Санкт–Петербургский государственный университет

Кузнецов Дмитрий Алексеевич

Выпускная квалификационная работа

Детектирование объектов на изображениях с использованием машинного обучения

Уровень образования: магистратура
Направление 03.04.01 «Прикладные математика и физика»
Основная образовательная программа ВМ.5521.2020 «Математические и информационные технологии»

Научный руководитель:
доцент, кафедра ТСУЭФА, к.ф. - м.н.
Козынченко Владимир Александрович

Рецензент:
Михайлов Евгений Александрович

Санкт-Петербург
2022 г.

Содержание

Введение

Курение — социальная проблема: важной задачей общества сегодня является не допустить того, чтобы число курильщиков росло, особенно среди детей и подростков. Согласно исследованиям ученых из Калифорнийского университета в Сан-Франциско (UCSF) [23], несмотря на существующие ограничения, количество сцен с курящими людьми в кино постепенно увеличивается, что действует как пропаганда курения и провоцирует рост числа курящих среди молодой аудитории. По соображениям учёных, чем чаще подростки видят курящих персонажей на экране, тем выше вероятность того, что они сами начнут курить.

Одной из мер по борьбе с ростом числа курильщиков, в том числе среди несовершеннолетних, является цензурирование визуального контента в интернете и на ТВ: фильмы со сценами курения как в России, так и в мире имеют более высокие возрастные ограничения, содержат специальные текстовые предупреждения о демонстрации курения и о его вреде. Некоторые прокатчики пошли дальше: они используют инструменты для закрашивания той небольшой части кадра, которая содержит сигарету или другое устройство для потребления никотина. Поиск и закрашивание таких областей обычно выполняется вручную покадрово, что сильно усложняет задачу ввиду потенциально большого объема работы.

Другим местом, где несовершеннолетние могут столкнуться с демонстрацией курения, являются социальные сети. В таких соцсетях, как Facebook, уже существует механизм автоматического распознавания потенциально неприемлемого контента — сцен насилия, жестокого обращения с животными и так далее. Изображения, содержащие сигареты или другие курительные предметы, также могут быть отнесены к категории потенциально неприемлемого контента.

Данная работа посвящена автоматизации поиска сигарет на изображениях (или кадрах). Детектирование объектов — задача, в рамках которой необходимо найти и выделить все объекты на изображении из множества заранее известных классов посредством нахождения координат их ограничивающих рамок и классификации контента внутри них. Детектирование —

второй по популярности тип задач компьютерного зрения после классификации. Его отличает относительная простота, так как обучение проходит с учителем, а самые современные модели позволяют достигнуть высокой точности даже в системах реального времени.

Научных работ, связанных с детектированием на изображениях именно сигарет, в ходе работы обнаружено не было, однако существуют работы, посвященные более широкой области — **курению**. Некоторые подходы пытаются найти дым и определить цвет его источника. В рамках другого подхода для детектирования курения используется дополнительная нейронная сеть для поиска на изображении областей интереса — лиц и кистей рук, — после чего стоит задача обнаружить сигарету или нечто подобное внутри каждой области. Такой подход ведет к уменьшению ошибок первого рода (то есть помогает не пропускать сигареты на изображениях), но при работе с сигаретами могут возникнуть две основные сложности:

1. наличие предметов, похожих на сигарету — ручек, трубочек, белых полосок и пр., которые может быть трудно отличить даже человеку;

2. хотя сигареты, как правило, одной формы, может использоваться не только белая, но и цветная бумага.

В рамках работы для решения задачи детектирования сигарет используются общие подходы решения такого класса задач — обучение модели (YOLOv5), которая будет предсказывать ограничивающие рамки. А для того, чтобы сигареты и посторонние предметы внутри предсказанных рамок были правильно классифицированы, предлагается использование дополнительной модели для классификации контента внутри предсказанных ограничивающих рамок. Классификатор не помогает найти другие объекты на изображении, однако способствует отсеиванию заведомо неверно классифицированных объектов — ручек, трубочек, белых полосок и пр. Таким образом, модель уменьшает количество ошибок второго рода, что может быть актуально при затратных операциях по закрашиванию обнаруженных областей. Предложенный подход имеет ряд преимуществ:

- дополнительные данные проще собрать;

- сигареты на дополнительных данных более разнообразны;
- изображения не требуют выделения объектов на них, для них необходима лишь метка класса;
- при обучении бинарного классификатора используются также данные класса «не сигарета», в который входят ручки, белые полоски и прочие объекты, похожие на сигареты.

В ходе работы для задачи классификации размечено около 700 изображений класса сигарета и такой же объем класса не сигарета — каждому изображению присвоена метка 0 либо 1, а для задачи детектирования размечено около 1500 изображений — на каждом из них сигарета выделена минимальным прямоугольником, ее содержащим. Это сравнительно небольшой объем — согласно [20], часто необходимо в 3–4 раза больше данных. Нейросетевые модели редко обучаются с нуля. Обычно для решения конкретной задачи берется заранее предобученная нейронная сеть — полностью либо частично, и затем дообучается на нужных новых данных. Такой подход называется Transfer Learning (передача обучения) и он уже доказал свою эффективность. Модель детектирования, используемая в этой работе заранее предобучена на датасете COCO [16], а модель классификации предобучена на ImageNet. COCO (Common Objects in Context) — огромный (более 1.5 млн объектов из 81 основного класса и 90 второстепенного (небо, трава)) набор данных для детектирования и сегментации, а также одноименная команда ученых, решающая задача в областях детектирования и сегментации (выделение всех пикселей, принадлежащих к объекту).

Цель работы — решить задачу детектирования сигареты при курении, а также повысить точность решения согласно выбранной метрике, используя особенности сигарет и их использования.

В первой главе проведен обзор наборов данных для детектирования и классификации. В ней представлены основные числовые характеристики наборов и примеры, отражающие их качественные составляющие. Во второй главе изложена теория, необходимая для понимания принципов работы используемых моделей и способов их оценивания. Третья глава посвящена

экспериментам — в ней указаны параметры экспериментов и собраны их результаты.

Цель и задачи работы

Цель работы — решить задачу детектирования сигарет на изображениях и исследовать возможность повышения точности классификации. Для достижения этой цели необходимо решить следующие задачи:

1. Провести поиск и разметку изображений, содержащих сигареты, и составить из них набор для обучения, валидации и проверки решения;
2. Выбрать и обучить модель детектирования сигарет на собранных данных;
3. Провести исследование возможности повышения точности детектирования за счет дополнения исходного набора данных с помощью аугментаций и выбора функции потерь;
4. Встроить в модель детектирования блок классификации для уменьшения ошибок второго рода.

Обзор литературы

Часть данных, используемых для задачи детектирования, взята из набора [6], который содержит 2400 изображений, разбитых на два класса: содержащих сцены курения и не содержащих, однако лишь около 500 из них содержат сигареты. Модель YOLOv5 не имеет связанных с ней публикаций, а реализация взята из [20]. Документация этой реализации неполная, однако код позволяет изучить функцию ошибки модели, описанную в разделе 2.2.2. Одна из трех составных частей этой функции — потери локализации — более подробно описана в [1]. Она может быть модернизирована на основе материалов [3, 4, 5], соответствующие эксперименты проведены. За более подробной информацией о семействе моделей YOLO можно обратиться к [10], в которой описана архитектура YOLOv4. Основные аспекты и улучшения YOLO в ходе ее эволюции на протяжении последних семи лет с момента появления первой версии описаны в [8, 9, 25] — пакетная нормализация для ускорения обучения и предотвращения переобучения, пространственный пирамидный блок для предсказаний в различных масштабах, техника якорных прямоугольников, позволяющая упростить задачу регрессии значений ограничивающих прямоугольников соответственно.

Информация о модели классификации MobileNetV2 содержится [22]. Для обучения этой модели используется функция ошибки — перекрестная энтропия [21] и оптимизатор Adam [24].

Вообще, используемые модели детектирования и классификации заранее предобучены на наборах COCO [16] и ImageNet[12] соответственно. О том, почему это работает, можно узнать из [13, 14].

Еще одна техника, повышающая эффективность обучения, — аугментация данных — используется в обеих моделях. Более подробно о технике можно узнать в [11]. Аугментации данных для задачи детектирования уделено больше внимания — в работе дополнительно рассмотрены техники смешивания двух изображений [19] и случайная вставка ограничивающих прямоугольников в произвольное место на изображении [18]. Для классификации же используются стандартные настройки аугментации, реализованные во фреймворке глубокого обучения Pytorch [15].

Глава 1. Набора данных

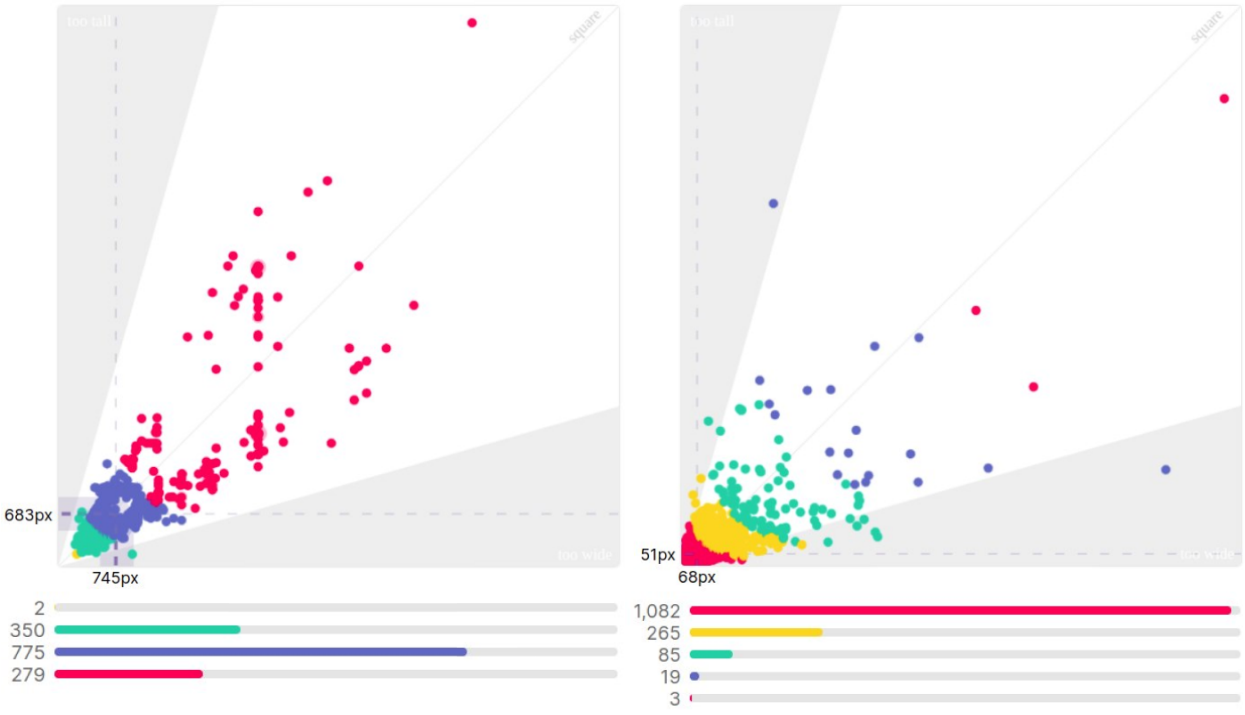
1.1 Набор данных для задачи детектирования

Общие сведения.

В рамках работы было размечено около 1500 изображений, взятых из интернета, а также из [6], содержащего 2400 изображений, из которых лишь приблизительно 500 содержат сигареты. На каждом изображении в основном располагается одна сигарета, реже — две. Набор данных разбит на три множества — тренировочное, валидационное, тестовое — в пропорции $0.7 : 0.1 : 0.2$ соответственно. Тестовое множество дополнительно содержит примерно 200 изображений, не содержащих сигарет.

Анализ набора.

Исходные изображения. Размеры исходных изображений (или точнее площади) имеют распределение, близкое к нормальному. На Рис. 1a точками отмечены соответствующие размеры исходных изображений (в пикселях):



(a) Исходные изображения.

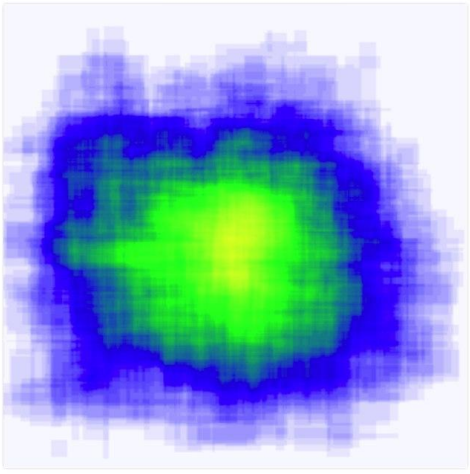
(b) Ограничивающие рамки.

Рис. 1: Визуальное представление размеров набора данных для задачи детектирования. На графиках отмечены медианные значения высоты (по вертикали) и ширины (по горизонтали)

желтых — 2, зеленых — 350, синих — 775 и красных — 279. То, что точки распределены близко к диагонали, проходящей через точку $(0, 0)$ говорит о том, что изображения близки к квадратной форме, и что их приведение к квадратному размеру (о нем пойдет речь далее) не сильно нарушит пропорции на изображениях.

Ограничивающие рамки. Такой же график для ограничивающих рамок приведен на Рис. 1b. На нем 1082 красных, 265 желтых, 85 зеленых, 19 синих и 3 розовые точки. Из него следует, что ограничивающие рамки имеют бóльшее соотношение сторон. Это логично — сигарета редко расположена под углом 45 градусов, также присутствует погрешность разметки.

(a) Примеры сигарет, заключенных в ограничивающие рамки.

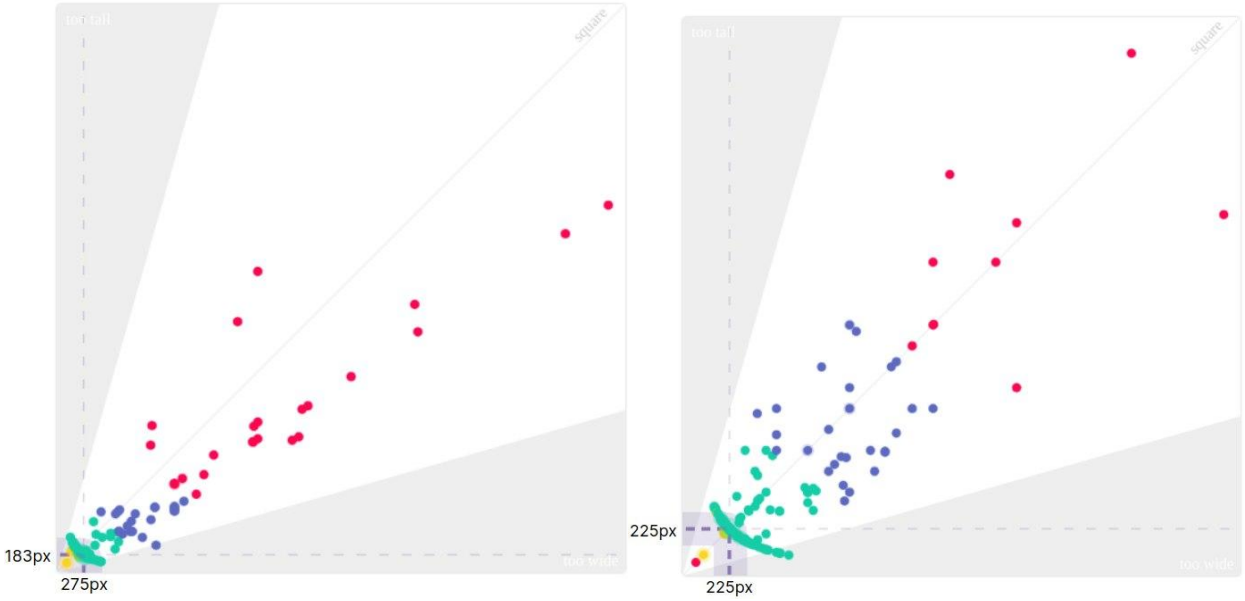


(b) Тепловая карта ограничивающих рамок

Сигарета, расположенная под углом, отличным от 45 градусов, имеет большее отношение своей площади к прямоугольнику, ее ограничивающему. Это значит, что она занимает меньше места и, следовательно, несет в себе больше информации. На рис. 2a и Рис. 2b.

1.2 Набор данных для задачи классификации

Для задачи классификации собрано около 1400 изображений, разделенных в равных пропорциях на два класса: сигарета и не сигарета. Первый класс содержит в себе изображения сигарет, которые выбраны так, чтобы сигарета занимала достаточно большую площадь изображения, как на Рис. 3. Большинство сигарет имеют классический вид (Рис. 3a: белая бумага, оран-



(a) Класс **сигарета**. **(b)** Класс **не сигарета**.

Рис. 2: Визуальное представление размеров набора данных для задачи классификации. На графиках отмечены медианные значения высоты (по вертикали) и ширины (по горизонтали)

жевый фильтр, но в датасет также добавлены полностью белые сигареты 3b и в меньшем объеме сигареты нестандартных цветов 3c.

Второй класс содержит изображения предметов, которые теоретически могут быть отнесены моделью классификации к сигаретам, но ими не являющимися. В него входят разнообразные в основном белые предметы — ручки, зажигалки, просто белые полоски и прочее (Рис. 4).

(a) **(b)** **(c)**

Рис. 3: Разнообразие набора изображений класса **сигарета**

(a) Оконная (дверная) ручка **(b)** Пишущая ручка **(c)** Полосатая ткань с мягкой рулеткой

Рис. 4: Пример изображений с предметами, теоретически похожими на сигареты для нейросети.

Глава 2. Модели

2.1 Двухуровневые детекторы

Архитектуры для детектирования принято разделять на две категории: «одноуровневые» — YOLO, SSD и др. и «двухуровневые» — RCNN, Fast-RCNN и Faster-RCNN и др.

«Двухуровневые» нейронные сети используют алгоритмы, разбивающие решения задачи обнаружения объектов на два этапа:

1. Детектирование регионов, которые могут содержать нужный объект;

2. Классификация этих регионов.

Популярные двухэтапные алгоритмы, такие как Fast-RCNN и Faster-RCNN, обычно используют сеть RPN (Region Proposal Network), которая предлагает области интереса, которые могут содержать объекты. Затем выходные данные RPN подаются в классификатор, который классифицирует регионы по классам.

Хоть этот подход и дает точные результаты обнаружения объектов, он имеет две проблемы:

1. Классифицирующая сеть «смотрит» на картинку не полностью, а только на отдельные регионы, что приводит к тому, что некоторые объекты остаются незамеченными;

2. одно и то же изображение просматривается нейросетью несколько раз, что замедляет скорость обнаружения алгоритма.

2.2 YOLO

В рамках архитектуры YOLO две описанные выше проблемы отсутствуют. В целом архитектура YOLO в первых блоках незначительно отличается по логике блоков от других детекторов: на вход подается изображение, далее с помощью CNN (в YOLO используется своя CNN; например, в YOLOv3 — Darknet-53) создаются карты признаков (feature maps), которые определенным

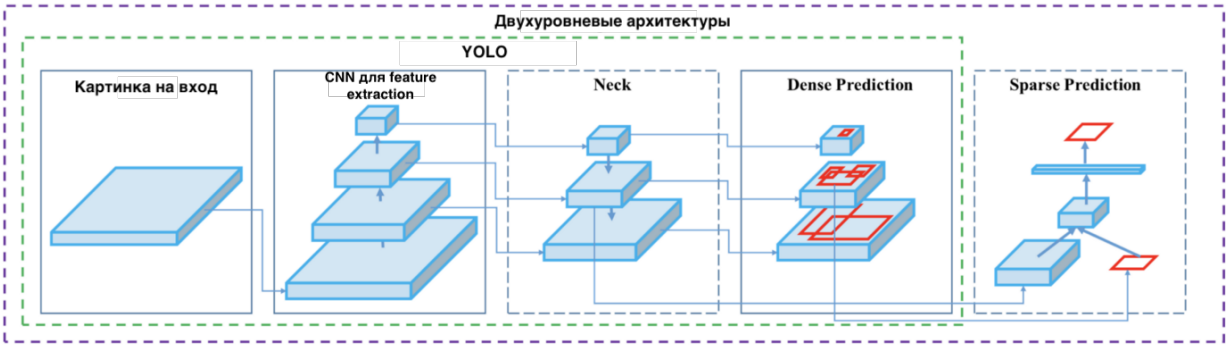


Рис. 5: Двухуровневые модели и YOLO

образом анализируются, выдавая на выходе позиции и размеры ограничивающих прямоугольников (bounding boxes) и классы, которым они принадлежат. На рис. 5 показаны составные части двухуровневых архитектур и YOLO:

Neck (или «шея»). Данный блок создан с целью агрегировать информацию от отдельных слоев из предыдущих блоков (как показано на рисунке выше) для увеличения аккуратности предсказания;

Sparse Prediction. Определение по отдельности регионов и затем их классификация (относится к двухуровневым моделям).

Dense Prediction. Данный блок отличает YOLO от других архитектур. Философия YOLO (от англ. You Only Look Once) состоит в том, чтобы делать все необходимые определения объектов за одно прохождение изображения через одну нейронную сеть. Это становится возможным благодаря с помощью следующего алгоритма:

1. Перед началом обучения нейронной сети все изображения приводятся к одному размеру, чтобы можно было ускорить обучение, подавая их на вход батчами;

2. Изображение делится на клетки размером $a \times a$. В YOLOv3-4 принято делить изображение на клетки размером 13×13. Такие клетки (grid cells) лежат в основе идеи YOLO: каждая клетка является «якорем», к которому прикрепляются ограничивающие прямоугольники (bounding boxes). Для определения объекта вокруг клетки располагается несколько прямоугольников разной формы, поскольку заранее неизвестно, пря-

моугольник какой формы будет наиболее подходящим. Позиции, ширина и высота ограничивающих прямоугольников вычисляются относительно центра клетки. Расположение и размер ограничивающих прямоугольников определяются с помощью техники якорных прямоугольников (anchor boxes) [25] (Рис. 6). Якорные прямоугольники либо задаются изначально вручную, либо их размеры определяются исходя из размеров ограничивающих прямоугольников из обучающего набора, на котором будет тренироваться YOLO. Обычно задают порядка 3 различных якорных прямоугольников, которые будут нарисованы вокруг (или внутри) одной клетки.

3. Изображение вместе с реальными ограничивающими прямоугольниками проходит через нейронную сеть. Для каждой клетки необходимо понять:

   - который из 3 расположенных вокруг клетки якорных прямоугольников является наиболее подходящим (для этого используется метрика IoU — отношение пересечения исходного ограничивающего прямоугольника и якорного прямоугольника к их объединению) и как его размеры можно изменить для того, чтобы он точно вписывал в себя объект.
   - какой объект находится внутри данного якорного прямоугольника и существует ли он вообще.

Таким образом, YOLO предсказывает 5 параметров (для каждого якорного прямоугольника для определенной клетки):

$$
\begin{aligned}
b_x &= \sigma\left(t_x\right) + c_x \\
b_y &= \sigma\left(t_y\right) + c_y \\
b_w &= p_w e^{t_w} \\
b_h &= p_h e^{t_h} \\
\Pr(\text{ object }) * \mathrm{IOU}(b,\ &\text{object }) = \sigma\left(t_o\right), \text{где}
\end{aligned}
$$

t_x, t_y, t_w, t_h — предсказания YOLO; c_x, c_y — координата верхней левой точки целевой клетки (grid cell); p_w, p_h — ширина и высота определенного якорного прямоугольника; b_x, b_y, b_w, b_h — параметры для предсказанного ограничивающего прямоугольника; $\sigma\left(t_o\right)$ — уверенность модели (confidence score).

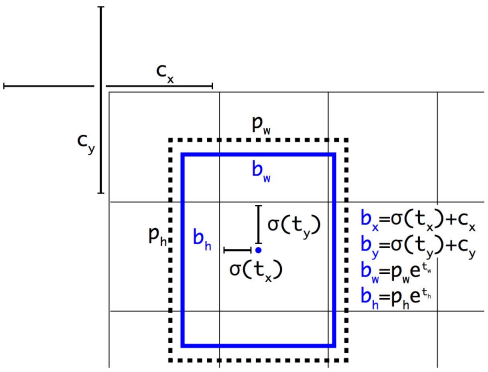


Рис. 6: Якорные прямоугольники.

Задача YOLO — максимально точно предсказать данные параметры, чтобы максимально точно определять объект на изображении. Параметр уверенности модели (confidence score), который рассчитывается для каждого предсказанного ограничивающего прямоугольника, становится фильтром, благодаря которому отсеиваются наименее точные предсказания. Для расчета данного параметра IoU каждого предсказанного ограничивающего прямоугольника умножается на вероятность того, что это определенный объект (вероятностное распределение рассчитывается во время обучения нейронной сети). Затем выбирается лучшая вероятность из возможных, и если число после умножения превышает определенный порог, то предсказанный ограничивающий прямоугольник на изображении сохраняется. Таким образом, прямоугольников может быть несколько. Далее с помощью техники NMS (non-max suppression) можно отфильтровать ограничивающие прямоугольники таким образом, чтобы для одного объекта остался только один из предсказанных — рис. 7.

Рис. 7: Non-max suppression

2.2.1 Архитектура YOLOv5

Первая версия YOLO не была лишена недостатков, а последующие обновления значительно повысили производительность данной модели. Так, изначально в YOLO не использовалась техника якорных коробок, она была добавлена лишь во второй версии модели. Также во второй версии была добавлена батч-нормализация [8].

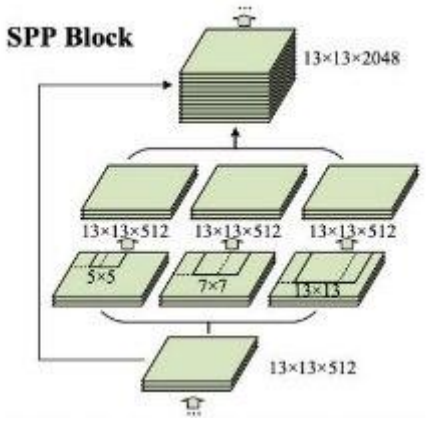


Рис. 8: Spatial Pyramid Pooling блок.

В третьей версии модели YOLOv3 в качестве **backbone** подсети вместо DarkNet-19 используется более сложная DarkNet-53. Архитектурная новизна YOLOv3 позволила делать прогнозы в трех разных масштабах, что становится возможным благодаря делению исходного изображения не только на ячейки размера 13×13, но и на ячейки других размеров (Рис. 8 и правый пунктирный прямоугольник на Рис. 9). Деление в данном случае условное, поскольку происходит в глубине сети в слоях 82, 94, 106.

Последующие версии модели нельзя назвать официальными, поскольку они представлены сторонними разработчиками. YOLOv4 (Рис. 9) предлагает добавление остаточных соединений (Residual Connections), известных благодаря моделям семейства ResNet (Рис. 9, левый пунктирный прямоугольник), перекрестной мини-пакетной нормализации, самосостязательного обучения, которое добавляет шум при обучении модели с целью сделать модель устойчивой к шумам и/или атакам.

YOLOv5 представляет собой проект с открытым исходным кодом, состоящий из семейства моделей, основанных на модели YOLO, предварительно обученных на наборе данных COCO [16]. Данная модель не отличается значительным образом от предыдущей версии, а наибольшим вкладом разработчиков в YOLOv5 является перевод фреймворка Darknet на платформу PyTorch. Фреймворк Darknet написан главным образом на языке Си и предлагает детальный контроль над операциями внутри сети, что может замедлить

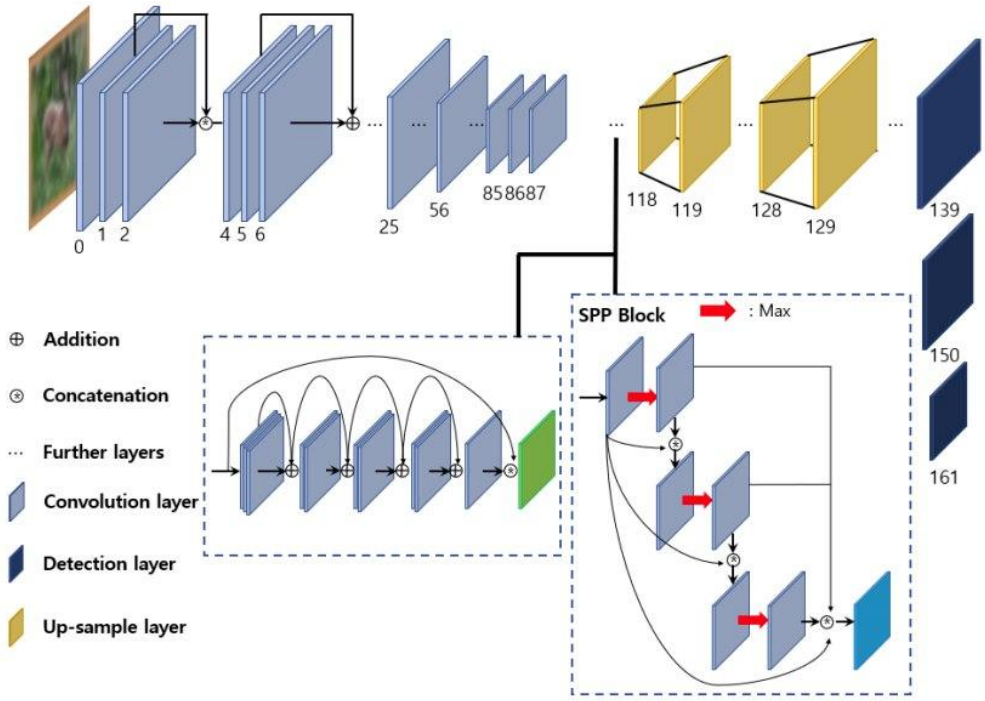


Рис. 9: Архитектура YOLOv4 [17]

внедрение новых исследовательских идей, поскольку существует необходимость добавлять пользовательские вычисления градиента.

Самая большая версия модели YOLOv5x и самая маленькая YOLOv5n не отличаются слоями. Отличие между ними, как и между другими версиями, заключается в двух гиперпараметрах — множителях масштабирования ширины α и глубины ρ сети. (Рис. 10).

Множитель ширины отвечает за количество каналов в каждом слое. Множитель глубины отвечает за пространственные размеры входных тензоров.

Например, $\alpha = 1, \rho = 1$ дает архитектуру YOLOv5l, а $\alpha = 0.33, \rho = 0.5$ — YOLOv5s — архитектуру с уменьшенным в три раза числом каналов на выходе каждого блока и уменьшенной вдвое высотой и шириной карты признаков, подаваемой на вход каждому слою.

2.2.2 Функция потерь YOLO

YOLO предсказывает несколько ограничивающих рамок на каждую клетку сетки (grid cell). Чтобы вычислить потери для истинно положительного предсказания, нужно, чтобы только один из них относился к объекту.

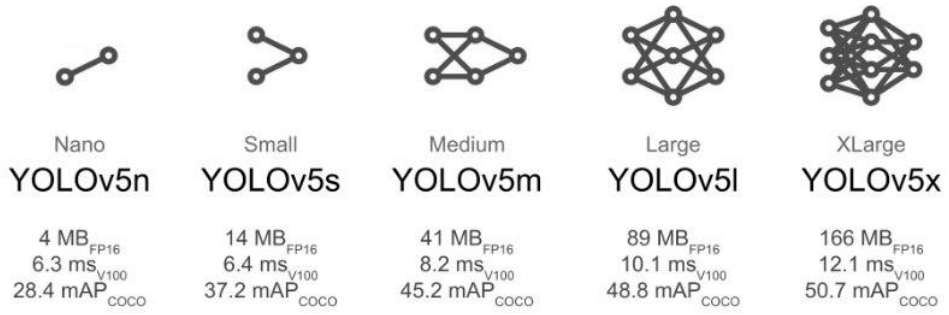


Рис. 10: Модели семейства YOLOv5 [20].

Для этой цели выбирается тот, у которого самое высокое значение IoU 1 с истинным. Функция потерь включает три составляющие:

- Ошибка классификации (classification);
- Ошибка локализации (localization);
- Ошибка уверенности (confidence);

Ошибка классификации.

Если объект обнаружен, то потеря классификации в каждой ячейке (grid cell) представляет собой бинарную перекрестную энтропию между вероятностями класса для каждого класса:

$$\ell_{\text{cls}} = \sum_{i=0}^{S^2} \mathbb{1}_i^{\text{obj}} \sum_{c \in \text{ classes}} -(p_i(c) \log(\hat{p}_i(c)) + (1 - p_i(c)) \log(1 - \hat{p}_i(c)))$$

где

$\mathbb{1}_i^{obj} = 1$ если объект содержится в ячейке i, иначе 0;

$\hat{p}_i(c)$ — вероятность содержания класса c в ячейке i.

Ошибка локализации.

Измеряет ошибки прогнозируемых положений и размеров тех ограничивающих рамок, которые ответственны за обнаружение объекта. Для этой части функции в YOLOv5 используется $\mathcal{L}_{IoU} = 1 - IoU$. Для предсказанного и истинного ограничивающих прямоугольников с координатами нижней ле-

вой и верхней правой точек (x_1, y_1), (x_2, y_2) и $(\hat{x}_1, \hat{y}_1)$, $(\hat{x}_2, \hat{y}_2)$ соответственно ошибка локализации может быть вычислена:

$$\ell_{\text{loc}} = \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \cdot IoU^j,$$

где

$\mathbb{1}_{ij}^{obj} = 1$, если рамка j в ячейке i соответсвует детектируемому объекту, иначе 0;

λ_{coord} — вес, отражающий важность этой составляющей

и

$$IoU^j = \frac{Intersection^j}{Union^j},$$

$$Intersection^j = (\min(x_2^j, \hat{x}_2^j) - \max(x_1^j, \hat{x}_1^j) * (\min(y_2^j, \hat{y}_2^j) - \max(y_1^j, \hat{y}_1^j),$$
$$Union = w_1^j * h_1^j + w_2^j * h_2^j - Intersection^j + \epsilon,$$
$$w = x_2 - x_1,$$
$$h = y_2 - y_1.$$

Ошибка уверенности. Если в прямоугольнике обнаружен объект, ошибка уверенности (измерение объективности прямоугольника) вычисляется как перекрестная энтропия:

$$\ell_{\text{conf}} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} (-(C_i \log(\hat{C}_i) + (1 - C_i) \log(1 - \hat{C}_i)))$$

где $\hat{C}_i$ показатель уверенности для рамки j ячейки i. $1_{ij}^{obj} = 1$, если рамка j ячейки i отвечает за обнаружение объекта, иначе 0.

2.3 Оценка качества модели

Модели обнаружения объектов решают две задачи:

- Классификация: определить, присутствует ли объект на изображении,

и указать его класс;

- Локализация: предсказать координаты ограничивающей рамки вокруг объекта, если он присутствует на изображении. В данном случае сравниваются координаты истинных и предсказанных ограничивающих рамок.

Из этого следует необходимость оценивать качество как классификации объектов, так и локализации их на изображении. Для обнаружения объектов используется концепция IoU. В рамках решаемой задачи для IoU устанавливается пороговое значение k (если $k = 1$, значит предсказанный и истинный прямоугольники совпадают):

- Если $IoU \geq k$, предсказанный объект считается истинно положительным (True Positive — **TP**);
- При $IoU < k$, классификация считается неправильной, а предсказанный объект — ложно положительным (False Positive — **FP**);
- Если объект присутствует на изображении, но модель не смогла его обнаружить, он классифицируется как ложно отрицательный (False Positive — **FN**);
- Истинно отрицательной (True Negative — **TN**) может считаться любая часть изображения, на которой модель не выявила объект и которая никакой объект не содержит.

Обычно порог для IoU устанавливается в диапазоне $[0.5, 0.95]$.

В качестве показателей для оценки качества модели используются $Precision$ и $Recall$, которые в свою очередь рассчитываются с помощью истинно положительных (**TP**), ложно положительных **FP** и ложно отрицательных (FN) результатов:

$$\text{Precision} = \frac{\text{True Positive Number}}{\text{True Positive Number } + \text{ False Positive Number}},$$
$$\text{Recall} = \frac{\text{True Positive Number}}{\text{True Positive Number } + \text{ False Negative Number}},$$

где Number означает количество. Иначе говоря, $precision$ есть отношение правильно классифицированных сигарет ко всем объектам, предсказанным как сигарета, а $recall$ — отношение правильно классифицированных сигарет ко всему множеству известных сигарет.

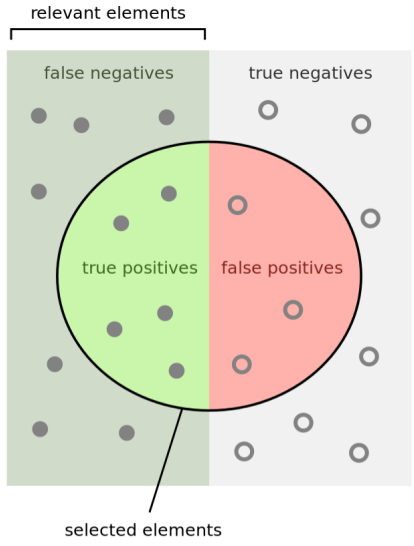


Рис. 11: Визуальное представление TP, FN, FP, TN

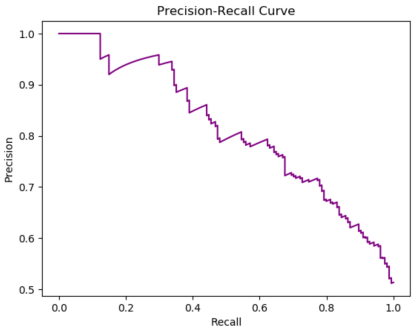


Рис. 12: $precision - recall$ кривая.

Также необходимо учитывать показатель уверенности предсказания ($confidence\ score$) для каждого объекта, обнаруженного моделью на изображении. Ограничивающие рамки, которые выше заданного порогового значения, считаются положительными, а все, что ниже — отрицательными.

Также необходимо учитывать показатель уверенности предсказания ($confidence\ score$) для каждого объекта, обнаруженного моделью на изображении. Ограничивающие рамки, которые выше заданного порогового значения, считаются положительными, а все, что ниже — отрицательными.

Следующим шагом является построение *precision-recall* (PR) кривой и вычислением площади под ее графиком. Для этого используется метод интерполированной точности (Interpolated Accuracy) из [7]. График PR-кривой — убывающий (в среднем), всегда есть компромисс между *precision* и *recall*. Увеличение одного приведет к уменьшению другого. Согласно [7], интерполированная точность (precision) $p_{interp}(r)$, вычисляется для каждого из 11 значений полноты (recall) $r \in [0, 0.1, 0.2, ..., 1]$:

$$p_{\text{interp}}(r) = \max_{\tilde{r}:\tilde{r} \geq r} p(\tilde{r})$$

Тогда средняя точность (Average Precision) может быть вычислена:

$$\text{AP} = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots, 1.0\}} p_{\text{interp}}(r)$$

Значение AP вычисляется для каждого класса. mAP (mean average precision) рассчитывается путем нахождения средней точности (AP) для каждого класса, а затем усредняется по всем классам:

$$\text{mAP} = \frac{1}{|N_{classes}|} \sum_{c \in classes} \text{AP}_c$$

Для детектирования сигарет, т.е. одного класса, метрики mAP и AP совпадают, далее будет использоваться обобщенное понятие mAP. Очевидно, что различным пороговым значениям для IoU соответствуют различные значения mAP. Для указания mAP с конкретным пороговым значением t для IoU используется обозначение mAP@t. В решаемой задаче детектирования сигарет модель оценивается по четырем показателям:

1. Precision;
2. Recall;
3. mAP@0.5;
4. mAP@0.5:0.05:0.95 — получается путем вычисления AP для каждого порогового значения $IoU_{\text{treshold}} \in [0.5, 0.55, ..., 0.9, 0.95]$ и усреднением всех значений:

$$\text{mAP@0.5:0.05:0.95} = \frac{1}{|K|} \sum_{k \in K} \text{mAP@}k,$$

где

$$K = [0.5, 0.55, ..., 0.9, 0.95]$$

2.4 Метрика $\mathcal{L}_{IoU}$ и ее производные

Детектирование объектов включает две подзадачи: классификацию объектов и локализацию объектов. Локализация объектов опирается на модуль регрессии ограничивающей рамки — Bounding Box Regression (BBR) Module.

Любая ограничивающая рамка однозначно может задаваться четырьмя значениями — например, координаты левой нижней (x_1, y_1) и верхней правой (x_2, y_2) точек прямоугольника. Либо это могут быть координаты центра (x, y), ширина w и высота h. Регрессия этих значений — задача, решаемая BBR модулем. Он использует функцию потерь на основе предсказанной ограничивающей рамки и истинной ограничивающей рамки (известной заранее). Между двумя прямоугольниками сначала вычисляется метрика, отражающая степень сходства, а затем это значение инвертируется, чтобы значение функции потерь было тем больше, чем более непохожи друг на друга прямоугольники согласно метрике. Инверсия производится путем вычитания метрики из единицы или применением $-ln(\cdot)$.

Intersection over Union. Первой после евклидова расстояния ℓ_2 предложенной метрикой является IoU (Intersection over Union) [1] (Рис. 13a), которая для сравнения подобия между двумя произвольными формами (объемами) $A, B \subseteq \mathbb{S} \in \mathbb{R}^n$ (в случае детектирования $n = 2$) вычисляется:

$$IoU = \frac{|A \cap B|}{|A \cup B|}, \tag{1}$$

соответствующая функция потерь:

$$\mathcal{L}_{IoU} = 1 - IoU.$$

Две привлекательные особенности, которые делают эту меру сходства популярной для оценки многих задач компьютерного зрения 2D/3D, заключаются в следующем:

- $\mathcal{L}_{IoU}$ — метрика по определению [2]. Это означает, что $\mathcal{L}_{IoU}$ выполняет все свойства метрики — неотрицательность, аксиома тождества, симметрия и неравенство треугольника.

- IoU инвариантна к масштабу. Таким образом, сходство между двумя произвольными формами A и B не зависит от масштаба их пространства $\mathbb{S}$ (доказательство приведено в [3]).

Однако у IoU выявлен серьезный недостаток:

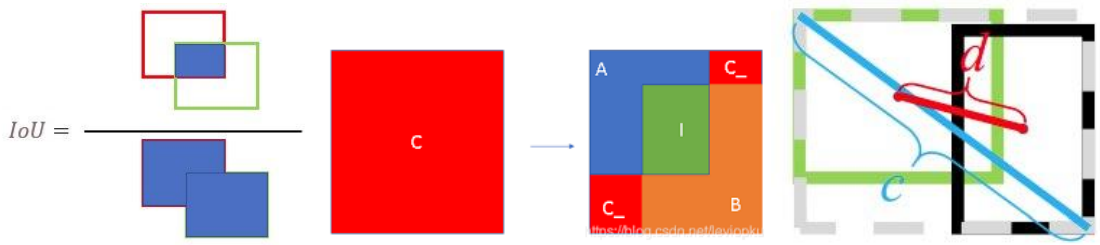
- Если $|A \cap B| = 0$, то $\mathrm{IoU}(A, B) = 0$. В этом случае IoU не отражает, находятся ли две фигуры рядом друг с другом или очень далеко друг от друга.

Generalized IoU. Для решения этой проблемы была предложена усовершенствованная метрика $GIoU$ [3] (Рис. 13b). Для двух произвольных выпуклых форм (объемов) $A, B \subseteq \mathbb{S} \in \mathbb{R}^n$ сначала находится наименьший прямоугольник $C \subseteq \mathbb{S} \in \mathbb{R}^n$, охватывающий как A, так и B. Тогда:

$$GIoU = IoU - \frac{|C \backslash (A \cup B)|}{|C|},$$
$$\mathcal{L}_{GIoU} = 1 - IoU + \frac{|C \backslash (A \cup B)|}{|C|}.$$

Distance IoU. Функция потерь $DIoU$ [4] (Рис. 13c) включает нормализованное расстояние между предсказанным и истинным ограничивающим прямоугольниками:

$$\mathcal{R}_{DIoU} = \frac{\rho^2\left(\mathbf{A}, \mathbf{B}\right)}{c^2}$$
$$\mathcal{L}_{DIoU} = 1 - IoU + \frac{\rho^2\left(\mathbf{A}, \mathbf{B}\right)}{c^2}$$



(a) IoU. **(b)** GIoU. **(c)** DIoU.

Рис. 13: Визуализация метрики IoU и ее производных.

Она не только наследует свойства IoU и $GIoU$, но также полезна в случаях, когда они неэффективны:

- $DIoU$ напрямую минимизирует расстояние между двумя прямоугольниками и, таким образом, сходится намного быстрее, чем $\mathcal{L}_{GIoU}$, особенно в случае, когда прямоугольники не пересекаются.

- В случае, когда один прямоугольник полностью включен в другой, минимизация $\mathcal{L}_{DIoU}$ может сделать регрессию очень быстрой, в то время как $\mathcal{L}_{GIoU}$ почти переходит в $\mathcal{L}_{IoU}$, т. к. $|C - A \cup B| \to 0$.

Однако $DIoU$ по-прежнему упускает из виду согласованность пропорций для сторон ограничивающих прямоугольников, что является важным геометрическим фактором.

Complete IoU. Для решения оставшейся проблемы предложена усовершенствованная метрика $CIoU$ [4], [5]. Таким образом, подводя итог, авторы выделяют три наиболее важных геометрических фактора: площадь пересечения, расстояния между центрами прямоугольников и соотношение сторон прямоугольников. Следовательно, исходя из $\mathcal{L}_{DIoU}$, предлагается $\mathcal{L}_{CIoU}$:

$$\mathcal{R}_{CIoU} = \mathcal{R}_{DIoU} + \alpha v = \frac{\rho^2\left(\mathbf{A}, \mathbf{B}\right)}{c^2} + \alpha v,$$

где α — положительный параметр, а v отражает отношение отношений сторон прямоугольников:

$$v = \frac{4}{\pi^2}\left(\arctan\frac{w^A}{h^A} - \arctan\frac{w^B}{h^B}\right)^2.$$

Тогда функция потерь:

$$\mathcal{L}_{CIoU} = 1 - IoU + \frac{\rho^2\left(\mathbf{A}, \mathbf{B}\right)}{c^2} + \alpha v$$

Параметр α — компромиссный параметр, определяющий приоритет для регрессии. Он определяется как:

$$\alpha = \begin{cases} 0, & \text{если } IoU < 0.5 \\ \frac{v}{(1-IoU)+v}, & \text{если } IoU \geq 0.5 \end{cases},$$

в результате чего фактор площади пересечения играет бóльшую роль для регрессии, особенно для случаев, в которых прямоугольники не пересекаются.

2.5 Классификатор MobileNetV2

Одним из ключевых критериев выбора модели для классификации является ее относительно небольшой размер, поскольку:

- имеется небольшое количество данных для обучения;
- размеры изображений (контент внутри ограничивающих прямоугольников), поданных ей на вход, относительно невелики: 100×100 пикселей;
- классификатор как дополнительная часть глобальной модели не должен быть больше модели детектирования.

Одной из таких моделей является MobileNetV2 [22] — третья версия, представленная компанией Google в 2019 году.

2.5.1 Эффективные сверточные блоки

В MobileNetV1 с целью уменьшить количество параметров введен слой со сверткой по глубине (depth-wise separable convolutions). Данный слой устроен следующим образом. Обычный сверточный слой состоит из набора фильтров $D_k * D_k * C_{in}$, где D_k — это размер ядра свертки, а C_{in} — количество каналов на входе. Общая вычислительная сложность сверточного слоя составляет $D_k * D_k * C_{\text{in}} * D_f * D_f * C_{out}$, где D_f — это высота и ширина слоя карты признаков (пространственные размеры входного и выходного тензоров совпадают), а C_{out} — число каналов на выходе.

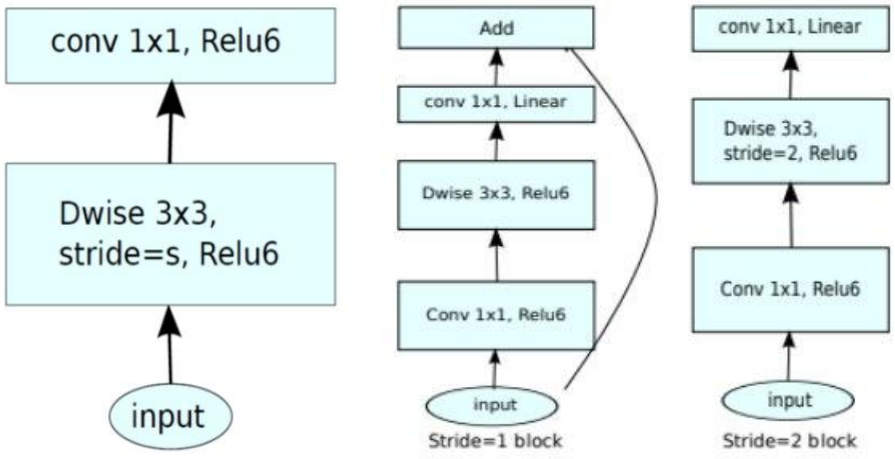
Идея depthwise separable convolution состоит в том, чтобы разложить подобный слой на depth-wise свертку, которая представляет из себя поканальный фильтр, и 1×1 свертку (также называемую pointwise convolution). Суммарное количество операций для применения такого слоя: $(D_k * D_k + C_{out}) * C_{in} * D_f * D_f$.

Таким образом, сверточный блок MobileNetV1 состоит из двух слоев: **depthwise свертки** и 1×1 **свертки**. Сверточная часть MobileNetV1 состоит из одного обычного сверточного слоя с 3×3 сверткой в начале и тринадцати сверточных блоков с постепенно увеличивающимся числом фильтров и понижающейся пространственной размерностью тензора.

Как и в первой версии, в MobileNetV2 используются сверточные блоки с шагом 1 (на Рис. 14b слева) и с шагом 2 (на Рис. 14b справа). Блоки с шагом 2 предназначены для снижения пространственной размерности тензора: в отличие от блока с шагом 1, они не имеют остаточных соединений (residual connections). Этот блок, называемый авторами расширяющим сверточным блоком (expansion convolution block или bottleneck convolution block with expansion layer), состоит из трёх слоёв (Рис. 14b):

1. Сперва производится pointwise свертка с большим числом каналов — слой расширения (expansion layer).

   На входе данный слой принимает тензор размерности $D_f * D_f * C_{in}$, а на



(a) MobileNetV1 блок **(b)** MobileNetV2 блок

Рис. 14: Сверточные блоки первой и второй версий MobileNetV.

выходе выдает тензор $D_f * D_f * (t * C_{in})$, где t — новый гиперпараметр, названный уровнем расширения (expansion factor). Авторы рекомендуют задавать этому гиперпараметру значение от 5 до 10, где меньшие значения лучше работают для более маленьких сетей, а большие для более крупных (в самой статье во всех экспериментах принимается $t = 6$). Этот слой создает отображение входного тензора в пространстве большой размерности, называемое целевым многообразием (manifold of interest);

2. Далее следует depthwise convolution с ReLU6-активацией:

$$\mathrm{ReLU6}(x) = \min(\max(0, x), 6).$$

Этот слой вместе с предыдущим уже образует строительный блок MobileNetV1. На входе этот слой принимает тензор размерности $D_f * D_f * (t * C_{in})$, а на выходе выдает тензор $(D_f/s) * (D_f/s) * (t * C_{in})$, где s — шаг свертки (stride), так как depthwise свертка не меняет число каналов.

3. В конце следует 1x1-свертка с линейной функцией активации, снижающая число каналов. Авторы статьи выдвигают гипотезу, что целевое многообразие высокой размерности, полученное после предыдущих шагов, можно «уложить» в подпространство меньшей размерности без потери полезной информации, что и делается на этом шаге.

На входе такой слой принимает тензор размерности $(D_f/s) * (D_f/s) * (t * C_{in})$, а на выходе выдает тензор $(D_f/s) * (D_f/s) * C_{\text{out}}$, где C_{out} – количество каналов на выходе блока.

2.5.2 Архитектура

Архитектура MobileNetV2 представлена в Таблице 1.

- Здесь t — коэффициент расширения, C_{out} — количество выходных каналов, n — число повторений, s (stride) — шаг. Ядра 3×3 используются для пространственной свертки;

Input	Operator	t	C_{out}	n	s
$224^2 \times 3$	conv2d	–	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	–	1280	1	1
$7^2 \times 1280$	avgpool 7×7	–	–	1	–
$1 \times 1 \times 1280$	conv2d 1x1	–	k	–	

Таблица 1: Архитектура модели MobileNetV2.

- conv2d — слой стандартной 2d свертки, bottleneck — описанный выше строительный блок (expansion conv block), avgpool — Average Pooling — среднее по всем значениям квадрата 7×7.

2.6 Аугментация

Решаемые задачи детектирования и классификации изображений — задачи обучения с учителем. Чтобы создать мощную модель, необходимо иметь большое количество данных. Если доступных размеченных данных нет — приходится вручную создавать собственный репрезентативный набор. Одним из ключевых аспектов создания датасета является аугментация (дополнение) данных. Аугментация — процесс изменения входных данных при сохранении контекста. В случае с изображениями это означает перевернуть изображение, добавить к нему шум, отразить, изменить цветовую температуру и т.д. Аугментация данных позволяет имитировать различные сценарии для одного и того же объекта в реальности. Преимущество подхода в том, что можно изменять одно изображение сколько угодно раз, случайно смешивая различные методы дополнения. Каждое из дополненных изображений выглядит для необученной модели по-новому, что позволяет ей лучше обобщать. Для задачи детектирования в работе используются аугментации HSV, при которых изображение переводится из палитры RGB в HSV (Hue, Saturation, Value) и варьирование этих параметров, что позволяет менять оттенок, насыщенность

и яркость изображения соответственно. Также в работе используется смешение изображений [19] и случайная вставка объектов в произвольное место на изображении вместе с рамками, их ограничивающими.

2.7 Перенос обучения

Transfer Learning (Перенос обучения) [13], [14] — парадигма для предотвращения переобучения — явления, при котором построенная модель хорошо объясняет примеры из обучающей выборки, но относительно плохо работает на примерах, не участвовавших в обучении (на примерах из тестовой выборки). Transfer Learning работает путем обучения сети на большом наборе данных, таком как ImageNet [12] для классификации или COCO [16] для детектирования, а затем использует эти веса в качестве начальных весов в новой задаче классификации или детектирования. Обычно копируются только веса в сверточных слоях, а не вся сеть, включая полносвязные слои. Это эффективно, поскольку большинство датасетов изображений имеют общие пространственные характеристики низкого уровня, которые лучше изучаются с помощью большого количества данных.

Глава 3. Эксперименты и результаты

3.1 Параметры обучения

В качестве основной модели задачи детектирования выбрана версия YOLOv5s. Обучение одной модели в 300 эпох занимает в среднем 6 часов. Все изображения приводятся к размеру 640×640. Согласно анализу датасета в 1.1 такое преобразование не сильно нарушает пропорции, так как форма изображений близка к квадратной. Основной метрикой сравнения выбирается mAP@0.5.

3.2 Решение на базе YOLOv5

3.2.1 Результаты и валидация

Для решения задачи детектирования в рамках работы выбрана модель YOLOv5s. Графики обучения представлены на Рис. 15 (L_{IoU} используется в функции потерь). Здесь определенные в 2.3 следующие показатели (со значениями на тестовом множестве):

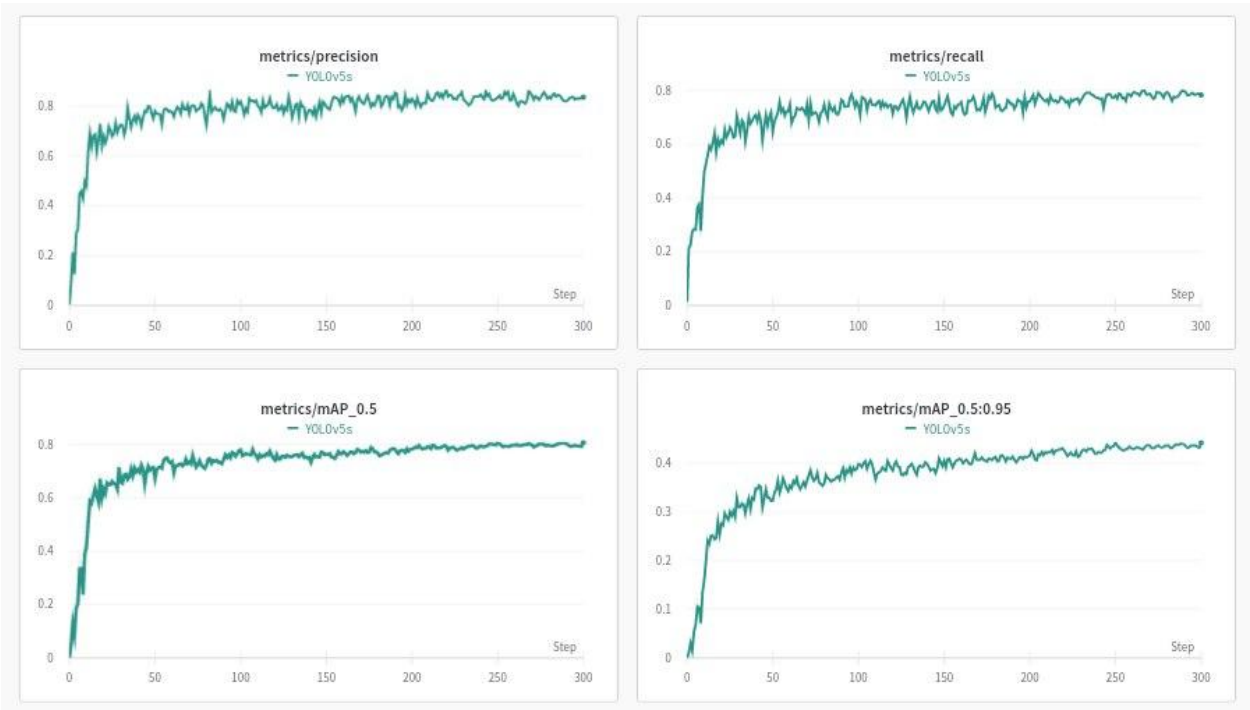


Рис. 15: Метрики качества модели для валидационного множества на каждой эпохе обучения (ось X).

- Точность (accuracy) — 0.83;
- Полнота (recall) — 0.78;
- Средняя точность для порога $IoU = 0.5$ (mAP@0.5) — 0.79;
- Средняя точность для совокупности порогов IoU от 0.5 до 0.95 с шагом 0.05 (mAP@0.5:0.05:0.95) — 0.43.

Чтобы убедиться в способности модели обобщать результаты, обучение и тестирование проведено несколько раз на одних и тех же данных, случайно разделенных на тренировочное и тестовое подмножества. Результаты показывают, что случайный выбор подмножеств не влияет на результаты модели. Последущие сравнения моделей выполнены на фиксированных тренировочном, валидационном и тестовом множествах.

3.2.2 Параметры аугментации

Для сравнения использованы две техники аугментации: смешивание изображений и вставка объекта (вместе с контентом внутри ограничивающей рамки) в произвольное место на изображении. Параметры аугментаций представлены в Таблице 3. Результаты приведены в Таблице 2. Из таблицы и 16 следует, что набор аугментаций со случайной вставкой объектов заметно повышает полноту предсказаний модели (Recall). Также стоит отметить что mAP@0.5 принимает наибольшее значение.

Результаты на тестовом множестве			
	№1	№2	№3
Precision	0.83	0.91	0.89
Recall	0.78	0.76	0.82
mAP@0.5	0.79	0.84	0.87
mAP@0.5:0.95	0.43	0.44	0.45

Таблица 2: Результаты моделей, обученных с разными параметрами аугментаций

Параметры аугментации			
	№1	№2	№3
HSV Тон (коэф)	0.015	0.015	0.015
HSV Насыщенность (коэф)	0.7	0.7	0.7
HSV Значение (коэф)	0.4	0.4	0.4
Поворот изображения (градусы)	0.0	0.0	0.0
Перемещение изображения вдоль оси X или Y (коэф)	0.1	0.1	0.1
Мозаика (вероятность)	1.0	1.0	1.0
Увеличение масштаба (доля)	0.5	0.9	0.9
Отражение по горизонтали (вероятность)	0.5	0.5	0.5
Отражение по вертикали (вероятность)	0.0	0.0	0.0
Смешивание (вероятность)	0.0	0.1	0.1
Копирование и вставка содержимого рамки (вероятность)	0.0	0.0	0.1

Таблица 3: Таблица параметров аугментации для каждого из трех экспериментов.

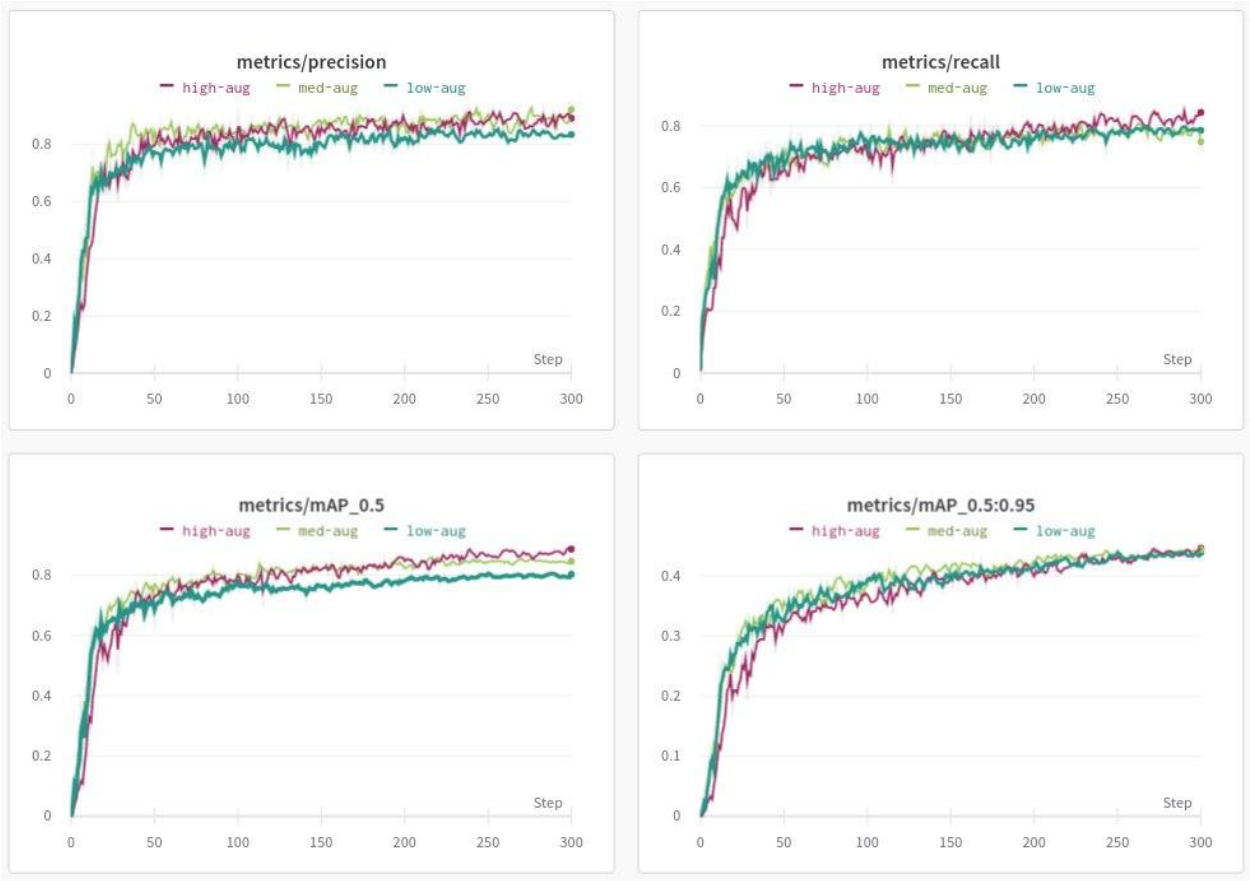


Рис. 16: Эксперименты с параметрами аугментаций.

3.2.3 Модификации функции потерь

Каждая из метрик, описанных в 2.4 (помимо IoU — $GIoU, DIoU, CIoU$), — может быть использована как часть составной функции потерь YOLO. Единственной из них, что учитывает соотношение сторон истинной и предсказанной ограничивающих рамок, является $CIoU$. Она по предположению должна лучше всего подходить для данной задачи, так как соотношение сторон есть тангенс угла между сигаретой и горизонтальной осью с точностью до отражения по вертикали. В ходе работы протестированы все 4 метрики. Результаты обучения, проведенного независимо из фиксированного начального состояния в 130 эпох, отражены на Рис. 17. Из него следует, что метрика $CIoU$ в данной задаче не дает преимущества. Однако, следуя тенденциям, для опредленности далее используется метрика $CIoU$.

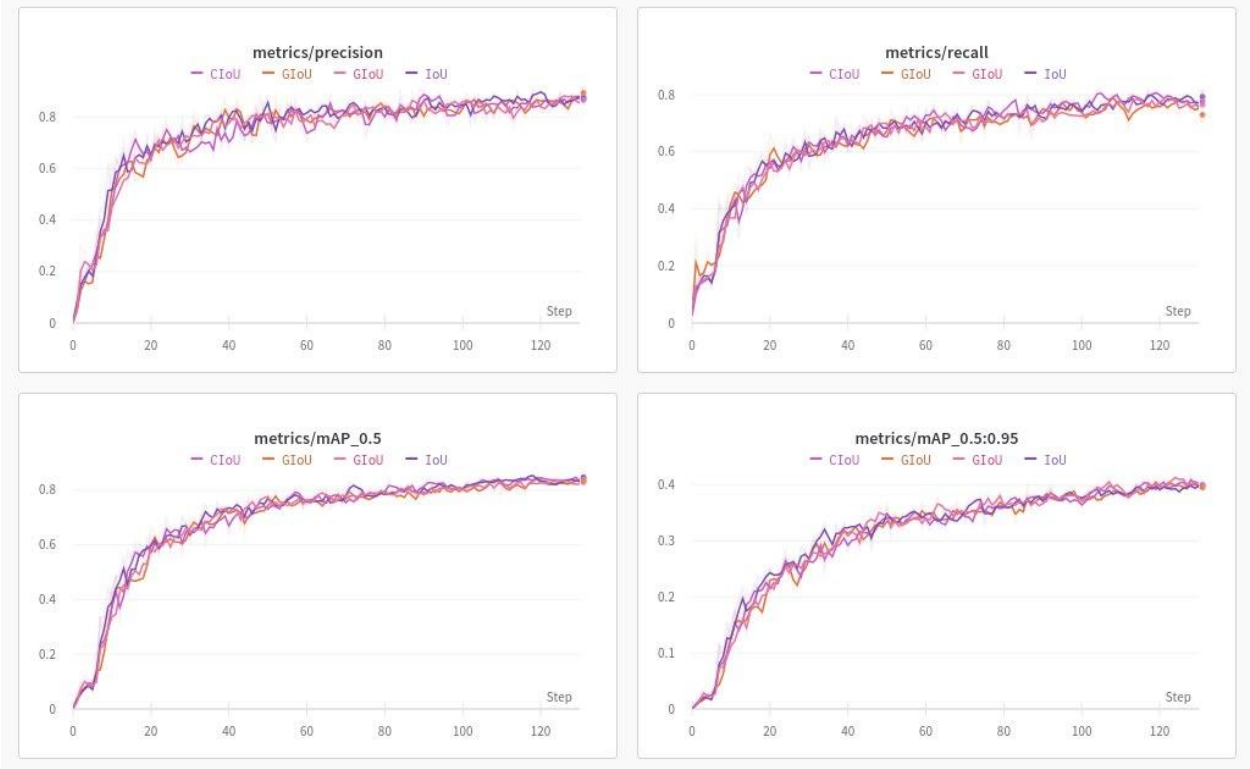


Рис. 17: Графики обучения YOLOv5 с $\mathcal{L}_{IoU}$ и ее различными модификациями.

3.2.4 Анализ результатов YOLOv5

Лучшей на текущий момент моделью выбрана модель, обученная в 3.2.2 со следующими показателями: Precision — 0.89, Recall — 0.82, mAP@0.5 — 0.87, mAP@0.5:0.95 — 0.45. На Рис. 18 показаны результаты лучшей из обученных моделей. Под лучшей понимается модель с наибольшим значением mAP.

И хотя 89% сигарет распознаны правильно (метрика precision), у модели есть существенный недостаток. Некоторые объекты, интуитивно более различимые как сигареты, имеют намного меньшую степень уверенности (confidence score) по сравнению с объектами, похожими на них в меньшей степени. В основном такие объекты имеют относительно малый размер и без контекста трудно различимы даже для человека. Это свидетельствует о том, что модель в таких случаях доверяет контексту. Повышение порога показателя уверенности отсеивает большинство правильно распознанных сигарет (True Positives) (см. Рис. 19) и не дает более высоких результатов.

(a)
Сигарета занимает много места на изображении.

(b)
Сигарета занимает мало места на изображении.

(c)
Сигарета неклассического цвета — бурого.

(d)
Две сигареты.

Рис. 18: Результаты лучшей согласно метрике mAP@0.5 (0.87) модели на различных примерах.

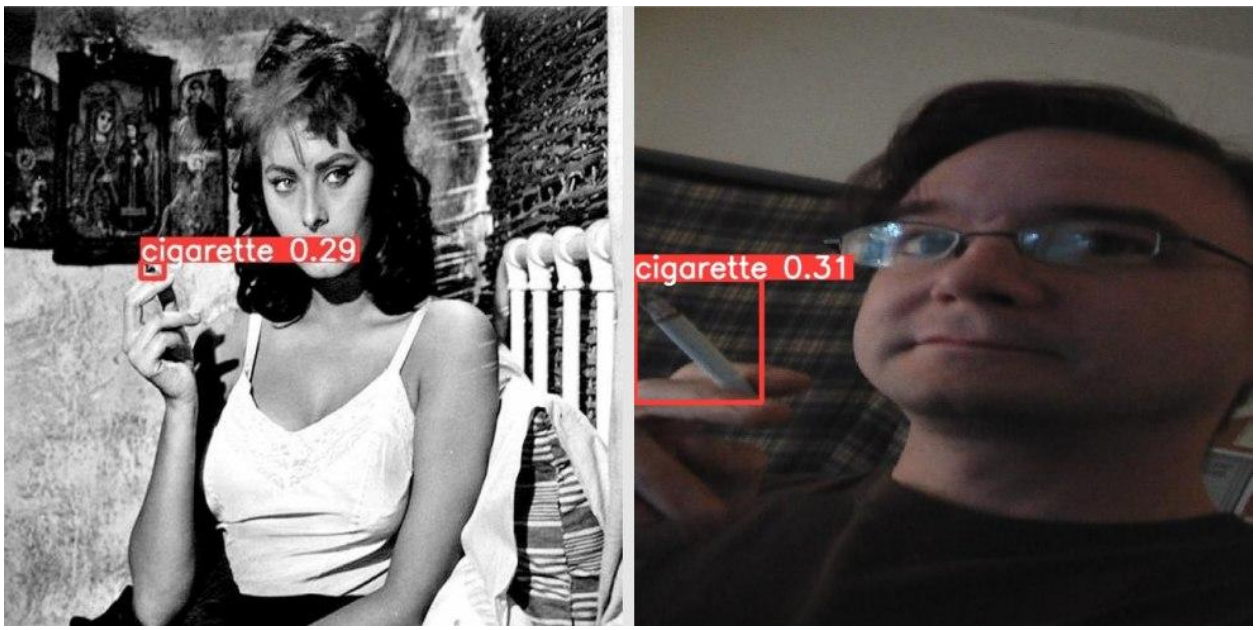


Рис. 19: Примеры правильно распознанных сигарет с низким значением показателя уверенности ($confidence\_score$).

3.3 Классификатор

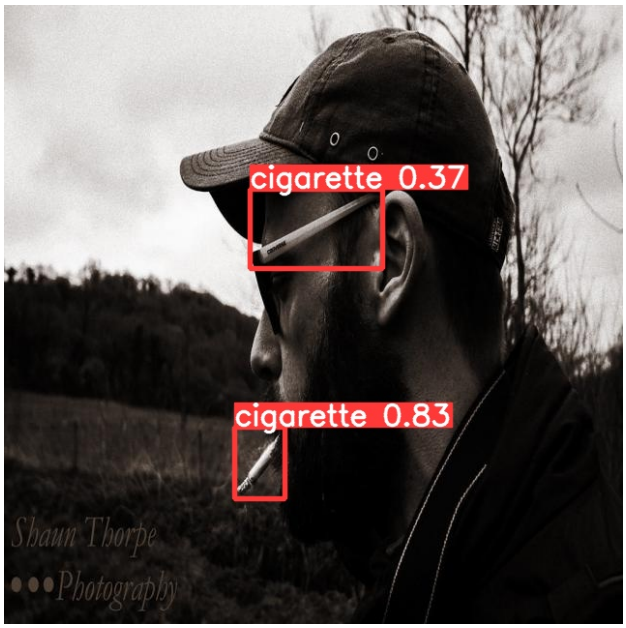
Построенная модель не лишена недостатков. На рис. 20 представлены примеры ложно-положительных распознаваний (False Positives). Такие распознавания отличаются между собой предметами, содержащимися внутри предсказанных прямоугольников, и показателем уверенности (в скобках):

- Рис. 20a — рисунок красных губ с белыми «бликами» на футболке (0.2);
- Рис. 20b — дужка очков (0.37);
- Рис. 20c — полоски на рубашке $([0.42, 0.52])$;
- Рис. 20d — зажигалка (0.73).

Увеличение значения показателя уверенности напрямую ведет к уменьшению не только ложно положительных (FP), но и истинно положительных (TP), а следовательно — к уменьшению метрики точности (precision).

Для решения этой проблемы и уменьшения количества ложных распознаваний как на Рис. 20 предлагается использование дополнительного классификатора.

(a)
$conf = 0.2$



(b)
$conf = 0.37$

(c)
$conf = [0.42, 0.52]$

(d)
$conf = 0.73$

Рис. 20: Ложно-положительные распознавания (**FP**) модели с различным показателем уверенности (**conf**).

Использование уточняющей модели подразумевает использование дополнительных данных. Применительно к алгоритмам машинного обучения увеличение объема датасета (при сохранении его качественных характеристик, например распределения примеров каждого класса) обычно повышает

качество модели, поэтому решение расширить датасет интуитивно кажется способствующим улучшению. Однако в данном случае предлагается расширение данных для задачи классификации, а не детектирования. У этого подхода есть ряд преимуществ:

- Дополнительные данные проще собрать;
- Сигареты на дополнительных данных более разнообразны;
- Изображения не требуют выделения объектов на них, для них необходима лишь метка класса;
- При обучении бинарного классификатора используются также данные класса **не сигарета**. Сюда могут входить ручки, белые полоски и прочие объекты, похожие на сигареты, которые также проще собрать.

Также не стоит забывать о том, что слабая модель будет не только пропускать истинно отрицательные классификации, но и браковать истинно положительные предсказания.

3.3.1 Обучение MobileNetV2

Все изображения приведены к размеру 128×128. Такое преобразование не сильно нарушает пропорции согласно анализу датасета в 1.2. В качестве функции ошибки выбрана перекрестная энтропия [21], оптимизатор — Adam [24], шаг обучения $lr = 0.001$. Обучение завершилось на 12 эпохах, после чего наблюдается переобучение: ошибка на валидационном множестве начинает расти. Также модель была дополнительно дообучена на примерах из **обучающего** набора для детектирования на ограничивающих прямоугольниках площадью более 4000 пикселов ($\approx 89^2$), которых всего 176 (и 50 в тестовом). Результаты на тестовом (для классификации) множестве приведены в Таблице 4.

3.4 Объединение YOLO и MobileNetV2.

Реализация объединения моделей детектирования и классификации имеет ряд особенностей:

- На большинстве изображений ограничивающая рамка не идеально охватывает сигарету — часть пикселей теряется. На это частично влияет изменение отношений сторон исходного изображения, несмотря на то, что к ограничивающей рамке применяется обратное преобразование. Потеря пикселей отчасти ведет к уменьшению информации, необходимой для классификации — в рамку могут явно не попасть фильтр или тлеющая часть сигареты, если она есть. Поэтому рамки всех изображений, поданных на вход классификатору на второй стадии увеличены на 10 пикселей с каждой стороны (слева, справа, сверху и снизу).

- Классификация объектов с маленькой площадью (примерно $\leq 2000px$), обнаруженных YOLO лишь благодаря контексту, — задача трудная не только для нейросети, но и для человека. Классификация таких объектов доверяется модели YOLO. Оставшиеся же рамки дополнительно классифицируются.

- Обученный классификатор имеет фиксированный вход: квадратное изображение 128×128. Предсказанные YOLO рамки наоборот — произвольной формы. Приведение всех рамок к фиксированному размеру (и соотношению сторон) вынуждает стягивать или растягивать изображение, но в данной задаче это недопустимо, так как приводит нарушению пропорций сигарет внутри рамок. Поэтому в этой работе изображение сначала масштабируется так, чтобы длина бóльшей из сторон рамки была равна 128 пикселям, а затем оставшаяся часть дополняется черными пикселями.

Результаты для задачи классификации		
	Обучение только на наборе для классификации	Дообучение на тренировочной части набора для детектирования
Precision	0.82	0.89
Recall	0.74	0.80

Таблица 4: Результаты моделей на тестовом множестве задачи классификации, обученных только

3.4.1 Принцип работы

Принцип работы глобальной модели не сильно отличается от «голой» YOLO:

- Исходное изображение подается на вход сети YOLO;
- Отедьно задается порог площади предсказанных прямоугольников $S_{treshold}$;
- Выход сети YOLO — множество координат всех ограничивающих прямоугольников, а также класс (в данной задаче класс единственный) и степень уверенности ($confidence\_score$) для каждого — разделяется на два подмножества. Первое остается без изменений и становится частью выхода глобальной модели. В него попадают объекты с площадью $\leq S_{treshold}$. Второе множество подается на вход модели классификатору;
- Модель классификатор преобразует все ограничивающие рамки к фиксированному размеру, как было сказано выше, и совершает предсказания. Выходом классификатора для отдельно взятого изображения является пара значений (p_1, p_2) обозначающих вероятность присутствия на изображении сигареты и не сигареты соответственно. Но в рамках глобальной модели эти значения не учитываются, берется лишь метка класса на основе того, какое значение больше:

$$ans_{predicted} = \begin{cases} 1, & \text{если } p_1 \geq p2 \\ 0, & \text{если } p_1 < p_2 \end{cases}, \text{где}$$

$$p_1 + p_2 = 1,$$
$$p_1, p_2 \in [0, 1];$$

- Таким образом, из второго подмножества остаются лишь те ограничивающие рамки, для которых классификатор выдал значение 1. Оба

подмножества объединяются в одно — это и есть выход глобальной модели для отдельно взятого изображения.

К минусам можно отнести увеличение времени работы общей модели, однако, как показывает анализ набора данных — ограничивающие прямоугольники имеют не высокое разрешение, а следовательно не сильно влияют на время выполнения.

3.4.2 Выводы

Построенная в результате работы глобальная модель способна работать полностью самостоятельно, но для оптимизации времени проведение экспериментов упрощено: сперва модель детектирования (одна из лучших по метрике mAP@0.5) совершает предсказания на тестовом множестве. Полученные предсказанные прямоугольники затем используются для следующего блока экспериментов, в котором рассматривается зависимость основных метрик от порога площади прямоугольника $S_{treshold}$ (Таблица 5). Результаты показывают, что добавление классификатора повышает результаты на тестовом множестве, однако это требует четко выставленного порога площади: чем больше места занимает сигарета на изображении, тем проще классификатору ее распознать.

При уменьшении порога часть сигарет, изначально правильно классифицированных, теряется. А с увеличением порога уменьшается количество

Площадь порога для классификации	Precision глобальной модели	Recall глобальной модели	mAP@0.5 глобальной модели
2000	0.75	0.69	0.74
4000	0.85	0.79	0.85
8000	0.92	0.86	0.91
12000	0.94	0.85	0.93
18000	0.9	0.85	0.87
24000	0.89	0.82	0.87
max	0.89	0.82	0.87

Таблица 5: Зависимость результатов глобальной модели от порога площади $S_{treshold}$ прямоугольников, подающихся на вход классификатору

прямоугольников, ему соответствующим. Поэтому при стремлении порога площади к максимальному возможному объему, все метрики стремятся значениям, соответсвующим одиночной модели YOLO.

Таким образом, порогом площади $S_{treshold}$ для данной задачи может быть выбрано значение из диапазона $[8000, 12000]$. Результаты показывают, что при таком выборе значения порога, метрика mAP@0.5 принимает наибольшее значение.

Заключение

В выпускной квалификационной работе:

1. Проведен поиск и разметка изображений, содержащих сигареты, а также из них составлен набор для обучения, валидации и проверки решения;
2. Выбрана и обучена модель детектирования на собранных данных;
3. Протестированы современные подходы искусственного дополнения исходного набора данных;
4. Произведены сравнения несколько подходов к определению функции потерь при обучении модели детектирования;
5. В модель детектирования встроен модуль классификации для уменьшения ошибок второго рода.

Список литературы

[1] J. Yu, Y. Jiang, Z. Wang, Z. Cao, T. Huang «UnitBox: An Advanced Object Detection Network». *arXiv:1608.01471v1*, 2016.

[2] S. Kosub «A note on the triangle inequality for the jaccard distance». *arXiv preprint arXiv:1612.02696*, 2016.

[3] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, S. Savarese «Generalized Intersection over Union: A Metric and A Loss for Bounding Box Regression ». *arXiv:1902.09630*, 2019.

[4] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, D. Ren «Distance-IoU Loss: Faster and Better Learning for Bounding Box Regression». *The AAAI Conference on Artificial Intelligence*, 2020.

[5] Z. Zheng, P. Wang, D. Ren, W. Liu, R. Ye, Q. Hu, W. Zuo «Enhancing Geometric Factors in Model Learning and Inference for Object Detection and Instance Segmentation». *IEEE Transactions on Cybernetics, doi: 10.1109/TCYB.2021.3095305*, 2021.

[6] Ali Khan «Dataset Containing Smoking and Not-Smoking Images (Smoker vs Non-Smoker)». *https://data.mendeley.com/datasets/7b52hhzs3r/1*, 2020.

[7] M. Everingham, L. V. Gool, Ch. K. I. Williams, J. Winn, A. Zisserman, «The PASCAL Visual Object Classes (VOC) Challenge». *International Journal of Computer Vision*, 2010.

[8] S. Ioffe, C. Szegedy «Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift», *eprint arXiv:1502.03167* 2015.

[9] K. He, X. Zhang, S. Ren, J. Sun «Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition», *eprint arXiv:1406.4729*, 2015.

[10] B. Alexey, W. Chien-Yao, L. Hong-Yuan Mark «YOLOv4: Optimal Speed and Accuracy of Object Detection», *eprint arXiv:2004.10934*, 2019.

[11] Shorten, C., Khoshgoftaar, T.M. «A survey on Image Data Augmentation for Deep Learning» *J Big Data https://doi.org/10.1186/s40537-019-0197-0*, 2019.

[12] Jia D, Wei D, Richard S, Li-Jia L, Kai L, Li F-F. «ImageNet: a large-scale hierarchical image database», *In: CVPR09*, 2009.

[13] Karl W, Taghi MK, DingDing W. «A survey of transfer learning», *J Big Data*, 2016.

[14] Shao L. «Transfer learning for visual categorization: a survey», *IEEE Trans Neural Netw Learn Syst*, 2015.

[15] https://pytorch.org/vision/stable/transforms.html

[16] `https://cocodataset.org/#home`

[17] Sungrae Kim, Hyun Kim «Zero-Centered Fixed-Point Quantization With Iterative Retraining for Deep Convolutional Neural Network-Based Object Detectors», *IEEE Access*, 2021.

[18] G. Ghiasi, Y. Cui, A. Srinivas, R. Qian, T.-Y. Lin, E. D. Cubuk, Q. V. Le, B. Zoph «Simple Copy-Paste is a Strong Data Augmentation Method for Instance Segmentation», *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

[19] `https://keras.io/examples/vision/mixup/`

[20] `https://ultralytics.com/`

[21] `https://pytorch.org/docs/stable/generated/torch.nn.BCELoss.html`

[22] A. Howard, M. Sandler, L. Chen, Y. Zhu, A. Zhmoginov «MobileNetV2: Inverted Residuals and Linear Bottlenecks», *Proceedings of the IEEE conference on computer vision and pattern recognition* 2018.

[23] `https://www.sciencedaily.com/releases/2012/09/120927123646.htm`

[24] https://pytorch.org/docs/stable/generated/torch.optim.Adam.html

[25] Y. Zhong, J. Wang, J. Peng, L. Zhang «Anchor Box Optimization for Object Detection» *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020.