РЕФЕРАТ

Магистерская диссертация содержит 40 страниц, 24 рисунка. Список использованных источников содержит 17 позиций.

МАШИННОЕ ОБУЧЕНИЕ, ГЛУБОКОЕ ОБУЧЕНИ, АЛГОРИТМЫ СЛЕЖЕНИЯ ЗА ОБЪЕКТОМ, ГЛУБОКИЕ КОНВОЛЮЦИОННЫЕ НЕЙРОННЫЕ СЕТИ, ОБРАБОТКА ИЗОБРАЖЕНИЙ, ОБРАБОТКА ВИДЕОПОСЛЕДОВАТЕЛЬНОСТЕЙ

Магистерская диссертация посвящена разработке алгоритма слежения за объектами малых размеров по видеопоследовательностям с использованием глубоких ковнволюционных нейронных сетей. В качестве входных данный алгоритм принимает изображение, являющееся первым кадром видеопоследовательности, с выделенным ограничивающей рамкой целевым объектом и набор изображений, являющимися последующими кадрами видеопоследовательности. В результате работы алгоритма на всех последующих кадрах видеопоследовательности будет обнаружен целевой объект.

Для решения поставленной задачи был разработан алгоритм на основе алгоритмов SiamMask и DETR, который комбинирует в себе их лучшие качества. Разработанный алгоритм показывает высокую точность слежения за объектами малых размеров на видеопоследовательностях, а также инвариантен к размерам входных изображений.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время глубокие конволюционные нейронные сети (ГКНС, CNN) является активно развивающимся направлением в области компьютерных технологий. ГКНС получили широкое распространение в области компьютерного зрения начиная с 2012 года. В тот год произошло знаменательное событие в истории компьютерного зрения – с помощью алгоритма базировавшемся на ГКНС были получены наилучшие результаты на ежегодном конкурсе по распознаванию изображений ImageNet (ILSVRC) [1]. Технический прогресс в области графических ускорителей сделал возможным практическое применение ГКНС большей глубины с более сложной архитектурой.

Задачи компьютерного зрения являются частью области применения нейронных сетей. Одной из важных задач компьютерного зрения является отслеживание объекта на видеопоследовательности. Примером практического применения отслеживания целевого объекта могут служить различные системы безопасности, видеонаблюдения, наведения. В настоящее время классические алгоритмы обработки изображений не могут сравниться по точности с алгоритмами, основанными на нейронных сетях, поэтому для решения задачи отслеживания целевого объекта, в основном, применяются нейросетевые методы, которые получили широкое распространение благодаря своей точности и скорости работы. К недостаткам нейросетевых методов можно отнести необходимость большой базы данных для обучения, а также скорость обучения, но в настоящее время существуют готовые, размеченные базы данных для глубокого обучения, а также графические ускорители, благодаря которым обучение нейронных сетей занимает несколько часов. Учитывая это, применение нейронных сетей для задачи отслеживания объекта является самым оптимальным решением.

Для решения задачи отслеживания целевого объекта на видеопоследовательности существуют различные нейросетевые алгоритмы,

например, MOTS [2] и SiamMask [3]. MOTS является достаточно точным алгоритмом, однако скорость его работы очень мала, что существенно сужает область его применения на практике. Также данный алгоритм работает по заранее известным классам объектов и для отслеживания целевого объекта неизвестного ранее класса необходимо дообучение нейронной сети. SiamMask является более перспективным методом для отслеживания объекта на видеопоследовательности, однако его недостатком является необходимость предобработки входных данных и постобработки выходных данных. Устранение данного недостатка является целью данной выпускной квалификационной работы.

В настоящей выпускной квалификационной работе предлагается метод решения задачи слежения за целевым объектом на видеопоследовательности с помощью комбинированного метода, который объединяет алгоритмы SiamMask и DETR [4]. Оригинальный алгоритм DETR применяется для задачи детекции и использует архитектуру трансформер в качестве детектирующей части. Благодаря этому можно исключить необходимость предобработки входных данных и постобработки выходных данных. Так же, благодаря такой комбинации достигается высокая скорость обучения и точность работы, а также простота предложенного подхода.

Предложенный подход совмещает два современных алгоритма для отслеживания и детектирования объекта, комбинируя их лучшие части. Благодаря этому становится возможным получить хорошую точность и скорость работы при сравнительно небольших затратах времени на обучение.

ОСНОВНАЯ ЧАСТЬ

1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

1.1. Отслеживание целевого объекта на видеопоследовательности

1.1.1. Постановка задачи

Целью данной дипломной работы является создания алгоритма отслеживания целевого объекта на видеопоследовательности. В разработке алгоритма отслеживания целевого объекта на видеопоследовательности можно выделить несколько задач:

1) Разобрать устройство и общие принципы работы ГКНС;
2) Изучить базовые архитектуры ГКНС;
3) Изучить основные алгоритмы отслеживания целевого объекта на видеопоследовательности с использованием ГКНС;
4) Выбрать подходящую архитектуру нейронной сети;
5) Выбрать подходящий алгоритм отслеживания целевого объекта на видеопоследовательности на основе ГКНС;
6) Подготовить набор данных для обучения для ГНКС;
7) Обучить ГКНС;
8) Провести тестирование ГКНС на тестовом наборе данных;
9) Сделать выводы по проделанной работе;

В результате выполнения данной дипломной работы будет разработан алгоритм отслеживания целевого объекта на видеопоследовательности. После проведения тестирования можно будет сделать выводы о целесообразности использования архитектуры трансформер для задачи отслеживания целевого объекта на видеопоследовательности.

1.1.2. Актуальность проблемы

На сегодняшний день область компьютерного зрения развивается очень быстро. Одной из популярных задач компьютерного зрения является отслеживание целевого объекта на видеопоследовательности. Для

отслеживания целевого объекта необходим быстрый и точный метод детекции, который, благодаря модернизации, позволит находить целевой объект на видеопоследовательности.

В настоящее время существуют различные методы поиска объектов на изображениях и видеопоследовательностях, но они имеют свои недостатки. Некоторые методы слишком медленные, некоторые недостаточно точные. Основной целью данной дипломной работы является создание автоматизированной системы для отслеживания целевого объекта на видеопоследовательности. На сегодняшний день самым оптимальным решением данной задачи является обучение нейронной сети.

В связи с этим возникает необходимость в разработке нейронный сети, которая сможет производить отслеживание целевого объекта на видеопоследовательности с высокой точностью и высокой скоростью. Такая нейронная сеть будет справляться с поставленной задачей лучше любых классических алгоритмов обработки изображений и видеопоследовательностей.

В настоящее время существует проблема качества, скорости, а также сложности предобработки и постобработки входных и выходных данных для отслеживания целевого объекта на видеопоследовательности. Существуют различные методы, которые позволяют устранить данные недостатки. В данной дипломной работе будет рассматриваться метод, суть которого заключается в использовании объединения сиамской сети и архитектуры трансформер в качестве головы отслеживания целевого объекта. Такое преобразование может увеличить качество отслеживания целевого объекта и существенно упростить процесс предобработки и постобработки входных и выходных данных.

1.2. Устройство ГКНС и методы взаимодействия с ними

1.2.1. Основные слои ГКНС

Глубокие конволюционные сети [5] состоят из повторяющихся блоков – слоев. Нейроны в рамках слоя не связаны между собой, но связаны друг с другом общими входными данными. В состав ГКНС обычно входят следующие слои: конволюционные, слои субдискретизации, слои нормализации, полносвязные слои. Пример работы конволюционной нейронной сети показан на рис. 1.1:

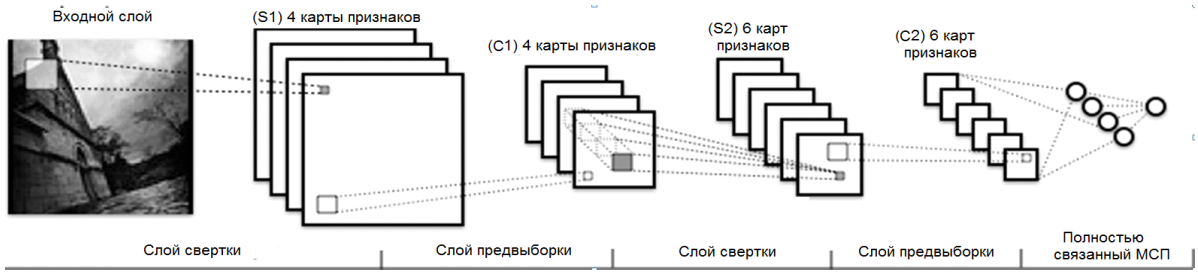


Рис. 1.1 Пример работы конволюционной нейронной сети

1) Конволюционный слой

Первым слоем глубокой конволюционной нейронной сети обычно является конволюционный слой. Основной идеей конволюционных слоев является поиск некоторого признака на предыдущем слое. Для данной задачи используется набор нейронов, каждый из которых обрабатывает одну возможную зону искомого признака. Для удобства восприятия в конволюционном слое нейроны объединены в срезы (фильтры). В рамках среза все нейроны имеют один и тот же набор ненулевых весов (ядро фильтра), отличающийся лишь сдвигом, соответствующим положению кодируемым нейроном. Каждый нейрон кодирует определенное положение на дискретной плоскости при обработке изображений. Результатом обработки входного изображения конволюционным слоем является матрица, номер и столбец которой соответствует положению, кодируемому нейроном, а значение – выходному сигналу. Работу среза можно представить, как обработку скользящим окном входных данных одним нейроном. Выходные данные

конволюционного слоя обычно представляю в виде прямоугольного параллелепипеда, который в свою очередь так же делится по срезам.

После конволюционного слоя необходима нелинейность – функция активации. В качестве функции активации часто используется ReLU, схема работы которого изображена на рис. 1.2:

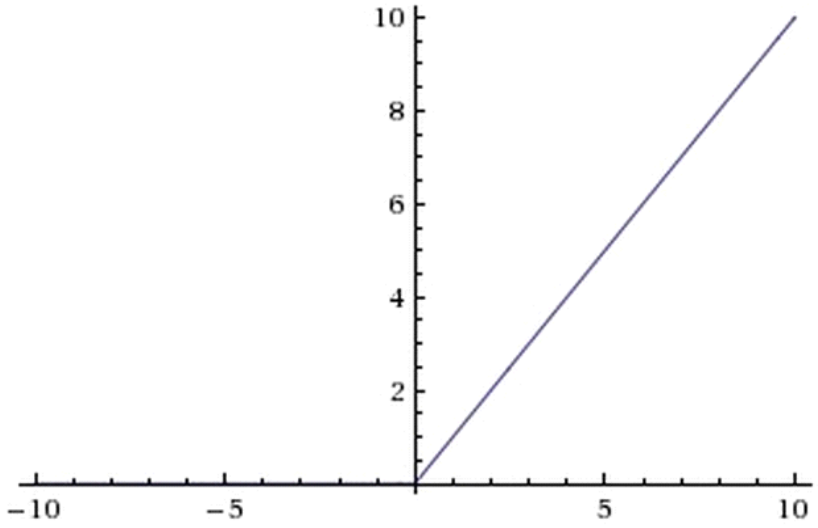


Рис. 1.2 Функция активации ReLu

2) Слой пулинга (слой субдискритизации)

Слой пулинга необходим для снижения размерности входных данных и обеспечения многомасштабности обработки входных данных конволюционной нейронный сети. В большинстве случаев снижение размерности проводят для каждого среза входных данных независимо.

Одной из наиболее популярной разновидностью пулинга является max – pooling. Операция max – pooling представляет собой операцию уменьшения входного слоя путем объединения соседних элементов в один, значение которого будет соответствовать максимальному элементу. Схема работы max – polling представлена на рис. 1.3:

Рис. 1.3 Max – pooling

3) Слой нормализации

Слой нормализации активно используется в большом количестве конволюционных нейронных сетей. Данный слой предназначен для повышения устойчивости конволюционной нейронной сети к помехам и для усиления слабых сигналов. За счет использования данного слоя достигается увеличение качества, а также скорости обучения ГКНС. Пример значимости использования нормализации приведен на рис. 1.4:

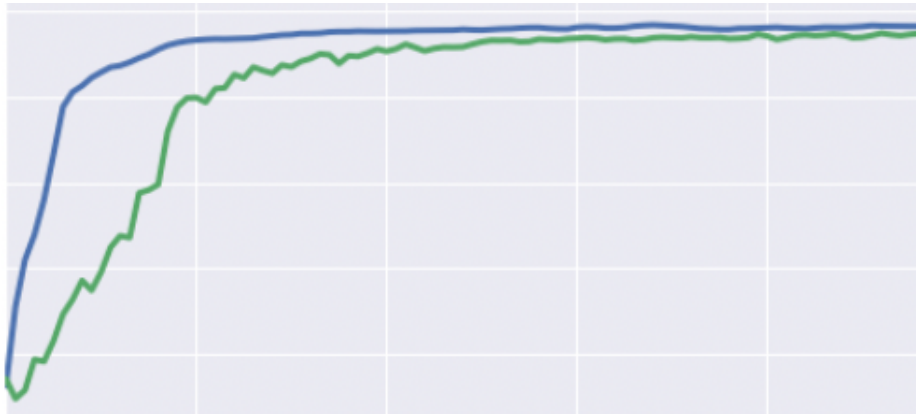


Рис. 1.4 Обучение ГКНС с использование нормализации (верхний график) и без нормализации (нижний график)

Существуют разных способы нормализации. Некоторые из них использует методы нормализации яркостей из обработки изображений, которые применяются независимо для каждого среза.

Одним из самых популярных способов нормализации является батч нормализация. Она реализована во всех современных фреймворках глубокого обучения. Алгоритм батч нормализации описывается формулой 1.1:

$$y = \frac{x - \mathrm{E}[x]}{\sqrt{\mathrm{Var}[x] + \epsilon}} * \gamma + \beta \tag{1.1}$$

где:

y – результат нормализации;

x – входная карта признаков;

E – матожидание;

Var – стреднеквадратическое отклонение;

ε - малая константа, для избегания деления на 0;

γ и β - обучаемые параметры.

4) Полносвязный слой

Полносвязный слой конволюционной нейронной сети является слоем классического многослойного персептрона, где на входы всех нейронов текущего слоя попадают выходы всех нейронов предыдущего слоя. Данный слой в основном формирует итоговый вектор признаков, например в задаче классификации. Полносвязный слой может быть один, или же их может быть несколько. Обычно на вход первого полносвязного слоя попадают данные с выхода последнего конволюционного слоя. Схема работы полносвязного слоя представлена на рис. 1.5:

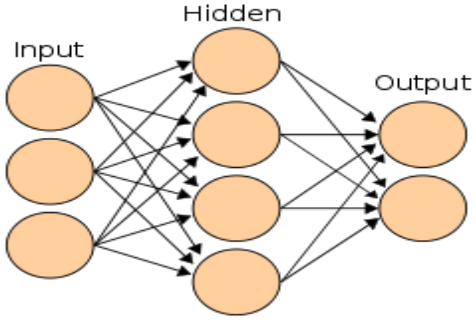


Рис. 1.5 Схема работы полносвязного слоя

1.2.2. Базовые архитектуры нейронных сетей

1) VGG

Сети VGG [6] появились в связи с исследованием проблемы затухания градиентов. Экспериментально было показано, что сети классической архитектуры, имеющие глубину более 19 слоев, не могут учиться методом обратного распространения ошибки. В результате появился класс сетей VGG, который долгое время был лидером по качеству распознавания в самых различных задачах. Другой особенностью этих сетей является использованием фильтров малого размера – 3x3. Структура сети VGG-16 показана на рис. 1.6:

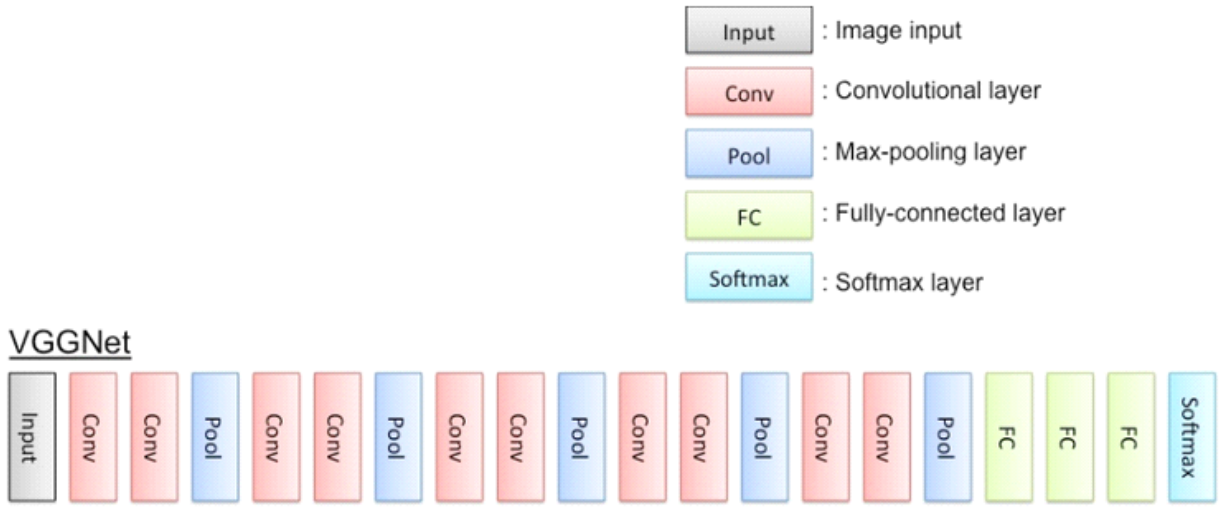


Рис. 1.6 Архитектура VGG – 16

2) ResNet

В настоящее время прослеживается тенденция к увеличению слоев в архитектуре глубоких конволюционных сетей (рис. 1.7).

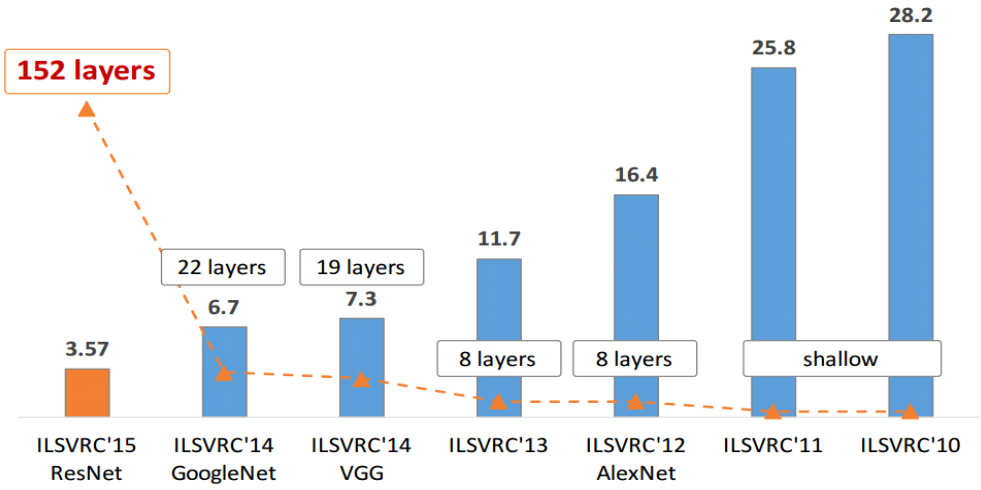


Рис. 1.7 Революция применения глубоких нейронных в задаче распознавания на примере конкурса ImageNet: уменьшение количества top-5 ошибок и рост глубины сети (количество слоев)

Авторы [7] смогли решить проблему затухания градиента (degradation problem) для сетей с большим количеством слоев, что позволило увеличивать качество обучения ГКНС при добавлении новых слоев. Сеть стала первой подобной так называемой резидуальной сетью и получила название ResNet (Deep Residual Network). Данная нейронная сеть представлена в нескольких модификациях: ResNet-18, ResNet-34, ResNet-50, ResNet-101 и ResNet-152.

Основная идея резидуальных сетей строится на том, что нейронная сеть может аппроксимировать почти любую функцию, например, некоторую сложную функцию H(x). Тогда справедливо, что такая сеть легко выучит остаточную (residual) функцию, представленную функцией 1.2:

$$F(x) = H(x) - x \tag{1.2}$$

Тогда первоначальная целевая функция будет выражаться формулой 1.3:

$$H(x) = F(x) + x \tag{1.3}$$

Если взять какую-либо неглубокую сеть одной из известных топологий и добавить к ней еще несколько слоев, то требуется, чтобы глубокая сеть давала результат как минимум не хуже своего неглубокого аналога. Проблема

деградации подразумевает, что сложная нелинейная функция F(x), полученная добавлением нескольких слоев, должна выучить тождественное преобразование, в случае если на предыдущих слоях был достигнут предел качества, однако, этого не происходит.

Идея построения сети подобной топологии может быть реализована при помощи добавления блоков с прямыми связями (резидуальными, shortcut-соединение), так, что сеть фактически решает сама насколько глубокой ей быть. Данная технология (рис. 1.8 и рис. 1.9) позволила создавать и обучать сети практически любой глубины (вплоть до 1000 слоев и более).

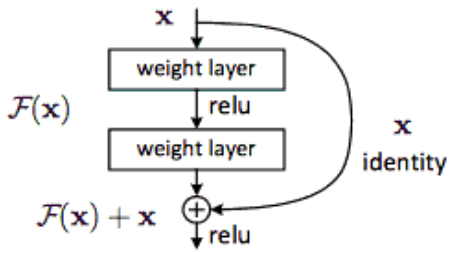


Рис. 1.8 Резидидуальный блок

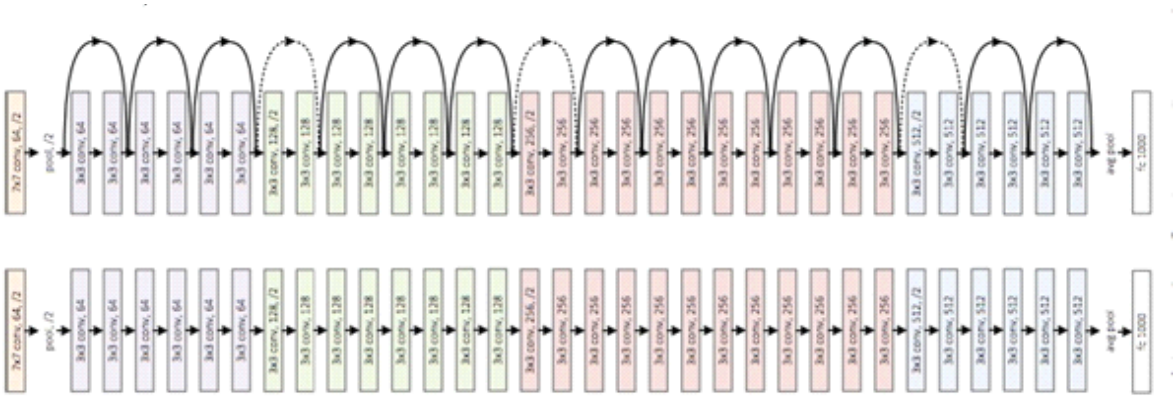


Рис. 1.9 Топология сети plain net (снизу) и ResNet (сверху)

На сегодняшний день все сети, показывающие лучшие результаты являются резидуальными.

3) DenseNet

Архитектура DenseNet (Densely Connected Convolutional Networks) является логическим продолжением ResNet. Авторы оригинальной статьи [8] утверждают, что конволюционные сети могут быть существенно более глубокими, более точными и эффективными для обучения, если они содержат более короткие соединения между слоями. Данная архитектура представлена на рис. 1.10 и 1.11.

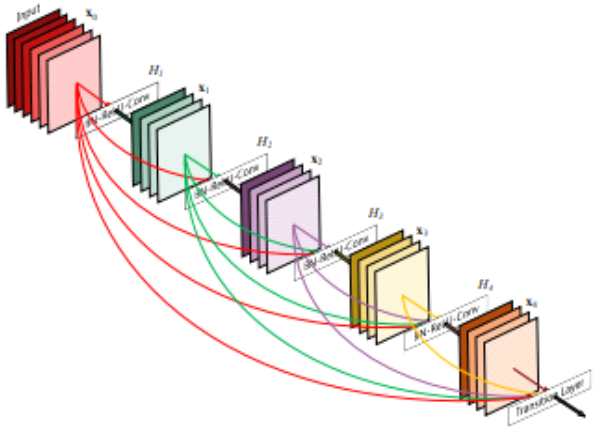


Рис 1.10 Архитектура DenseNet.

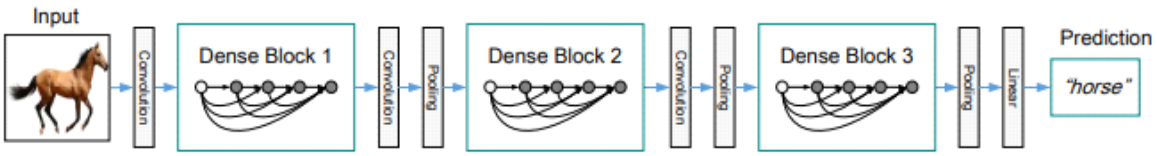


Рис. 1.11 Схемы работы нейронной сети DenseNet

В то время как традиционные конволюционные нейронные сети с L уровнями имеют L соединений - по одному между каждым уровнем и его последующим уровнем – сеть DenseNet имеет L * (L + 1) / 2 прямых соединений. Карты признаков внутри Dense блока передаются на все последующие слои, в отличии от классических конволюционных нейронных сетей, где карта признаков с предыдущего блока передается только на вход последующего блока. Так же отличием является то, что после Dense блока выполняется операция конкатинации, а не суммирования, как в сети ResNet.

1.2.3. Сравнительный анализ фреймворков глубокого обучения

1) Caffe

Caffe представляет собой платформу, включающую в себя предустановленные наборы обучаемых нейронных сетей. Данный фреймворк известен своими возможностями обработки изображений, также в платформу была включена поддержка пакета прикладного ПО MATLAB.

К плюсам данного фреймворка можно отнести следующее:

- Все модели фреймворка имеют открытый исходный код;
- Фреймворк обеспечивает высокую скорость и эффективность работы;
- Данный фреймворк является исторически самым ранним, благодаря этому у него существует большое активное сообщество;
- Сопряжение и поддержка языков C, C++ и Python, поддержка CNN;
- Фреймворк также специализируется на решении различных вычислительных задач;

К минусам можно отнести то, что фреймворк глубокого обучения Caffe не способен обрабатывать комплексные массивы данных, но зато он сравнительно быстр при визуальной обработке изображений.

2) MXNet

MXNet - это фреймворк для глубокого обучения созданный Apache, который поддерживает изобилие языков, например, Python, Julia, C++, R или JavaScript. Он применяется в Microsoft, Intel и веб-сервисах Amazon.

К плюсам данного фреймворка можно отнести следующее:

- Высокая скорость работы и гибкость при работе с алгоритмами глубокого обучения;
- Обеспечение продвинутой поддержки GPU;
- Возможность запуска на любом устройстве;
- Высокопроизводительное императивное API;
- Легкую поддержку моделей;
- Масштабируемость;

К минусам можно отнести то, что фреймворк глубокого обучения MXNet имеет сравнительно меньшее сообщество, и то, что он не пользуется большой популярностью в научных кругах, в отличие от других фреймворков глубокого обучения.

3) TensorFlow

Фреймворк глубокого обучения TensorFlow разработан компанией Google и написан на языках программирования Python и C++. TensorFlow является одной из лучших открытых библиотек для численных вычислений. О качестве данного фреймворка говорит то, что даже такие гиганты как DeepMind, Uber, AirBnB или Dropbox выбрали этот фреймворк для своих нужд.

К плюсам данного фреймворка можно отнести следующее:

- Большое количество руководств и документации;
- Мощные средства мониторинга процесса обучения моделей и визуализации (Tensorboard);
- Большое сообщество;
- Поддержка распределенного обучения;

К минусам можно отнести низкую скорость работы по сравнению с другими фреймворками, такими как, к примеру, MXNet. Еще одним важным минусом является то, что в данном фреймворке лишь один полностью поддерживаемый язык программирования – Python.

4) PyTorch

В отличие от TensorFlow, библиотека PyTorch оперирует динамически обновляемым графом, то есть позволяет вносить изменения в архитектуру в процессе работы. PyTorch используется в основном, чтобы обучать модели быстро и эффективно, поэтому его выбирает большое количество разработчиков.

К плюсам данного фреймворка можно отнести следующее:

- Простой и прозрачный процесс создания модели;
- Популярность в научных кругах;
- Поддержка популярных отладчиков, таких как pdb, ipdb или PyCharm;
- Поддержка параллелизма данных;
- Множество предварительно обученных моделей и готовых модульных частей, которые легко комбинировать;

В данной дипломной работе будет использоваться именно этот фреймворк. Он выбран по причине удобства взаимодействия и скорости работы, а также потому, что на нем реализованы все базовые алгоритмы слежения за объектом на видеопоследовательности, которые будут рассмотрены в рамках данной дипломной работы.

1.2.4. Метрики качества для оценки результатов работы ГКНС

Для оценки качества работы алгоритмов слежения за объектом на видеопоследовательности существует ряд оценочных метрик, схожих с метриками для оценки качества обнаружения объектов. Для задачи обнаружения объектов обычно используется такая метрика качества, как средняя точность (Average Precision (AP)), полученная из двух показателей: точности (precision) и полноты (recall). При обнаружении объекта одного конкретного класса возникают такие события:

• Истинно положительные события (true positives) – объект нужного класса был правильно заключен в охватывающую рамку.

• Ложно положительные события (false positives) – в охватывающую рамку было заключено нечто иное, отличное от объекта нужного класса.

1) Точность и полнота

Точность (precision) обозначает процент верных предсказаний для охватывающей рамки (true positives) среди всех результатов для этого класса. Точность рассчитывается по формуле 1.4.

$$Precision = \frac{True\ positives}{True\ positives + False\ positives} \tag{1.4}$$

Полнота (recall) обозначает процент найденных обрамляющих окон для данного класса, среди всех, представленных в ground truth (достоверных даных) этого изображения. Полнота рассчитывается по формуле 1.5.

$$Recall = \frac{True\ positives}{Number\ of\ ground\ truth\ boxes} \tag{1.5}$$

2) Пересечение по объединению

Для того чтобы была возможность оценить правильность заключения объекта в обрамляющее окно используется метрика IoU (рис. 1.12). Типичное

пороговое значение, указывающее на правильное обнаружение, составляет IoU > 50%.

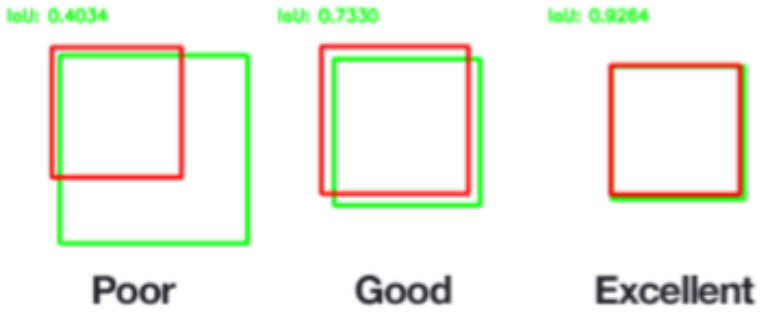


Рис. 1.12 Метрика IoU

Пересечение по объединению (Intersection over Union) – метрика, используемая для оценки точности работы алгоритмов, детектирующих объекты. Значение IoU равно частному площади пересечения предсказанного обрамляющего окна и окна-ответа из ground truth на площадь объединения этих окон. Сравнение описанных метрик приведено на рис. 1.13.

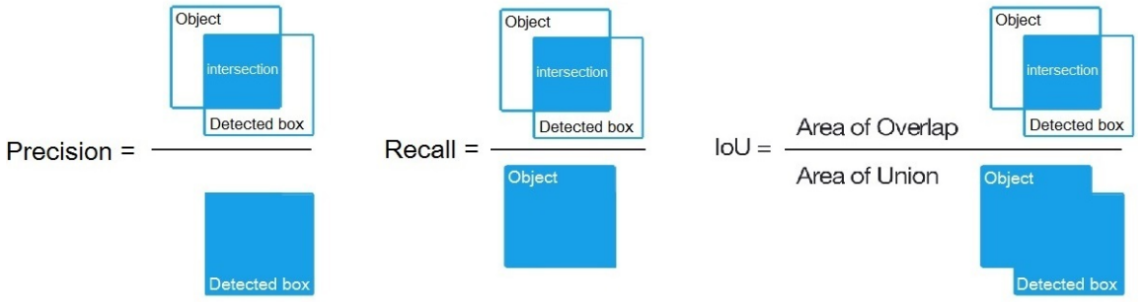


Рис. 1.13 Сравнение метрик Precision, Recall, IoU

3) Average Precision

Average Precision (AP) – метрика качества, которая показывает среднюю точность. AP вычисляется как площадь под кривой precision – recall. Что бы вычислить значение AP необходимо взять определенный интеграл от функции кривой precision - recall в интервале от 0 до 1 (формула 1.6):

$$\mathrm{AP} = \int_0^1 p(r)dr \tag{1.6}$$

1.2.5. Наборы данных для обучения нейронной сети

Для обучения нейронной сети необходимы наборы данных. Такие данные принято называть датасетами. Поскольку данная квалификационная работа посвящена компьютерному зрению, то будет вести речь о датасетах для обучения нейронных сетей в области изображений или видеопоследовательностей, которые представляются набором изображений (1 секунда видеопоследовательности разбивается на несколько изображений). Весь набор изображений делится на три выборки: тренировочную, валидационную и тестовую. На тренировочной выборке происходит обучение нейронной сети. Во время обучения происходит валидация - предварительное тестирование для устранения переобучения нейронной сети. После завершения обучения происходит тестирование обученной нейронной сети на тестовой выборке, чтобы получить финальный результат качества.

Датасеты можно создавать вручную, а можно найти уже готовые. Создавать вручную конечно же лучше для конкретной задачи, но датасеты состоят из огромного количества данных, прошедших обработку. Чтобы создать датасет необходимо много времени и ресурсов.

Существует множество различных готовых датасетов, некоторые находятся в свободном доступе, некоторые нет. Примеры датасетов, находящихся в свободном доступе: YouTube-VOS, DAVIS, VOT.

1.2.6. Аугментация данных

Для обучения нейронных сетей требуется большое количество данных, которые довольно трудоемко собрать и размечать. Для того, чтобы увеличить уже имеющиеся наборы обучающих данных применяются различные аугментации – наборы преобразований, которые некоторым образом

изменяют изображения. Таким образом можно существенно увеличить размер обучающей выборки, что положительно сказывается на качестве обучения. Помимо этого, аугментации так же позволяют добавить робастности нейронной сети, а также избежать переобучения, за счет добавления неких искажений к оригинальным изображениям. В задаче отслеживания, а также в задаче детекции применяются следующие аугментации: отражение, поворот, смещение, изменение размера, изменение яркости, аффинное преобразование, изменение контраста, зашумление. Примеры аугментаций приведены на рис. 1.14:

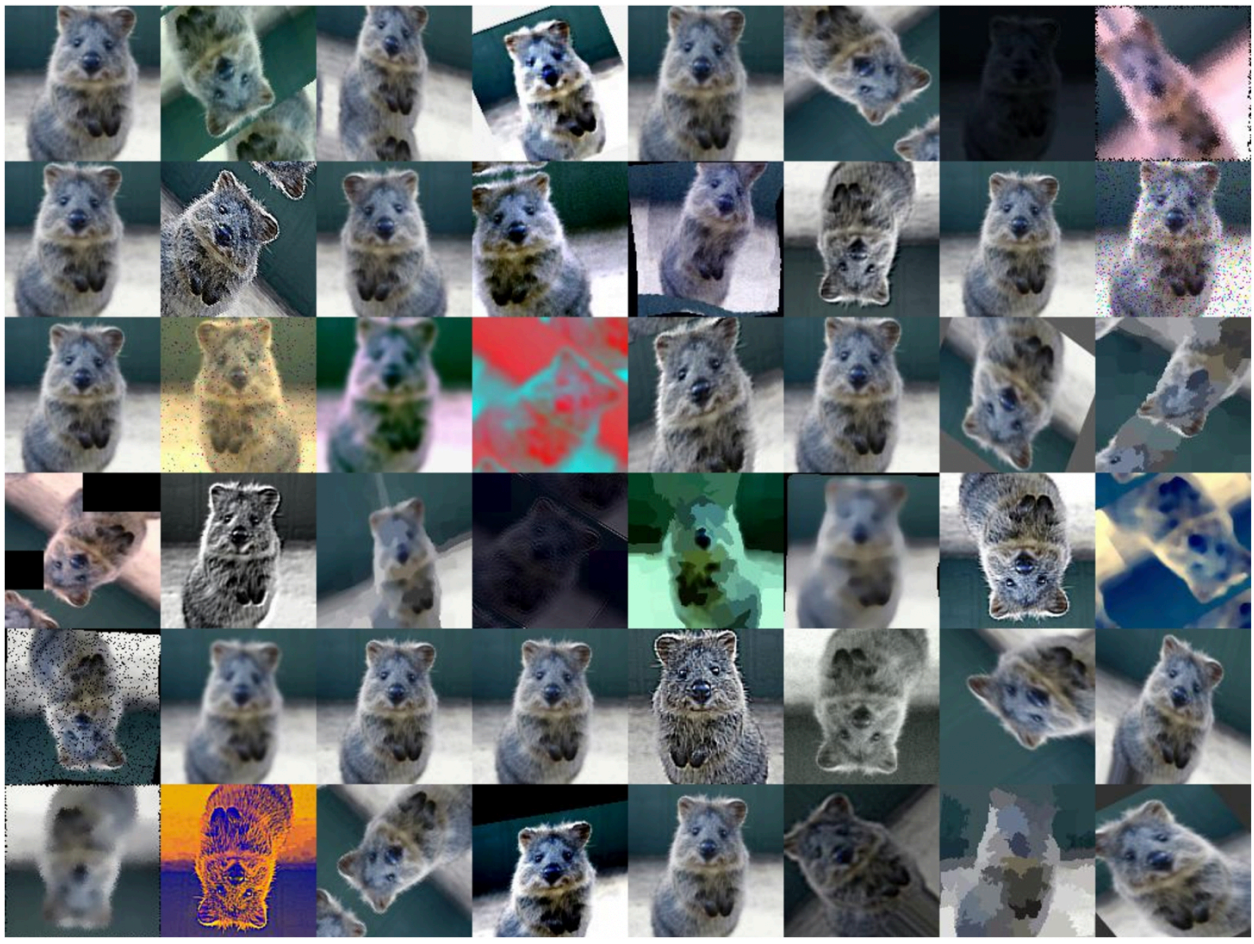


Рис. 1.14 Примеры аугментации изображений

Аугментации изображений можно выполнять с помощью библиотек для работы с изображениями, например:

- Pillow
- OpenCV

- Scikit-image
- Torchvision

2. ПРАКТИЧЕСКАЯ ЧАСТЬ

2.1. Анализ алгоритмов SiamMask и SiamRPN

Алгоритм SiamMask (рис. 2.1) был представлен на CVPR2019 и стал лучшей нейросетевой моделью для отслеживания целевого объекта на видеопоследовательности. Он сочетал в себе качество и скорость работы. Существовало две вариации алгоритма, с сегментацией и без. В рамках данной выпускной квалификационной работы будет рассматриваться вариация без масок сегментации под названием SiamRPN (рис. 2.2). Алгоритм работает следующим образом:

Нейронная сеть получает на вход 2 изображения:

1) Изображение, на котором необходимо искать объект.

2) Изображение, на котором присутствует только искомый объект, вырезанное из исходного изображения.

С помощью нейронной сети с архитектурой ResNet-50 из входных изображений выделяются карты признаков, по которым далее проходит глубокая кросс-корреляция.

На выходе сети получаются 2 вектора:

1) Вектор ограничивающего прямоугольника.

2) Вектор класса (в данной задаче определяющий вероятность того, что найденный объект соответствует искомому).

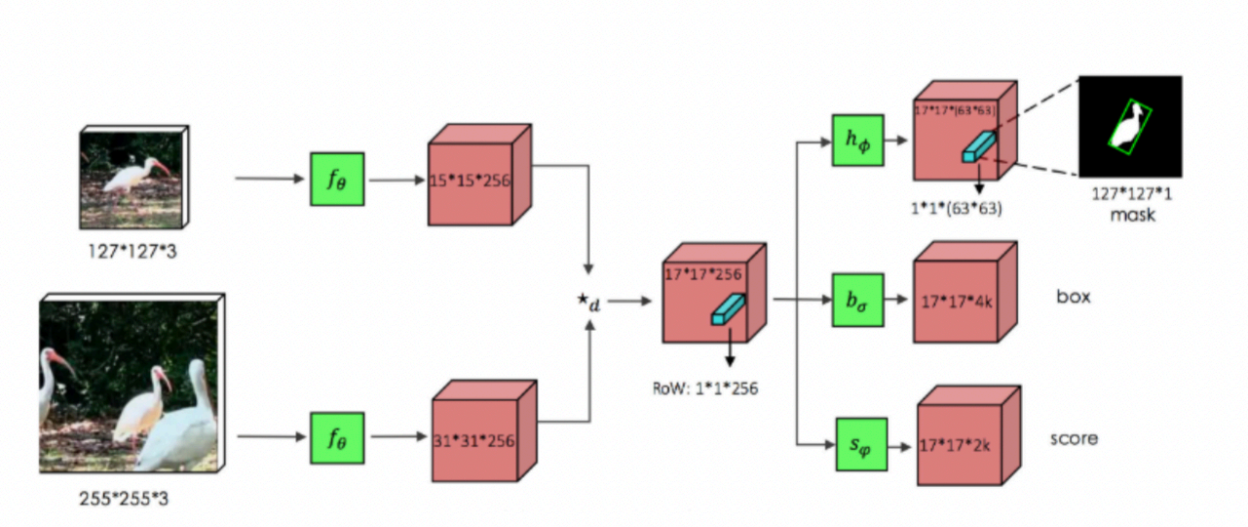


Рис. 2.1 Схема работы алгоритма SiamMask

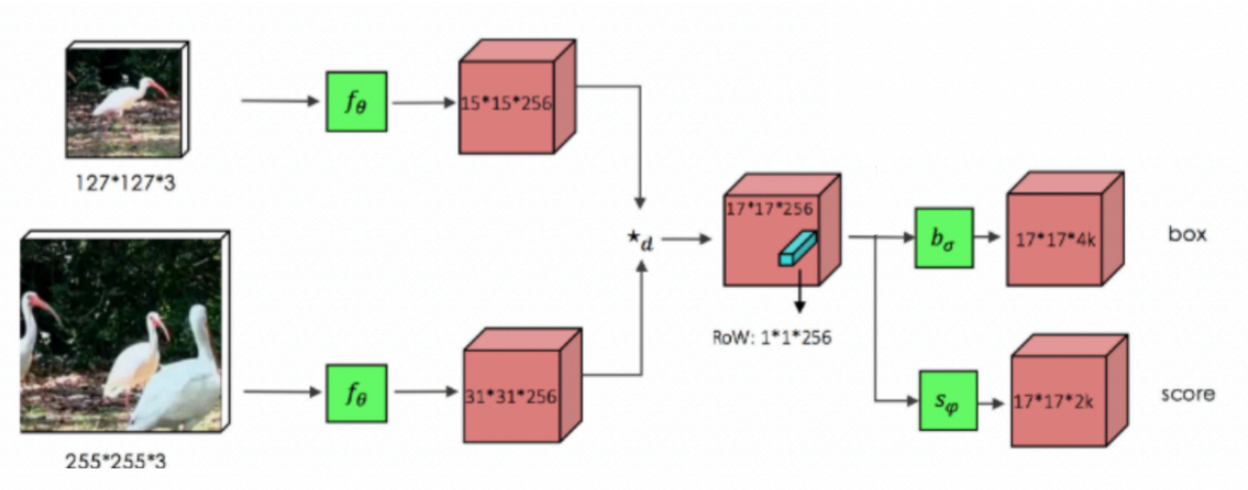


Рис 2.2 Схема работы алгоритма SiamRPN

При обучении SiamRPN в качестве функции потерь для классификации используется кросс энтропийная функция потерь, а в качестве функции потерь для обнаружения используется сглаженная L1 функция потерь.

Кросс энтропийная функция потерь рассчитывается по формуле 2.1:

$$\text{loss}(x, class) = -\log\left(\frac{\exp(x[class])}{\sum_j \exp(x[j])}\right) = -x[class] + \log\left(\sum_j \exp(x[j])\right) \quad (2.1)$$

где:

x – вектор вероятностей для каждого класса;

target – истинный класс.

Сглаженная L1 функция потерь рассчитывается по формуле 2.2:

$$\text{loss}(x, y) = \frac{1}{n} \sum_i z_i \tag{2.2}$$

где:

x – предсказанный вектор;

y – целевой вектор;

z – определяется по формуле 2.3:

$$z_i = \begin{cases} 0.5(x_i - y_i)^2 / beta, & \text{if } |x_i - y_i| < beta \\ |x_i - y_i| - 0.5 * beta, & \text{otherwise} \end{cases} \tag{2.3}$$

где:

beta – настраиваемый параметр.

При задании параметра beta равным нулю, данная функция потерь становится аналогична обычной L1 функции потерь.

Общая функция потерь алгоритма, представлена формулой 2.4:

$$L = \lambda_1 * L_{score} + \lambda_2 * L_{box} \tag{2.4}$$

Где:
L - общая функция потерь алгоритма;
λ_1 - вес функции потерь классификации;
λ_2 - вес функции потерь местоположения;
L_{score} - функция потерь классификации;
L_{box} - функция потерь местоположения.

2.2. Анализ алгоритма DETR

Целью задачи обнаружения объектов является предсказание набора ограничивающих рамок и меток класса для каждого интересующего объекта. Современные детекторы решают поставленную задачу прогнозирования с помощью алгоритма предобработки и постобработки. Предобработка включает себя алгоритм генерации анкеров (набора прямоугольников с изменяемыми размерами и соотношениями сторон) и кодирования эталонных ограничивающих рамок с учетом сгенерированных анкеров (построения карты смешений). Постобработка включает в себя декодирование карты смещений и алгоритм NMS (Non-maximum Suppression – алгоритм подавления немаксимумов), который служит для устранения большого количества ограничивающих прямоугольников с низкой точностью обнаружения. Данный алгоритмы предобработки и постобработки негативно сказываются на производительности. Пример работы алгоритма NMS приведен на рис. 2.3.

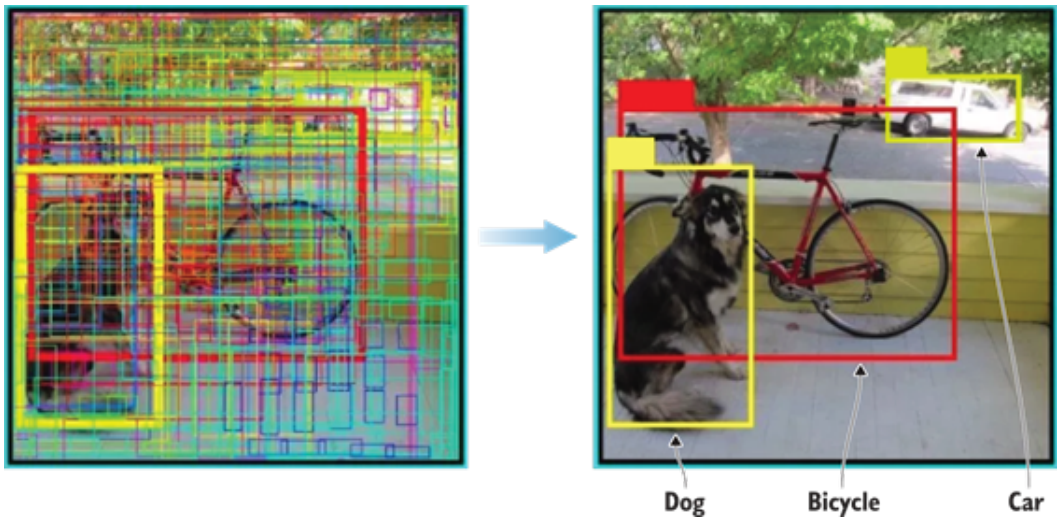


Рис. 2.3 Пример работы алгоритма NMS

Для упрощения данной части алгоритмов обнаружения объектов в 2020 году был предложен принципиально новый подход с использованием архитектуры трансформер на выходе нейронной сети, описанный в статье

«End-to-End Object Detection with Transformers». Схема работы алгоритма DETR приведена на рис. 2.4:

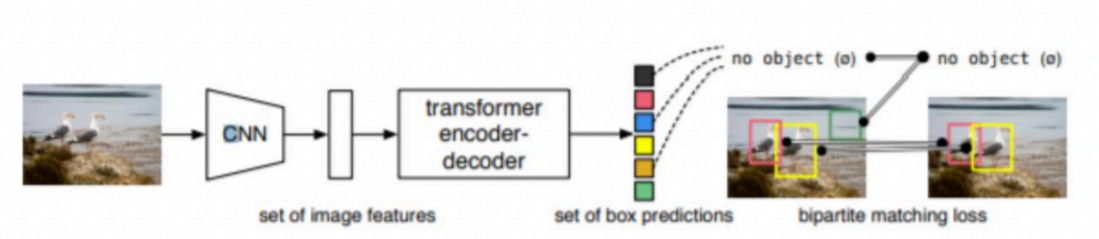


Рис. 2.4 Схема работы алгоритма DETR

Алгоритм DETR работает следующим образом:

1. На вход ГКНС подается входное изображение, из которого извлекаются карты признаков;
2. Выходные карты признаков объединяются с признаками, полученными посредством пространственного кодирования;
3. Полученный данные передаются на вход трансформера, в результате чего получаются 2 выхода: Набор меток классов и набор ограничивающих рамок. Стандартно обнаруживаются 100 рамок. Если в найденной рамке нет объекта, то класс такой рамки будет нулевым, и она не будет учитываться в финальном предсказании нейронной сети.

Алгоритм DETR предсказывает все объекты одновременно и обучается с заданной функцией потерь, которая выполняет сопоставление между предсказанными и истинными объектами. DETR, по сравнению с классическими алгоритмами обнаружения объектов, такими как RCNN [9], Fast-RCNN [10], Faster-RCNN [11], упрощает алгоритм обнаружения, убирая вручную разработанные компоненты. Алгоритм не требует каких-либо настраиваемых слоев и, таким образом, может быть легко воспроизведен в любой структуре, содержащей стандартные классы ГКНС и трансформеров.

Функция потерь алгоритма DETR устроена следующим образом: DETR выводит набор фиксированного размера из N прогнозов за один проход через нейронную сеть, где N установлено так, чтобы быть значительно большим, чем типовое количество объектов в изображении. Одной из основных трудностей обучения является оценка предсказанных объектов (класс, положение) по отношению к истине. Функция потерь в данном алгоритме вычисляет соответствие между предсказанными и истинными классами объектов, а затем вычисляет соответствие между ограничивающими рамками, в которых есть объекты.

Функция потерь алгоритма вычисляется по формуле 2.5:

$$\hat{\sigma} = \underset{\sigma \in \mathfrak{S}_N}{\arg\min} \sum_{i}^{N} \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}) \tag{2.5}$$

где:

σ - основное набор элементов;

$\mathfrak{S}$ – дополненный набор элементов классов отсутствия объекта;

y – основное истинное множество объектов;

$\hat{y}$ – набор из N предсказаний.

$\mathcal{L}_{match}(y_i, \hat{y}_{\sigma(i)})$ - коэффициент штраф, вычисляемый с помощью Hungarian algorithm [12].

Алгоритм DETR является не самым быстрым алгоритмом, за счет большого количества предсказаний (в оригинальной статье на выходе нейронной сети получается 100 рамок), а так же, его обучение занимает большое количество времени (порядка недели на восьми графических ускорителях Tesla V100). В рамках данной работы потребуется лишь одно предсказание, так как отслеживается только выделенный объект. Также, из алгоритма DETR будет взята только часть нейронной, включающая в себя

трансформер. Благодаря этому, устраняется описанные выше недостатки алгоритма.

2.3. Построение модели нейронной сети для задачи слежения за целевым объектом на видеопоследовательности

В рамках данной выпускной квалификационной работы был разработан алгоритм слежения за целевым объектом на видеопоследовательности с использованием алгоритмов SiamMask и DETR, описанных выше. От алгоритма SiamMask была взята идея сиамской нейронной сети, с небольшим изменением пространственных изменений входных данных, а также функция глубокой кроскорреляции. На вход сиамской нейронной сети поступают два изображения, вырезанные особым образом (афинное преобразование с учетом примерного местоположения отслеживаемого объекта): первое изображение обозначим как изображение образец (размер изображения 127 x 127 пикселей), а второе обозначим как изображение поиска (размер изображения 255 x 255 пикселей). Для данных изображений выполняются различные аугментации, такие как: отражение, поворот, смещение, размытие, изменение контраста. После прохождения через нейронную сеть для извлечения карт признаков (в качестве такой нейронной сети выступает ResNet-50, который описан ниже), полученные карты признаков попадают на слой глубокой кросскореляции, что помогает получить информацию о положении объекта с изображения образца на изображении поиска. Глубокая кросскореляция реализуется двумерной конволюцией, в качестве весов которой выступают карты признаков, полученный с изображения образца. На этом использование части алгоритма SiamMask в данной работе завершено.

В разработанном алгоритме слежения за объектом на видеопоследовательности в качестве нейронной сети выступает измененный ResNet-50. В качестве изменений можно выделить две особенности: слои батч

нормализации замораживаются (батч нормализация использует предобученные статистики на самом большом датасете ImageNet[13], который находится в свободном доступе), и алгоритм пространственного кодирования входных данных [14], который необходим для работы архитектуры трансформер, описанной далее.

Архитектура трансформер была предложена в статье «Attention is all you need» [15] в 2017 году и позиционировалась как замена рекурентным сетям благодаря механизму внимания. Данная архитектура получила широкое распространение благодаря своей точности и скорости работы (скорость на несколько порядков выше, чем у рекурентных нейронных сетей). Она применяется во многих областях глубокого обучения, например в компьютерной обработке тексов и изображений. Архитектура трансформер приведена на рис. 2.5:

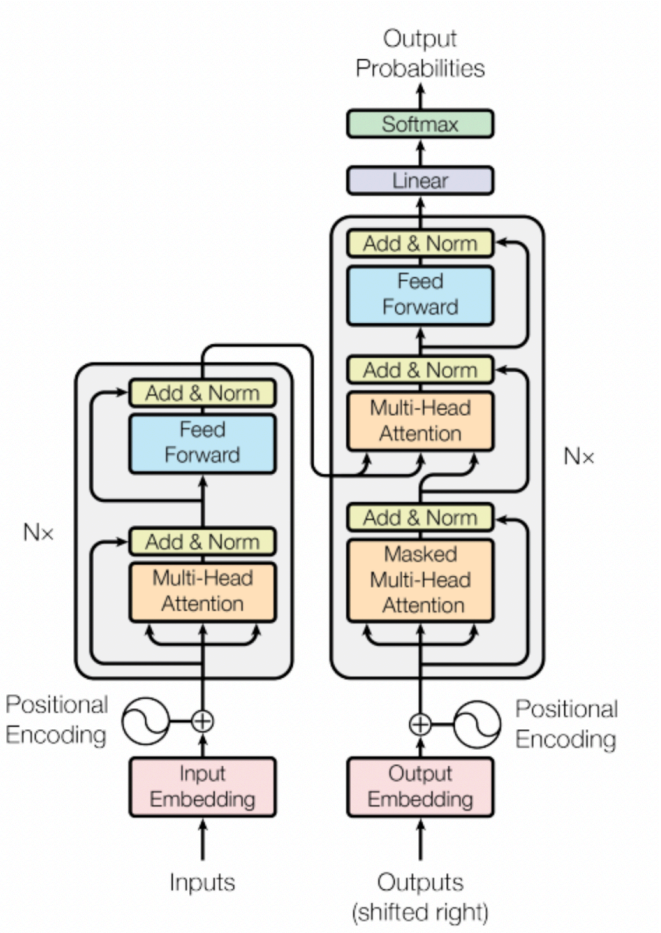


Рис. 2.5 Архитектура трансформер

Архитектуру трансформер можно разделить на 3 составные части: энкодер, декодер и механизм внимания (Multi-head attention).

Энкодер (изображен в левой части рис. 20) принимает на вход изображения и их вектора позиционного кодирования. Они проходят через стандартные полносвязные слои и резидуальные связи (как в архитектуре ResNet), а также через слои внимания.

Механизм внимания – это новый слой, который в задаче обработки текста позволяет каждому слову в предложении взаимодействовать с другими словами в предложении. В задаче обнаружения объектов задача механизма внимания состоит в том, чтобы найти область интереса (найти области искомого объекта на входном изображении). Схема механизма внимания и визуализации его работы в области обнаружения объектов приведена на рис. 2.6:

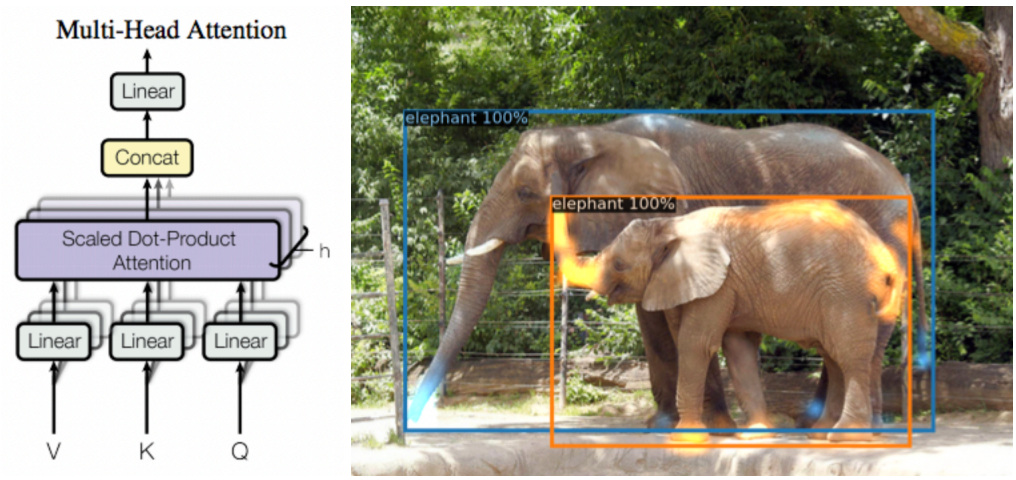


Рис. 2.6 Механизм внимания и его визуализация

На вход механизма внимания подаются вектора Query (в данной работе Query представляет собой один эмбединг ограничивающей рамки), и несколько пар Key и Value (Key и Value это чаще всего один и тот же вектор). Каждый из них проходит через полносвязный слой, а потом вычисляется скалярное произведение Q со всеми K по очереди, прогоняется результат этих скалярных произведений через softmax, с полученными весами все вектора V суммируются в единый вектор. Таких механизмов внимания параллельно тренируется несколько (на рис. 2.6 их количество обозначено через h). Далее

результат конкатенируется (объединяется по какой-либо оси), еще раз прогоняется через полносвязный слой и передается на выход. Вход и выход такого блока имеют одинаковую размерность, таким образом можно легко добавлять глубину сети (благодаря использованию резидуальных связей не будет возникать проблема затухания градиента при обучении). Обычно используется 6 блоков механизма внимания.

В декодере есть два разных типа использования Multi-head attention: первый – аналогичен использованию в энкодере, а второй – дает возможность обратиться к выходу энкодера. Во втором типе использования Query — это вектор входа в декодере, а пары Key/Value — финальные эмбеддинги энкодера. Данный механизма также стандартно повторяется 6 раз, как и в части энкодера.

В классической архитектуре трансформер в конце стоит слой Softmax. В данной работе, как и в алгоритме DETR стоит трехслойный перцептрон (для ограничивающей рамки) и полносвязный слой (для классификации). Данное преобразование на рис. 2.7 обозначено как FFN (feed-forward network).

Для разработанного алгоритма отслеживания целевого объекта на видеопоследовательности выходом сети является одна ограничивающая рамка (координаты центра целевого объекта) и класс найденного объекта (найден целевой объект или найдено что-то другое). Схема разработанного алгоритма представлена на рис. 2.7:

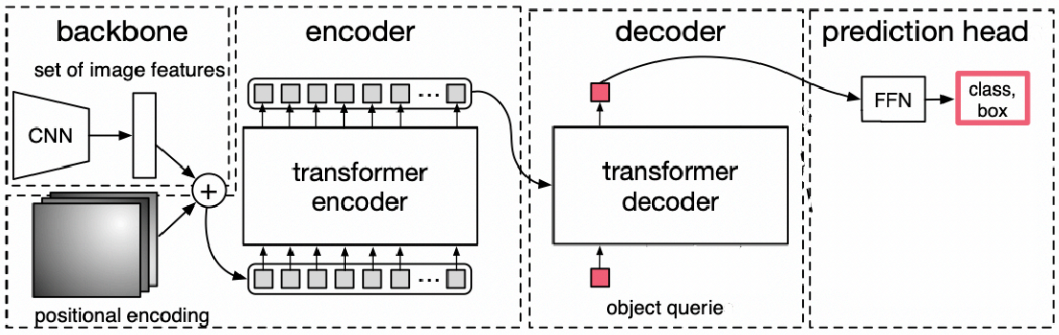
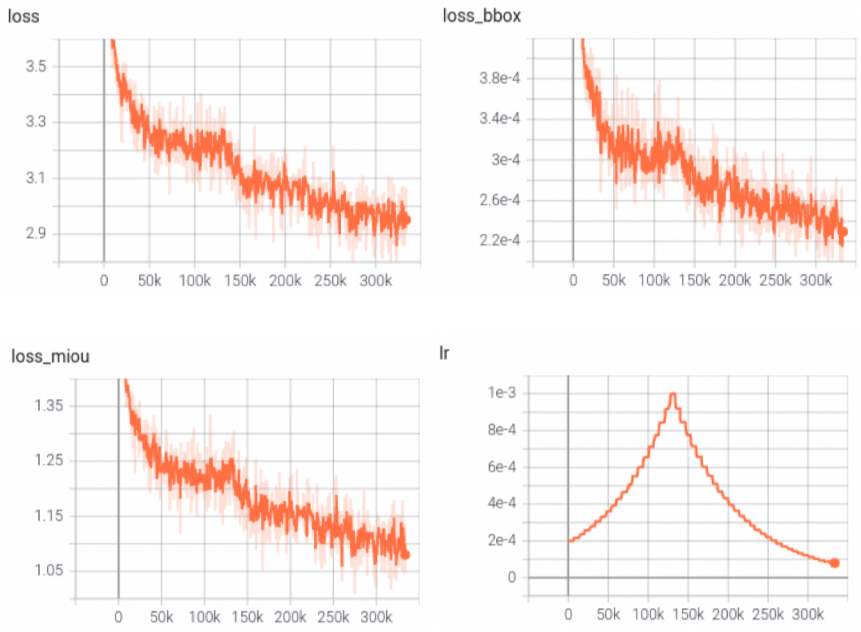


Рис. 2.7 Схема разработанного алгоритма.

Обучение и тестирование разработанного алгоритма производилось на датасете YouTube-VOS. Качество работы алгоритма оказалось выше, чем у алгоритма SiamRPN (0.55 против 0.53), но скорость работы оказалась чуть ниже, что не критично. Разработанный алгоритм инвариантен к размерам входных и выходных изображений, а также к размерам целевых объектов, что отлично позволяет отслеживать даже объекты малых размеров.

2.4. Обучения и примеры работы разработанного алгоритма слежения за целевым объектом на видеопоследовательности

Нейронная сеть, разработанная в рамках данной выпускной квалификационной работы, обучалась 50 эпох в течении суток на 10 видеокартах Nvidia RTX 2080Ti. В качестве оптимизатора был выбран алгоритм оптимизации Radam + Lookahead. В качестве функции потерь использовалась сумма кроссэнтропийной функции потерь (для классификации), сглаженный L1 функция потерь и функция потерь на основе IoU (бралось значение 1 – IoU). Графики обучения представлены на рис. 2.8:



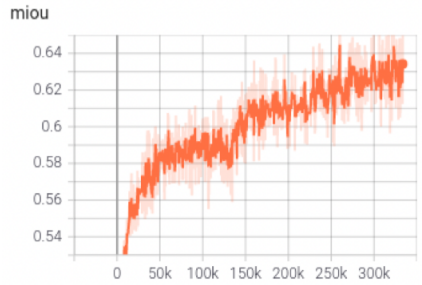


Рис. 2.8 Графики обучения разработанного алгоритма

Для визуализации работы разработанного алгоритма отслеживания целевого объекта на видеопоследовательности было выбрано произвольное видео из тестового набора. На нулевом кадре вручную был выделен целевой объект. На остальных кадрах видеопоследовательности алгоритм обнаруживал положение ограничивающей рамки. Визуализация работы разработанного алгоритма приведена на рис. 2.9:

Рис. 2.9. Визуализация работы разработанного алгоритма

ЗАКЛЮЧЕНИЕ

Данная выпускная квалификационная работа посвящена разработке алгоритма слежения за объектами малых размеров по видеопоследовательностям с использованием глубоких ковнволюционных нейронных сетей. В ходе работы получены следующие основные результаты:

Разработан алгоритм слежения за целевым объектом на видеопоследовательности, в том числе:

- Проведен анализ существующих методов слежения за объектами на видеопоследовательностях, описанных в открытых источниках.;
- Разработан алгоритм слежения за целевым объектом на видеопоследовательности, сочетающий в себе сильные стороны алгоритмов SiamMask и DETR;
- Проведено обучение и тестирование нейронной сети для слежения за целевым объектом на видеопоследовательности.

Результаты тестирования показали, что использование подхода на базе разработанного алгоритма позволяет достигать качества, сравнимого с результатами взятым за основу алгоритмов и превосходить их по точности. Так же результаты тестирования показали эффективность работы архитектуры трансформер в задаче обнаружения целевого объекта на видеопоследовательности.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. http://image-net.org/ – Large Scale Visual Recognition Challenge.
2. MOTS: Multi-Object Tracking and Segmentation / Paul Voigtlaender, Michael Krause, Aljosa Osep, Jonathon Luiten, Berin Balachandar Gnana Sekar, Andreas Geiger, Bastian Leibe // arXiv:1902.03604v2 - 2019 – URL https://arxiv.org/pdf/1902.03604.pdf
3. Fast Online Object Tracking and Segmentation: A Unifying Approach / Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, Philip H.S. Torr // arXiv:1812.05050v2 - 2019 – URL https://arxiv.org/pdf/1812.05050v2.pdf
4. End-to-End Object Detection with Transformers / Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, Sergey Zagoruyko // https://arxiv.org/abs/2005.12872 - 2020 - URL https://arxiv.org/pdf/2005.12872.pdf
5. https://arxiv.org/abs/1512.07108 - Recent Advances in Convolutional Neural Networks. Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Li Wang, Gang Wang, Jianfei Cai, Tsuhan Chen.
6. https://arxiv.org/abs/1409.1556 - Very Deep Convolutional Networks for Large-Scale Image Recognition Karen Simonyan, Andrew Zisserman.
7. K. He, X. Zhang , S. Ren , J. Sun, «Deep Residual Learning for Image Recognition», CVPR, 2015.
8. https://arxiv.org/abs/1608.06993 Densely Connected Convolutional Networks, Gao Huang, Zhuang Liu Laurens van der Maaten.
9. R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich fea- ture hierarchies for accurate object detection and semantic segmentation. In CVPR, 2014
10. Ross Girshick. Fast R-CNN.
11. Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.

12. Stewart, R.J., Andriluka, M., Ng, A.Y.: End-to-end people detection in crowded scenes. In: CVPR (2015).
13. http://image-net.org/ – Large Scale Visual Recognition Challenge.
14. Parmar, N., Vaswani, A., Uszkoreit, J., Kaiser, L., Shazeer, N., Ku, A., Tran, D.: Image transformer. In: ICML (2018).
15. Vaswani,A.,Shazeer,N.,Parmar,N.,Uszkoreit,J.,Jones,L.,Gomez,A.N.,Kaiser, L., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017).
16. Сахаров С.В., Морозов А.Ю. Быстрый алгоритм слежения за целевым объектом на видеопоследовательности. Материалы XIII международной конференции по прикладной математике и механике в аэрокосмической отрасли (AMMAI'2020), Москва, 2020.
17. Сахаров С.В. Использование механизма внимания в задаче отслеживания объекта на видеопоследовательности. Постер для заочного участия во II Международной конференции «Математическое моделирование в материаловедении электронных компонентов» (ММЭК2020), Москва, 2020.