейросетевого регрессора, показывает целесообразность применения предложенной стратегии регрессии, так как во втором случае имеется значительный прирост качества оценки времени ожидания в очередях. Это связано с тем, что в случае детекций, полученных с помощью эвристики, часто получаются слишком широкие bbox, в которые попадает фон или другие люди (рис. 3). Это влечет за собой ошибки реидентификации.

Также из проведенных экспериментов можно сделать вывод о том, что предложенная в данной работе Upper Body Random Size Augmentation для Re-ID благоприятно влияет на качество оценки времени ожидания в очередях, так как эта аугментация решает проблему, описанную в разделе 3.1.3. Решение этого момента позволяет снизить число разорванных траекторий и, как следствие, улучшить качество сопровождения.

В заключение стоит отметить последний эксперимент, в котором Res2Net был заменен на MobileNetV2 в алгоритме реидентификации. Как можно судить по метрикам, такая конфигурация предлагаемого метода вполне годится для практического применения. Более того, замеры скорости работы решения для оценки времени ожидания в очередях с Res2Net и MobileNetV2 показали, что во втором случае имеется возможность сопровождать в 4 раза больше людей на одной сцене.

4. ЗАКЛЮЧЕНИЕ

В данной работе на примере двух практически важных задач подсчета людей и оценки времени ожидания в очередях была рассмотрена проблема масштабируемости современных алгоритмов компьютерного зрения, используемых в видеоаналитике. Были предложены два алгоритма, основанные на трекинге людей в видеопотоке и решающие упомянутые ранее задачи. В обоих методах используется подход применения детекций на разреженных кадрах для снижения частоты запуска детектора. Первый метод способен эффективно сопровождать людей на коротких интервалах времени, а второй – на длинных промежутках, используя детекции на очень редких кадрах. Также в данной работе была проведена экспериментальная оценка предложенных решений, показавшая их состоятельность.

СПИСОК ЛИТЕРАТУРЫ

1. *Bewley A. et al.* Simple Online and Realtime Tracking // CoRR. 2016. V. abs/1602.00763.
2. *Wojke N., Bewley A., Paulus D.* Simple Online and Realtime Tracking with a Deep Association Metric // ArXiv e-prints. 2017.
3. *Yu F. et al.* Poi: Multiple object tracking with high performance detection and appearance feature // European Conference on Computer Vision. Springer, 2016. P. 36–42.
4. *Kuplyakov D. et al.* A distributed tracking algorithm for counting people in video by head detection // Proceedings of the 30th International Conference on Computer Graphics and Machine Vision. V. 2744. M. Jeusfeld c/o Redaktion Sun SITE, Informatik V, RWTH Aachen, 2020. P. 1–12 (CEUR Workshop Proceedings). https://doi.org/10.51130/graphicon-2020-2-3-26.
5. *Kuplyakov D., Shalnov E., Konushin A.* Further Improvement on an MCM-based Video Tracking Algorithm // Proceedings of the 26th International Conference on Computer Graphics and Vision GraphiCon'2016. 2016. P. 440–444 (GraphiCon).
6. *Bochinski E., Eiselein V., Sikora T.* High-Speed tracking-by-detection without using image information // Advanced Video and Signal Based Surveillance (AVSS), 2017 14th IEEE International Conference on. IEEE, 2017. P. 1–6.
7. *Bochinski E., Senst T., Sikora T.* Extending IOU Based Multi-Object Tracking by Visual Information // 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS). 2018. P. 1–6. https://doi.org/10.1109/AVSS.2018.8639144.
8. *Kuhn H.W.* The Hungarian method for the assignment problem // Naval Research Logistics (NRL). 1955. V. 2, № 1/2. P. 83–97.
9. *Bergmann P., Meinhardt T., Leal-Taixé L.* Tracking without bells and whistles // CoRR. 2019. V. abs/1903.05625. arXiv: 1903.05625. URL: http://arxiv.org/abs/1903.05625.

10. *Vojir T., Noskova J., Matas J.* Robust Scale-Adaptive Mean-Shift for Tracking // Image Analysis / Ed. by *J.-K. Kämäräinen, M. Koskela*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. P. 652–663. ISBN 978-3-642-38886-6.
11. *Liu W. et al.* SSD: Single Shot MultiBox Detector // Computer Vision – ECCV 2016 / Ed. by *Leibe B.* Cham: Springer International Publishing, 2016. P. 21–37. ISBN 978-3-319-46448-0.
12. *He K. et al.* Deep Residual Learning for Image Recognition. 2015. arXiv: 1512.03385 [cs.CV].
13. *Howard A.G. et al.* MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. 2017. arXiv: 1704.04861 [cs.CV].
14. *Shao S. et al.* CrowdHuman: A Benchmark for Detecting Human in a Crowd. 2018. arXiv: 1805.00123 [cs.CV].
15. *Bertinetto L. et al.* Staple: Complementary Learners for Real-Time Tracking. 2016. https://doi.org/10.1109/CVPR.2016.156
16. *Leal-Taixé L. et al.* MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking // arXiv:1504.01942 [cs]. 2015. URL: http://arxiv.org/abs/1504.01942.
17. *Milan A. et al.* MOT16: A Benchmark for Multi-Object Tracking // arXiv:1603.00831 [cs]. 2016. URL: http://arxiv.org/abs/1603.00831.
18. *Dendorfer P. et al.* MOT20: A benchmark for multi object tracking in crowded scenes. 2020. arXiv: 2003.09003 [cs.CV].
19. *Benfold B., Reid I.* Stable multi-target tracking in real-time surveillance video // CVPR 2011. 2011. P. 3457–3464. https://doi.org/10.1109/CVPR.2011.5995667
20. *Купляков Д.А. и др.* Распределенный алгоритм сопровождения людей в видео // GraphiCon 2018: труды 28-й Междунар. конф. по компьютерной графике и машинному зрению. Нац. исслед. Том. политех. ун-т. Томск, 2018. С. 208–213 (GraphiCon).
21. *Luo H. et al.* Bag of Tricks and a Strong Baseline for Deep Person Re-Identification / IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops. 2019.
22. *Gao S.-H. et al.* Res2Net: A New Multi-Scale Backbone Architecture. V. 43. Institute of Electrical, Electronics Engineers (IEEE), 02.2021. P. 652–662. DOI: URL: https://doi.org/10.1109/tpami.2019.2938758
23. *Wen Y. et al.* A Discriminative Feature Learning Approach for Deep Face Recognition // Computer Vision – ECCV 2016 / Ed. by *B. Leibe* Cham: Springer International Publishing, 2016. P. 499—515.
24. *Yan C. et al.* Beyond Triplet Loss: Person Re-identification with Fine-grained Difference-aware Pairwise Loss. 2020. arXiv: 2009.10295 [cs.CV].
25. *Wu Y. et al.* Detectron2. 2019. https://github.com/facebookresearch/detectron2.
26. *Wei L. et al.* Person Transfer GAN to Bridge Domain Gap for Person Re-Identification. 2018. arXiv: 1711.08565 [cs.CV].
27. *Zheng L. et al.* Scalable Person Re-identification: A Benchmark // IEEE International Conference on Computer Vision (ICCV). 2015. P. 1116–1124. https://doi.org/10.1109/ICCV.2015.133.
28. *Ristani E. et al.* Performance Measures and a Data Set for Multi-Target, Multi-Camera Tracking. 2016. arXiv: 1609.01775 [cs.CV].
29. *Sandler M. et al.* MobileNetV2: Inverted Residuals and Linear Bottlenecks. 2019. arXiv: 1801.04381 [cs.CV].