Разработка и исследование алгоритма отслеживания объектов с использованием нескольких камер

В.Р. Советников[1], П.Ю. Якимов[1,2]

[1]Самарский национальный исследовательский университет им. академика С.П. Королева, Московское шоссе 34А, Самара, Россия, 443086
[2]Институт систем обработки изображений РАН - филиал ФНИЦ «Кристаллография и фотоника» РАН, Молодогвардейская 151, Самара, Россия, 443001

Аннотация. Классификация объектов в видеопотоке и их отслеживание завоёвывает всё большую популярность в настоящее время. В данной статье рассматриваются способы детектирования людей на видео и дальнейшее их отслеживание. Описываются наиболее актуальные средства, в том числе использующие нейронные сети. А также исследуется концепция объединения нескольких видеопотоков в единую сеть для слежения за объектами.

1. Введение

В последние годы за счёт значительного повышения качества и снижения стоимости оборудования системы видеонаблюдения получили огромное распространение. Closed Circuit Television (система телевидения замкнутого контура, CCTV) используются сегодня в самых разных сферах: в учебных заведениях, в торговых центрах, на улицах и так далее. В основном они используются для обеспечения безопасности или сбора каких-либо данных. Так, например, в Москве в 2017 году более 160 000 камер было подключено к системе распознавания лиц [1]. Китай при этом, являясь лидером по темпам наращивания цифровых средств слежения, отчитался об установке свыше 20 миллионов камер, объединенных в одну сеть [2]. Совершенно очевидно, что тенденция к росту количества сенсоров будет сохраняться ближайшее время. Таким образом огромные массивы данных для обработки также будут только расти, что влечёт за собой необходимость разработки новых методов анализа видеозаписей с максимальной эффективностью и производительностью.

Применение CCTV возможно не только для решения задач безопасности, но также и в задачах маркетинга или в концепциях smart home [3] и smart cities [4]. Так, например, информация об удовлетворённости клиентов при посещении торговых площадей или карта интенсивности посещаемости тех или иных торговых прилавков является особо ценной для владельцев малого и среднего бизнеса, обработка которой позволит увеличить оборот и минимизировать издержки. Также применение CCTV в домашних условиях может позволить, к примеру, настроить автоматическое кормление домашних животных или реализовать концепцию автоматизированного домашнего хозяйства.

Для решения всех вышеперечисленных задач одним из базовых алгоритмов является отслеживание объектов на видео. При этом, учитывая ограниченную область видимости одной, камеры в CCTV обычно используются несколько камер для охвата как можно большей части

наблюдаемой области. В такой постановке актуальной становится задача отслеживания объекта (например, человека) не только в видеопотоке одной камеры, но и при переходе между зонами ответственности различных устройств.
В настоящей работе описан алгоритм отслеживания объекта при переходе из видеопотока одной камеры в зону видимости другой с сохранением идентификационного номера объекта.

2. CNN для детектирования объектов в реальном времени

Решение указанной задачи можно разделить на три этапа: детектирование объекта, отслеживание объекта с течением времени, идентификация объекта в видеопотоках с нескольких камер.
Одним из вариантов решения задачи детектирования основывается на принципе вычисления разности двух изображений (текущего кадра и модели фона) и применения к разности порогового фильтра, в результате чего получается бинарная маска, обозначающая объекты, отсутствующие на кадре фона. Этого можно добиться с помощью библиотеки OpenCV [5]. В качестве достоинств этого подхода можно отметить высокую скорость работы и большое число найденных объектов.
Однако в реальных условиях такой подход приводит к следующим проблемам: в кадре появляются движущиеся объекты, которые не являются людьми: отражения, открывающиеся двери. Также этот способ плохо работает в местах большого скопления людей, где несколько людей сливаются друг с другом [6].
Вторым же вариантом является использование нейронных сетей для распознавания объектов в видеопотоке. К примеру, система отслеживания объектов YOLO [7] справляется с этой задачей с высокой точностью. В тестах использовалась одна из реализаций YOLO на Tensorflow[8] и OpenCV. Этот проект с открытым исходным кодом под названием darkflow [9]. В результате был получен весьма точный результат (гораздо более точный, чем в первом варианте). Однако требуются огромные ресурсы, для быстрой обработки видео этим способом. Так в моём случае на обработку видео длительностью 5 секунд на моём CPU AMD FX 6300, составило примерно 2 минуты, на GPU GeForce GTX 660 около 1 минуты.
Слежение за найденными объектами реализуется с помощью алгоритма Калмана [10], который позволяет получить вероятное положения объекта в следующем кадре, и основанных на нём алгоритмах и библиотеках (например, SORT [11]). Эти средства позволяют продолжать слежения за объектом, даже в том случае, если его не было на нескольких кадрах (например, в случае перекрытия препятствием, или плохой точности распознавания). Если найденная на карте область, содержащая фигуру человека не сопоставлена ни с одной из текущих траекторий, она становится началом новой траектории. Траектория завершается, если на протяжении некоторой последовательности кадров, к траектории объекта не добавляются новые положения.
Сейчас же, благодаря развитию нейронных сетей, появилась библиотека Deep SORT [17], которая является развитием SORT с нейронными сетями, которые значительно повышают точность предсказания.
Наибольшие трудности возникают при попытке определить, является ли объект, который мы видим в двух видеопотоках с пересекающейся зоной покрытия, одним и тем же. Мы не знаем, перешел ли объект из первой зоны во вторую, или во второй зоне появился новый объект. Поэтому нам придётся использовать нейронные сети, которые будут детектировать объект по его сохранённому изображению в предыдущем кадре. Либо строить карту и особым образом указывать на ней зоны переходов.

2.1. Сверточные нейронные сети

Распознавание и классификация изображений достаточно успешно осуществляются с помощью сверточных нейронных сетей. Нейронная сеть – это некоторая математическая модель, которая состоит из соединенных между собой искусственных нейронов [12]. Сеть принимает в качестве входных данных вектор признаков, затем последовательно пропускает их через слои сети. На

выходе получаются вероятности принадлежности объекта к заданным классам. Обычно нейронная сеть оперирует числовыми, а не символьными величинами.
На рисунке 1 представлена схема искусственного нейрона [13]. Он принимает на вход параметры $x_1, x_2, x_3, \dots, x_n$, которые являются либо исходными данными, либо выходными параметрами других нейронов. У каждого параметра есть вес, $w_{1j}, w_{2j}, w_{3j}, \dots, w_{nj}$, на который умножается значение параметра. Затем взвешенные параметры суммируются с помощью некоторой функции $transfer\ function$. Полученное значение отправляется в $activation\ function$, которая после вычисления результата принимает решение, передавать ли сигнал следующему нейрону. Выходное значение поступит в качестве одного из параметров в другой нейрон.

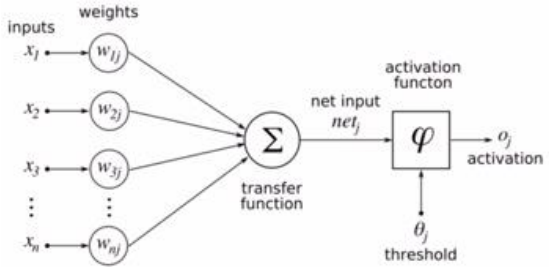


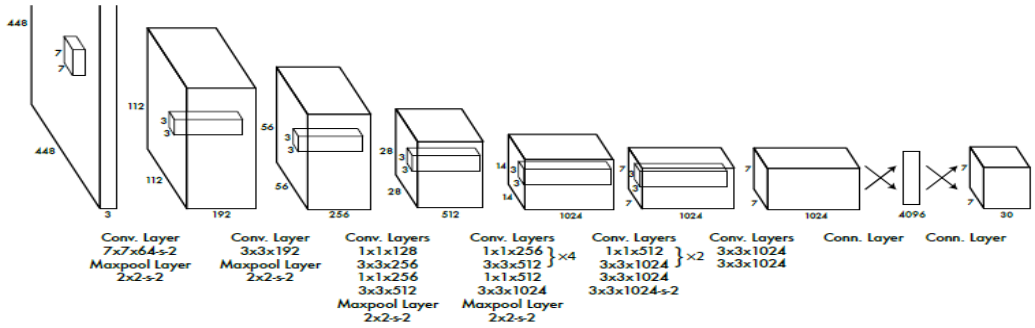
Рисунок 1. Схема искусственного нейрона другой нейрон.

В настоящее время для обработки изображений наиболее эффективны сверточные нейронные сети. На вход свёрточной нейронной сети подается двумерное изображение, которое затем обрабатывается свёрточными слоями. Свёрточные слои преобразуют фрагменты изображения в карту признаков.
В нашей задаче необходимы свёрточные нейронные сети для максимально точного детектирования объектов.

2.2. YOLO
YOLO CNN – это свёрточная нейронная сеть, позволяющая детектировать и классифицировать объекты в виде обрамляющих прямоугольников. YOLO работает по принципу Single Shot. Это значит, что архитектура сети устроена таким образом, что за один проход кадра, на нём детектируются все объекты.
На рисунке 2 представлена архитектура YOLO CNN. На вход YOLO подается трёхканальное изображение, у которого меняется размер до 448x448, над полученным изображением проводятся преобразования. Первое преобразование заключается в прогоне изображения через часть модифицированной архитектуры GoogLeNet. После этого преобразования получается feature maps размером 14x14x1024. Далее применяются две конволюции. После второй конволюции размерность уменьшается до 7x7x1024. Далее проводится ещё одна конволюция. Результат дважды прогоняется через полносвязный слой, изменяется до размерности 1470x1 и в итоге трансформируется в тензор размером 7x7x30. К полученному тензору применяется процедура детектирования, на выходе которой получается результирующее детектирование. Тензор представляет собой отображение сетки 7x7 на изображении. 30 значений несут информацию о ячейке: 10 значений для двух возможных рамок, 20 значений отношения к каждому из 20 доступных классов. Вся эта информация фильтруется, отфильтрованные данные отображаются.

2.3. Построение двумерной карты.
Следующим шагом будет построение двумерной карты. Для этого нам понадобятся координаты найденных нами объектов, например, записанные в файл, и кадр за кадром будем переносить их на нашу «карту». Примеры построения карты по известным координатам можно найти в интернете [14]. Однако, стоит понимать, что перенесенные нами координаты не всегда соответствуют (в большинстве случаев не соответствуют) реальным координатам объектов. Это связано с тем, что расстояние от камеры до объекта и угол наклона камеры варьируются в

каждом конкретном случае. Поэтому нам будет необходимо воспользоваться проективным преобразованием [15], т.е. изменить каждую координату в зависимости от угла наклона камеры, высоты на которой она расположена и расстояния до объекта. В результате мы



получили двумерную карту с координатами объектов. Пример такой карты, построенный по имеющимся координатам из файла в произвольный момент времени, показан на рисунке 3.

Рисунок 2. Архитектура YOLO CNN.

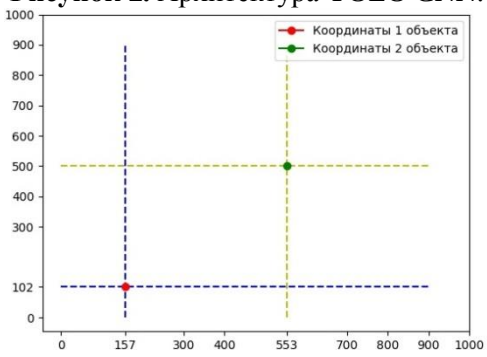


Рисунок 3. Пример карты.

В дальнейшем мы сможем использовать полученную карту для связи нескольких видеопотоков. К примеру, если у нас есть две статичные камеры, с пересекающейся зоной покрытия, мы сможем отметить зоны пересечения на картах. Далее в случае входа объекта в эту зону на одной карте и одновременном появлении на другой, будет вызываться метод, передающий ID объекта и слежение за ним другой камере. Концептуальная модель двух карт с отмеченными зонами пересечения представлены на рисунках 4а и 4б.

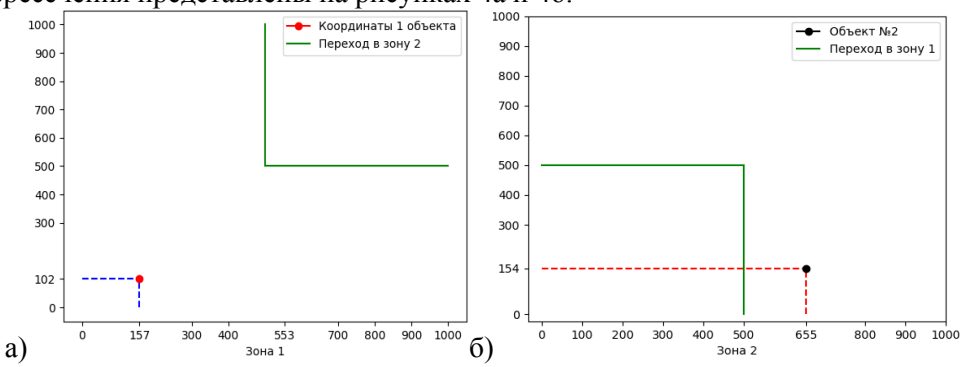


Рисунок 4. Пример карт с пересекающейся зоной.

3. Реализация информационной технологии

3.1. YOLO

В YOLO CNN предусмотрена возможность компиляции и запуска сети с помощью CPU, GPU, CUDA, CUDNN, OpenCV, OpenMP и их комбинаций. На рисунке 5 представлен результат работы YOLO на тестовом изображении.

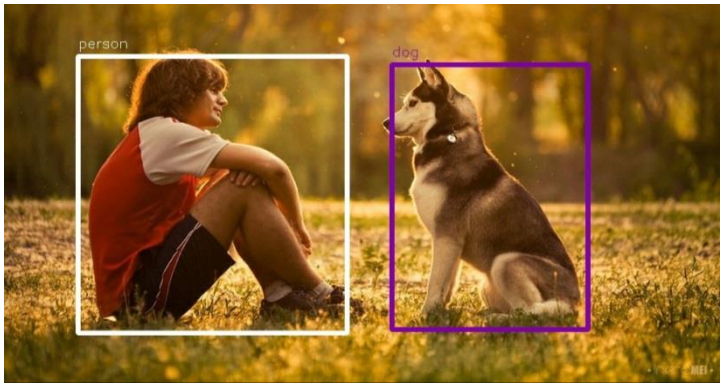


Рисунок 5. Результат работы YOLO.

Для запуска YOLO необходимы файл конфигурации (пример yolo.cfg), файл весов для всех необходимых классов (пример yolo.weights) и медиа-файл (изображение или видео, также есть возможность запуска видеопотока с веб-камеры). Пример команды:
./darknet detect cfg/yolo.cfg yolo.weights data/image.png
Нейронная сеть имеет большую базу уже обученных классов, на которых можно протестировать все возможности YOLO, а также производительность аппаратного обеспечения. Но нейронную сеть можно обучить любому классу изображений, если грамотно подобрать исходные данные. Для обучения YOLO необходимо большое количество изображений. К каждому изображению должен прилагаться текстовый файл с размеченными регионами тренируемого класса объектов. Также необходим файл с начальными весами и файл с системной информацией, в котором указаны пути к изображениям и путь, по которому будут записываться точки восстановления. Точки восстановления представляют из себя файлы весов на определенном шаге обучения. Файлы весов записываются в постоянную память каждые 100 итераций, что позволяет в любое время прервать обучение, а затем продолжить с последним полученным весом. Чем дольше учится сеть, тем качественнее будет детектирование. Команда для обучения YOLO выглядит следующим образом:
./darknet detector train cfg/voc.data cfg/yolo-voc.cfg darknet.conv.23

3.2. Реализация алгоритма детектирования и отслеживания объектов
Для начала нам придётся выбрать способ детектирования объектов. Можно использовать программный комплекс Darkflow, который включает в себя и openCV, YOLO и tensorflow.
Darkflow отлично справляется с задачей по детектированию объектов. Мной были проведены тестовые запуски данного программного комплекса. Запуск проводился на одном из тестовых видео, взятых с MOT challenge [16], фрагмент которого показан на рисунке 6.
Запуск производился с помощью команды
./flow --model cfg/yolo.cfg --load bin/yolo.weights --demo 1598.mp4 –saveVideo

Рисунок 6. Фрагмент видео до обработки.

На первый взгляд кажется, что на фрагменте 5 человек. Но в процессе работы программа обнаружила на этом изображении 10 человек, это видно на рисунке 7.

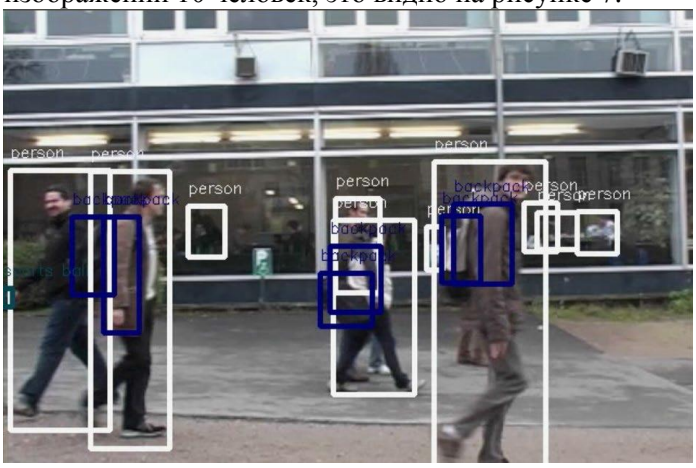


Рисунок 7. Фрагмент видео после *обработки.*

При этом программа сработала достаточно точно и выделила людей без ошибок, несмотря на то, что некоторые из них трудно различить из –за расстояния и шумов в виде отражений на стекле.

Также Darkflow позволяет получить координаты объектов на видео и вывести их в формате json или в другом, удобном для нас формате. Однако эта функция не работает для видео целиком, а лишь для отдельных фреймов. Поэтому нам придётся сначала разделить видео на кадры. Это можно сделать с помощью библиотеки OpenCV [18].

```
import numpy as np
import cv2
from darkflow.net.build import TFNet
cap = cv2.VideoCapture("/home/viktor/darkflow/1598.avi")
while(True):
   ret, frame = cap.read()
   //Необходимые операции над фреймом.
cap.release()
cv2.destroyAllWindows()
```

Далее с помощью небольшого кода (на языке Python)

```
from darkflow.net.build import TFNet
import cv2
options = {"model": "cfg/yolo.cfg", "load": "bin/yolo.weights", "threshold": 0.1}
tfnet = TFNet(options)
imgcv = cv2.imread("./sample_img/sample_dog.jpg")//или frame в нашем случае
result = tfnet.return_predict(imgcv)
print(result)
```

сможем получить координаты всех объектов на видео и, например, сохранить их в файл.

Отличить же одного человека от другого поможет программный комплекс Deep SORT, который добавит каждому объекту на видео свой уникальный ID, что позволит нам отслеживать конкретные объекты/конкретных людей.

В итоге мы имеем обработанный видеопоток с отмеченными на нём людьми, каждому из которых присвоен свой уникальный ID. На рисунке 8 показан результат запуска darkflow с использование Deep SORT.

Также мы имеем файл с координатами объектов на каждом кадре, по которому в последствии можно будет построить двумерную карту (подобную показанной на рисунке 3) и, при необходимости, связать её с другой подобной картой. В данной момент, эта часть технологии находится в разработке.

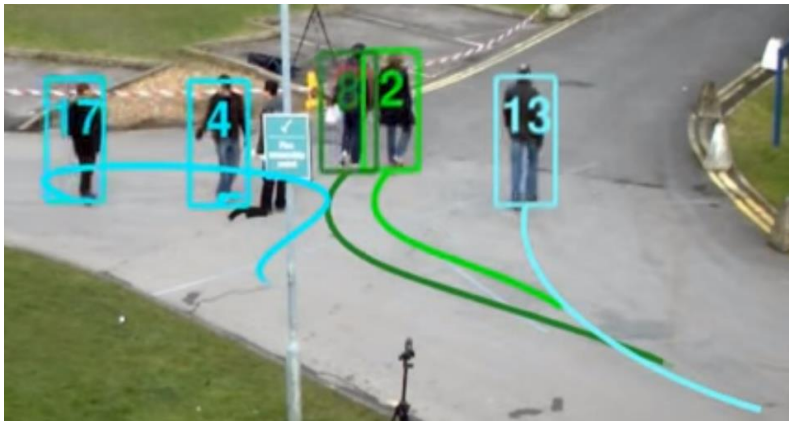


Рисунок 8. Результат работы darkflow+Deep SORT.

4. Заключение

В статье описывается реализация работы программного комплекса darkflow, включающего в себя openCV, YOLO и tensorflow. Нейронная сеть успешно справляется с задачей детектирования людей на видео. Также рассмотрены варианты улучшения точности слежения с помощью Deep SORT. А концепция построения двумерной карты с координатами объектов открывает перспективы для дальнейших исследований и построения алгоритмов на их основе.

Это позволит решить многие задачи, в том числе слежение за человеком в здании, при переходе из комнаты в комнату, при переходе между этажами. Также есть вариант использования в маркетинговых целях, например, сопровождение посетителя торгового центра для сбора каких – либо статистических данных. Одним же из приоритетных вариантом использования, конечно, является отслеживание объектов в целях безопасности, предотвращения несанкционированного доступа.

Стоит отметить, что разработанная технология не требует больших изменений в уже существующей инфраструктуре, главное лишь иметь зоны пересечения камер. Также система легко масштабируема, т.е. добавление новой камеры в сеть не составит много времени.

5. Литература

[1] Электронное периодическое издание «Ведомости» (Vedomosti) [Электронный ресурс]. – Режим доступа: https://www.vedomosti.ru/politics/news/2017/09/28/735806-v-moskve.
[2] The Jamestown Foundation [Electronic resource]. – Access mode: https://jamestown.org/program/chinas-domestic-security-spending-analysis-available-data/.
[3] Vivint.SmartHome [Electronic resource]. – Access mode: https://www.vivint.com/resources/article/what-is-a-smart-home.
[4] Wikipedia. The free Encyclopedia [Electronic resource]. – Access mode: https://en.wikipedia.org/wiki/Smart_city.
[5] OpenCV: Open Source Computer Vision Library [Electronic resource]. – Access mode: https://opencv.org/.
[6] Научная электронная библиотека «Киберленинка» [Электронный ресурса]. – Режим доступа: https://cyberleninka.ru/article/n/sistema-identifikatsii-i-otslezhivaniya-dlya-analiza-povedeniya-posetiteley-roznichnogo-magazina-na-osnove-videodannyh.
[7] YOLO: Real-Time Object Detection [Electronic resource]. – Access mode: https://pjreddie.com/darknet/yolo.
[8] Tensorflow [Electronic resource]. – Access mode: https://www.tensorflow.org/.
[9] Darkflow: Real-time object detection and classification [Electronic resource]. – Access mode: https://github.com/thtrieu/darkflow.
[10] Wikipedia. The free Encyclopedia [Electronic resource]. – Access mode: https://en.wikipedia.org/wiki/Kalman_filter.

[11] SORT: simple online and realtime tracking [Electronic resource]. – Access mode: https://arxiv.org/pdf/1602.00763.pdf.
[12] Николенко, С.И. Глубокое обучение / С.И. Николенко, А.А. Кадурин, Е.О. Архангельская // С.-Пб: Издательский дом "Питер", 2017. – 480 с.
[13] Заенцев, Е.В. Нейронные сети: основные модели. – Воронеж, 1999. – 76 с.
[14] OpenCV guides [Electronic resource]. – Access mode: https://github.com/MicrocontrollersAndMore?tab=repositories.
[15] Wikipedia. The free Encyclopedia [Electronic resource]. – Access mode: https://ru.wikipedia.org/wiki/Проективное_преобразование.
[16] MOTChallenge: The Multiple Object Tracking Benchmark [Electronic resource]. – Access mode: https://motchallenge.net/.
[17] Simple Online and Realtime Tracking with a Deep Association Metric (Deep SORT) [Electronic resource]. – Access mode: https://github.com/nwojke/deep_sort.
[18] Документация по библиотеке OpenCV [Электронный ресурс]. – Режим доступа: https://docs.opencv.org/3.1.0/.

Development and research of multiple object tracking algorithm using multiple cameras

V.R. Sowetnikov[1], P.Y. Yakimov[1,2]

[1]Samara National Research University, Moskovskoe Shosse 34A, Samara, Russia, 443086
[2]Image Processing Systems Institute of RAS - Branch of the FSRC "Crystallography and Photonics" RAS, Molodogvardejskaya street 151, Samara, Russia, 443001

Abstract. The classification of objects in a video stream and their tracking has gained immense popularity at present. The article considers ways of detecting people on video and their further tracking. The article describes the most relevant tools, including those using neural networks. Also explores the concept of combining multiple video streams into a single network to track objects.