Обзор нейронных сетей для компьютерного зрения

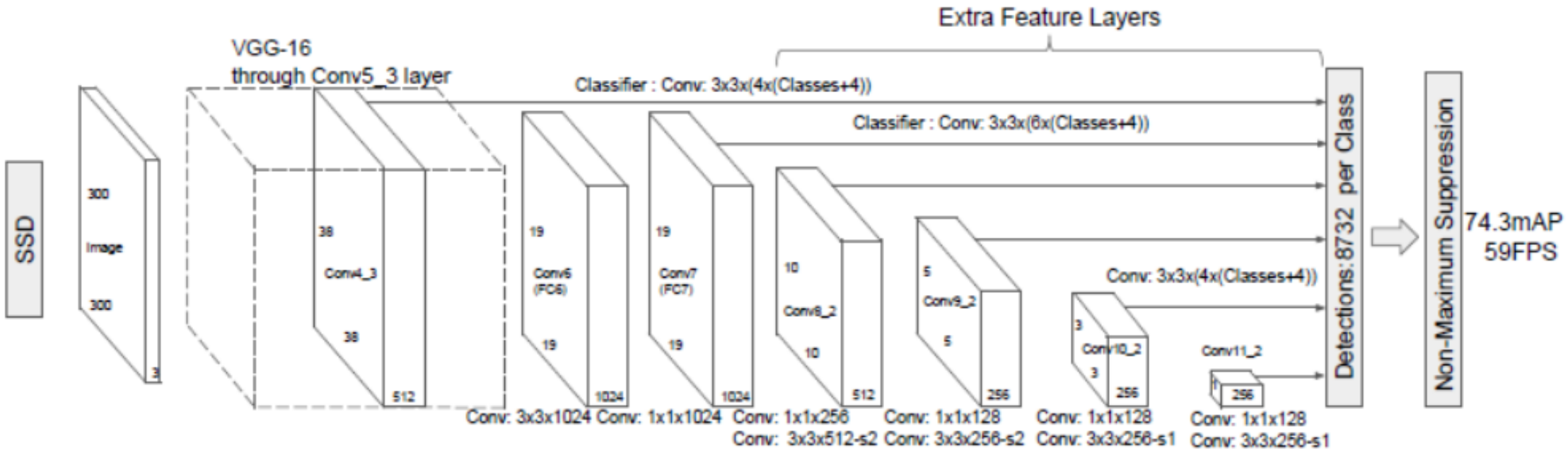
Детекция объектов: От Двухстадийных к Одностадийным и Современное Состояние

Двухстадийные детекторы объектов (такие как R-CNN, Fast R-CNN, Faster R-CNN, Mask R-CNN) сначала выделяют области-кандидаты, а затем уточняют их классификацию и границы, обеспечивая высокую точность, но меньшую скорость.

Одностадийные детекторы (такие как SSD, RetinaNet, EfficientDet, YOLO) выполняют задачу за один проход, одновременно предсказывая классы и границы объектов, что делает их быстрыми, но менее точными для мелких или плотно расположенных объектов.

Выбор подхода зависит от приоритета: точность или скорость.

SSD (Single Shot MultiBox Detector) — 2016



Сочетает в себе скорость и точность, обеспечивая одноступенчатый подход к детекции объектов

SSD (Single Shot MultiBox Detector)

- Входное изображение обрабатывается через предобученную свёрточную нейронную сеть (например, VGG16) для извлечения признаков.
- SSD создаёт анкерные рамки различных размеров и соотношений сторон по всей сетке, покрывающей изображение.
- На каждом уровне сети осуществляется предсказание как для классов объектов, так и для координат ограничительных рамок.
  Это происходит через полносвязные слои, которые используются для предсказания классов и корректировки координат.
- Используется функция потерь, учитывающая как ошибки классификации, так и ошибки локализации ограничительных рамок, что позволяет оптимизировать оба процесса одновременно.

Основные особенности SSD

Одноэтапная детекция:
SSD выполняет предсказания ограничительных рамок и классов объектов за одно обращение к сети, что делает его более быстрым по сравнению с двухступенчатыми методами, такими как R-CNN и Faster R-CNN.

Многоячеечная сеть:
SSD использует несколько уровней сети для предсказания объектов, что позволяет детектировать объекты различного масштаба и аспектов.
Модель делает предсказания на разных уровнях иерархии, начиная с глубоких слоёв, что помогает обрабатывать как крупные, так и мелкие объекты.

Анкерные ограничительные рамки:
SSD использует анкерные рамки (anchors) различных соотношений сторон и размеров для генерации ограничительных рамок. Каждая ячейка сетки отвечает за предсказание нескольких ограничительных рамок.

RetinaNet (2017)

Архитектура для детекции объектов **решает проблему несбалансированности классов** при использовании **одноступенчатых** подходов (например, детекция объектов, где объектов мало, а фона много).

Основной инновацией RetinaNet является внедрение **новой функции потерь**, называемой **Focal Loss**, которая помогает улучшить точность детекции объектов, особенно при наличии большого количества фонов и малочисленных объектов.

Обычная кросс-энтропийная потеря

Обычная кросс-энтропийная потеря (Cross-Entropy) для бинарной классификации:

CE(p, y) = -log(p) если y=1,
-log(1-p) если y=0,
где p — предсказанная вероятность для правильного класса.

Проблема: при дисбалансе классов простые примеры (например, фон) доминируют в общей ошибке.

Модификации в Focal Loss

1. **Балансирующий параметр α (alpha)** Добавляет вес для редкого класса:

$$CE\_\alpha(p, y) = \begin{cases} -\alpha \cdot \log(p) & \text{если } y=1 \\ -(1-\alpha) \cdot \log(1-p) & \text{если } y=0 \end{cases}$$

2. Фокусирующий параметр γ (gamma) Ключевая инновация — модуляция по легкости примера:

$$FL(p, y) = \begin{cases} -(1-p)^{\gamma} \cdot \log(p) & \text{если } y=1 \\ -p^{\gamma} \cdot \log(1-p) & \text{если } y=0 \end{cases}$$

3. Комбинированная формула

Для класса с меткой y=1: $FL(p) = -\alpha \cdot (1-p)^{\gamma} \cdot \log(p)$

Для класса с меткой y=0: $FL(p) = -(1-\alpha) \cdot p^{\gamma} \cdot \log(1-p)$

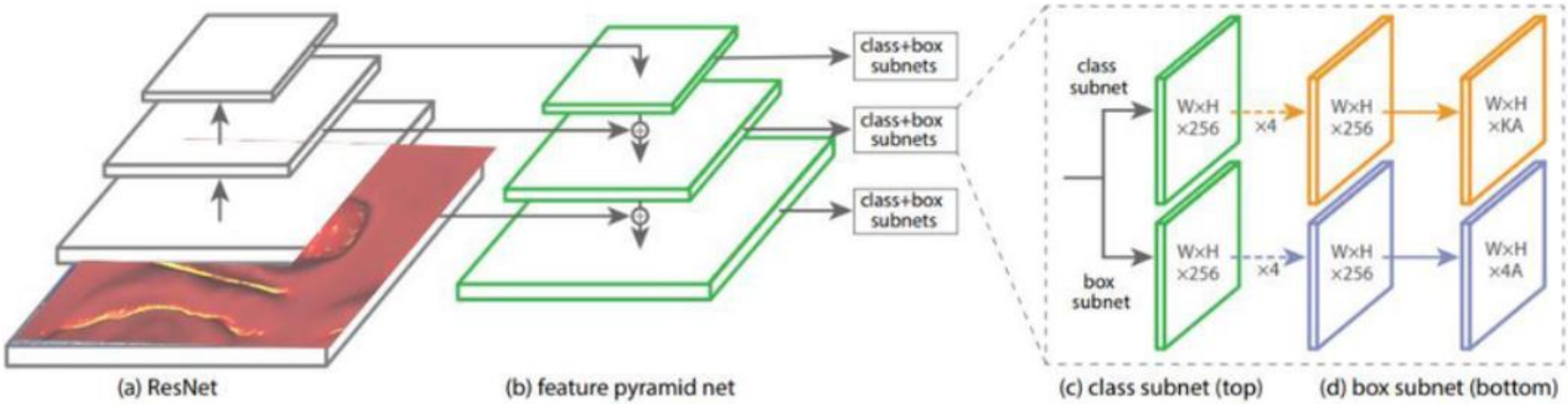
где:

p — предсказанная вероятность правильного класса (от 0 до 1)

α (альфа) — балансирующий параметр (обычно 0.25)

γ (гамма) — фокусирующий параметр (обычно 2)

RetinaNet (2017)



Входное изображение проходит через предобученную сверточную нейронную сеть (например, ResNet), чтобы извлечь высокоуровневые признаки.
Анкерные рамки генерируются на основе сетки, покрывающей изображение, с использованием различных соотношений сторон и размеров.
На выходе от сети происходит классификация (определение классов объектов) и регрессия (коррекция координат ограничительных рамок) для каждой анкерной рамки.

Основные особенности RetinaNet

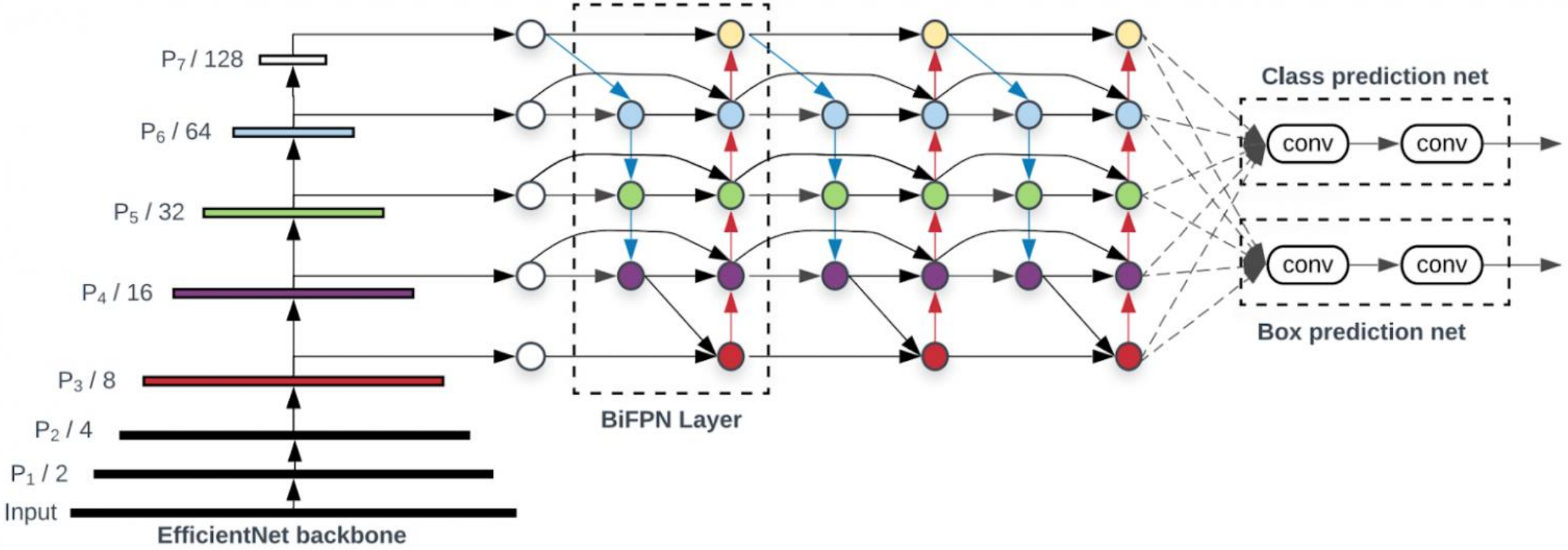
Одноэтапная детекция: RetinaNet выполняет детекцию объектов за одно обращение к сети, что позволяет ему быть быстрым, при этом обеспечивая высокую точность.

Focal Loss разработан для борьбы с **проблемой несбалансированных** классов, когда фоновые примеры значительно преобладают над примерами объектов. Эта функция потерь снижает вклад легко классифицируемых объектов, позволяя модели **сосредоточиться на трудных** для классификации примерах.

Feature Pyramid Network (FPN) для работы с объектами разных размеров. FPN позволяет извлекать признаки на нескольких уровнях разрешения, что помогает лучше обрабатывать мелкие и крупные объекты.

Использование анкерных рамок: подобно SSD, RetinaNet использует анкерные рамки различных размеров и соотношений сторон для генерации ограничительных рамок.

EfficientDet (2020)



Разработана для достижения оптимального баланса между производительностью и точностью

EfficientDet

Основана на концепциях, заложенных в EfficientNet, и использует комплексный подход для оптимизации всех компонентов модели.

Входное изображение обрабатывается через эффективную базовую сеть (EfficientNet), чтобы извлечь признаки.

Используется BiFPN (Bidirectional Feature Pyramid Network) для объединения признаков с различных уровней иерархии, что позволяет улучшить детекцию объектов разных размеров.

EfficientDet генерирует анкерные рамки на основе слоев BiFPN, что улучшает локализацию объектов.

На выходе сети осуществляется как классификация объектов, так и регрессия для коррекции ограничительных рамок.

BiFPN (Bidirectional Feature Pyramid Network)

BiFPN — это улучшение стандартной Feature Pyramid Network (FPN),
позволяет объединять признаки с разных уровней с учётом их относительной важности.

Использует обучаемые веса для слияния признаков, что помогает лучше учитывать контекст между разными уровнями разрешения.

Основные особенности EfficientDet

Эффективность и производительность:
EfficientDet была разработана с акцентом на экономию вычислительных ресурсов и повышение производительности, что делает её подходящей для работы на **устройствах с ограниченными ресурсами**, таких как мобильные телефоны и встраиваемые системы.

Базовая архитектура:
EfficientDet использует архитектуру EfficientNet в качестве базового стержня, которая оптимизирует свёрточные нейронные сети с помощью Compound Scaling.
Этот подход позволяет масштабировать ширину, глубину и разрешение модели одновременно, улучшая её эффективность.
Это позволяет создавать различные версии EfficientDet (D0, D1,..., D7), адаптированные для устройств с разной вычислительной мощностью.

Основные особенности EfficientDet

Сетевые слои:
EfficientDet включает в себя специализированные блоки, такие как BiFPN, которые обеспечивают эффективное объединение признаков с разных уровней, улучшая возможность детекции объектов на различных масштабах.

Анкерные рамки:
подобно другим архитектурам, EfficientDet использует анкерные рамки, но с улучшенным методом выбора размеров и аспектов рамок, что повышает точность обнаружения.

Многоуровневое предсказание:
EfficientDet производит предсказания на нескольких уровнях, что позволяет модели адаптироваться к различным масштабам объектов и улучшает общую точность.

YOLO: эволюция и революция в детекции объектов

YOLO (You Only Look Once) — это архитектура нейронной сети для детекции объектов, впервые представленная в 2016 году.

Она произвела революцию в области компьютерного зрения, предложив уникальный подход к задачам детекции объектов.

YOLO объединила предсказание классов и локализацию объектов в единый процесс, что позволило добиться впечатляющего сочетания высокой скорости и точности.

Это самая популярная на сегодняшний день архитектура благодаря её простоте в использовании и соотношению скорости и качества.

YOLO — огромное количество версий

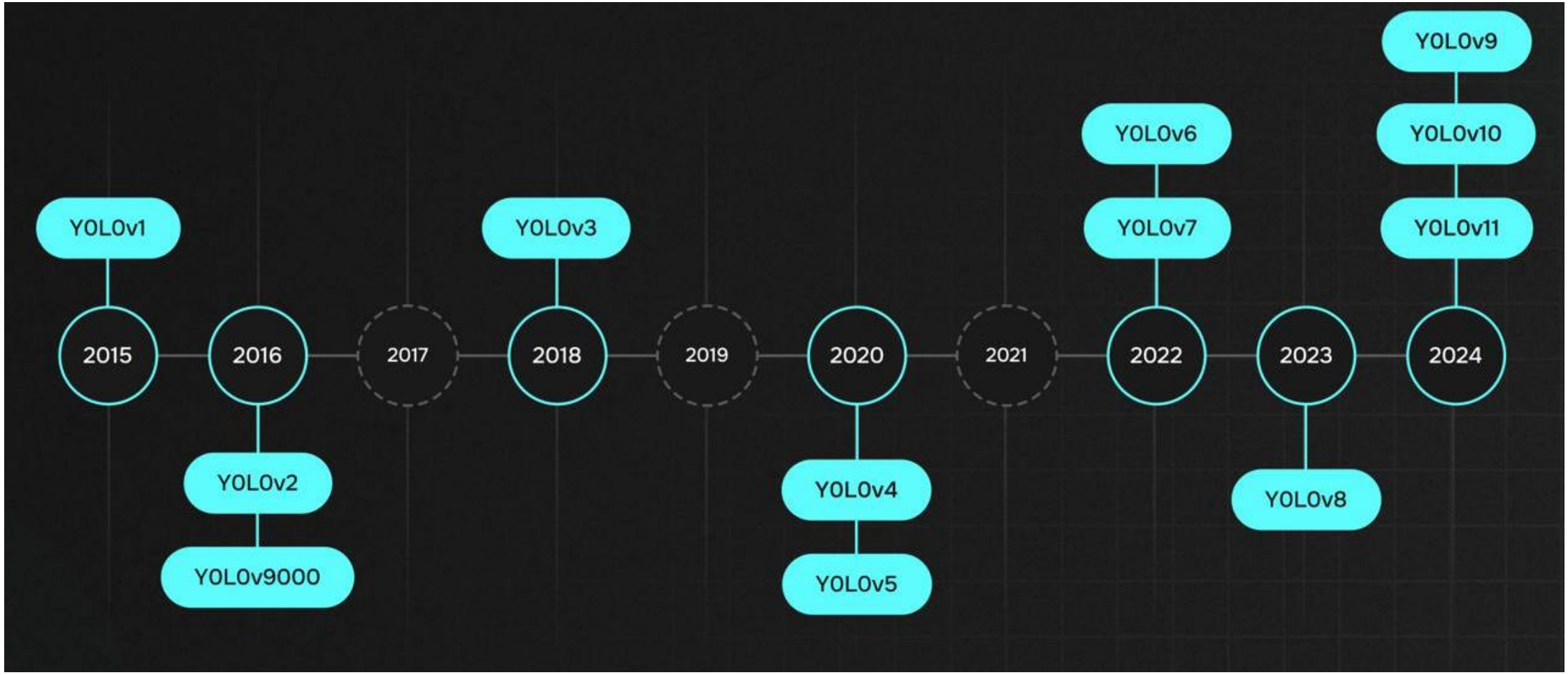
Первая из них вышла в 2016 году и решала только задачу детекции объектов на изображении

Одиннадцатая представляет из себя целую фундаментальную модель, которую можно использовать для:

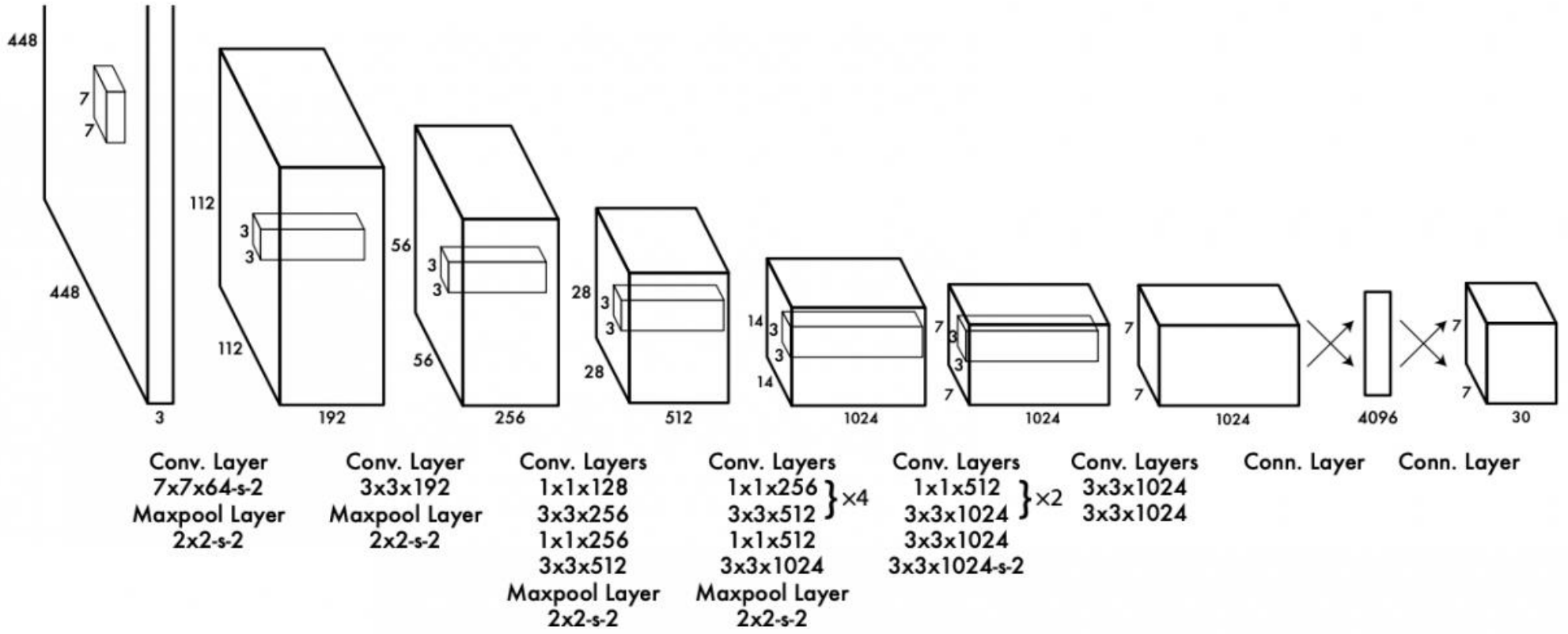
- классификации,
- трекинга объектов на видео,
- задач pose estimation
- и тд.

Всё это — в реальном времени

Семейство или экосистема



YOLOv1



Это немного видоизмененный GoogLeNet: в оригинальной CNN 22 сверточных слоя, но создатели YOLO добавили ещё два + полносвязные слои в конце.

YOLOv1

На вход этой сети подаётся изображение 448x448 (если подать изображение другого размера, то оно просто обрежется и/или отправится на съедение функции resize).

Изображение предварительно разделено на одинаковые квадратные ячейки размера 64x64 таким образом, что получается таблица 7x7 (нет, 7x7 – не волшебная константа, вы можете использовать и другой размер ячеек, но как это повлияет на сеть – никто особо не исследовал).

Ячейки нужны для того, чтобы все операции далее происходили как бы на «клеточном» уровне.

В этом заключено главное новшество YOLO: её создатели смогли сформулировать и решать задачу детекции как задачу регрессии.

YOLOv1

Выходной тензор сети имеет размер 7x7x30.
То есть для каждой из 7x7 ячеек изображения модель предсказывает вектор из 30 чисел.

Внутри этого вектора и скрывается описание б-боксов и меток классов.

Если точнее, то первые 10 значений отвечают за координаты двух б-боксов-кандидатов: координаты центра + ширина + высота + confidence score, т.е. уверенность модели в том, что внутри б-бокса находится центр объекта.

Оставшиеся 20 значений вектора ответственны за метки классов, то есть оценку вероятности того, что объект определённого класса присутствует в ячейке.

Почему 20? Потому что столько было классов в оригинальном датасете.

Основные Принципы YOLO

1. **Одноэтапная детекция**.
YOLO решает задачу детекции объектов за одно обращение к сети.

Это одностадийная архитектура, которая резко контрастирует с традиционными двухстадийными методами, такими как R-CNN или Faster R-CNN, где сначала генерируются регионы-кандидаты, а затем происходит их классификация.

Подход YOLO значительно ускоряет обработку изображения, что делает его идеальным для задач, требующих реального времени.

2. **Сетка фиксированного размера.**
Изображение делится на сетку, где каждая ячейка отвечает за предсказание ограничительных рамок (bounding boxes) и вероятностей классов объектов, находящихся в пределах этой ячейки.

Это упрощает обработку сложных сцен с несколькими объектами.

Основные Принципы YOLO

3. Предсказание ограничительных рамок (bounding boxes).
Для каждой ячейки YOLO предсказывает фиксированное количество bounding boxes. Каждая рамка описывается координатами центра, шириной, высотой и вероятностью содержания объекта.

4. Классификация объектов.
Помимо предсказания рамок, YOLO также определяет вероятность принадлежности объекта к определенному классу. Это позволяет идентифицировать, какой именно объект находится в пределах каждой рамки.

Функция потерь

regression loss — ошибка в предсказании координат центра (x,y), высоты (h) и ширины (w) б-боксов.
Индекс i отвечает за объекты, индекс j – за боксы.

Regression loss

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

confidence loss отвечает за оценки уверенности модели в том, что внутри б-бокса находится центр объекта.
При этом это не просто вероятность.
Это предсказанное значение функции IoU, то есть Intersection over Union.

Confidence loss

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

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

classification loss — ошибки на 20 метках классов.
При этом индикаторная функция здесь снова гарантирует, что будут учтены только те ячейки, в которых действительно есть объект.

Classification loss

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

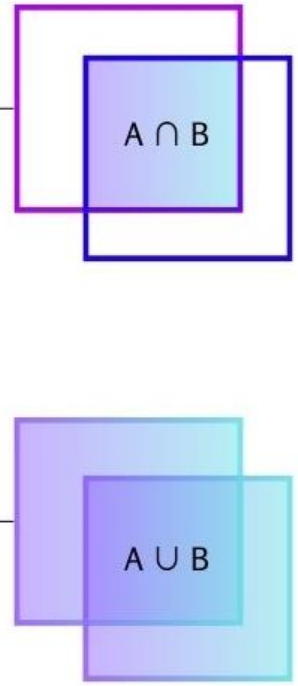
Функция IoU (Intersection over Union)

В терминах б-боксов IoU – это перекрытие между прогнозируемой рамкой и истинным прямоугольником.

Если модель предсказывает высокое IoU, то она верит в то, что перекрытие будет большим, т.е. сильнее уверена в существовании определенного б-бокса.

И наоборот, если предсказывает низкое, то IoU уверена, что в ячейке объекта вообще нет.

$$IoU = \frac{\text{Area of Intersection}}{\text{Area of Union}}$$



одна из определяющих метрик компьютерного зрения

Преимущества YOLO

Высокая скорость:
Благодаря тому, что вся обработка изображения происходит за один проход через сеть, YOLO способна работать с видео или потоковыми изображениями в режиме реального времени. Это делает ее идеальной для приложений, где критически важна скорость, таких как: видеонаблюдение, автономные транспортные средства, робототехника.

Целостный подход:
YOLO обрабатывает всё изображение сразу, что позволяет учитывать глобальный контекст. Это помогает минимизировать ложные срабатывания на фоне.

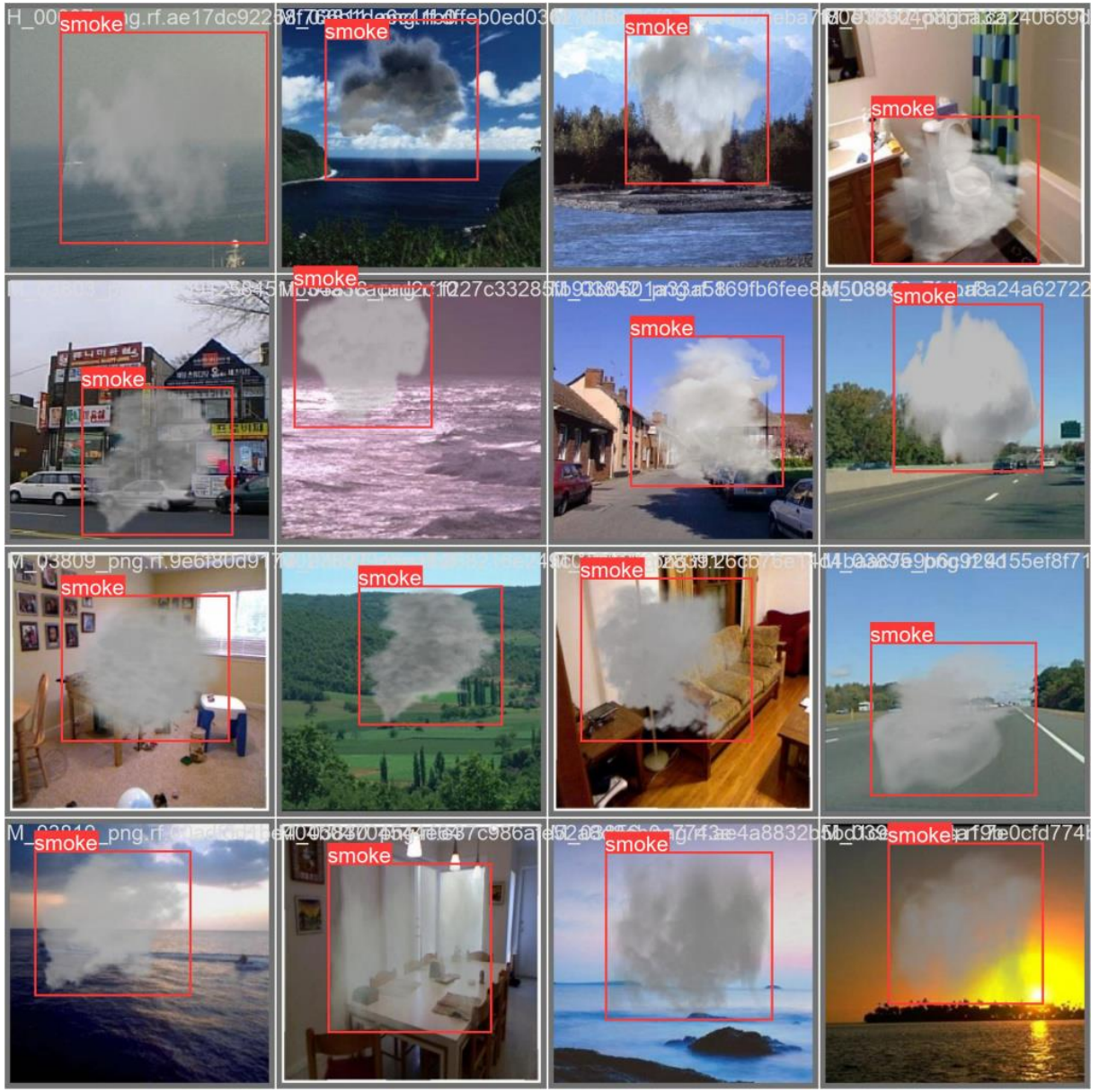
Простота использования:
благодаря отличной документации и готовым фреймворкам, начать работать с YOLO просто. С помощью встроенных инструментов и Google Colab можно обучить модель и получить результаты за несколько часов.

Какие данные нужны?

Изображения или видео + их разметка, где для каждого объекта указываются координаты ограничивающего прямоугольника и его класс.

Для старта часто используют открытые наборы данных.
Наиболее известные среди них:

- COCO, содержащий сотни тысяч изображений с десятками категорий объектов
- VOC, который стал классикой для обучения и тестирования алгоритмов.



Инференс

Предположим, мы обучили модель, и она даёт нам на выходе некоторый тензор.
Как собрать из него окончательный ответ, который должен содержать отфильтрованные б-боксы с метками классов?

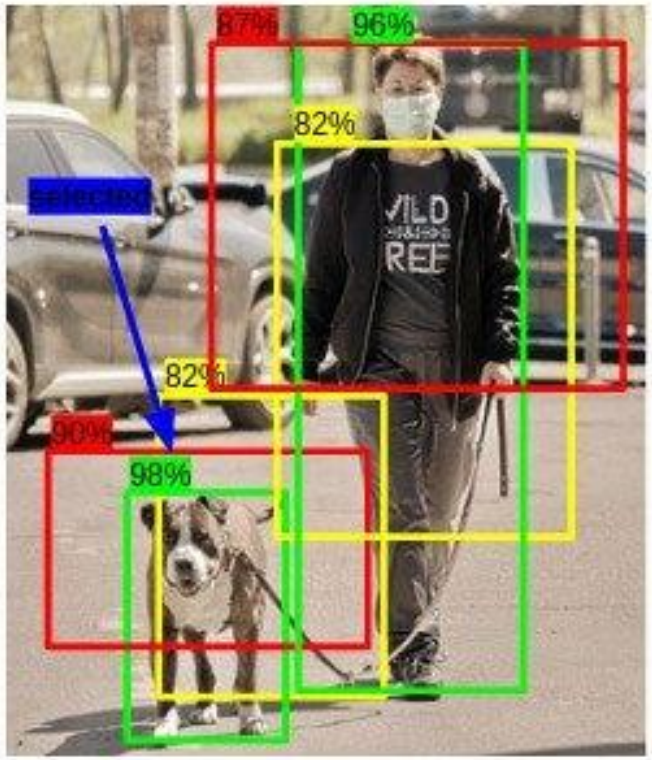
1. Нужно привязать метки классов к определённым б-боксам: взять confidence score каждого б-бокса и перемножить его с каждой вероятностью класса. В итоге набор прямоугольников, для каждого из которых определены лейблы классов, координаты центра, ширина, высота и IoU aka confidence score.

2. Нужно решить, какие из б-боксов мы оставим, а какие удалим: удалим все рамки, для которых IoU < 0.5

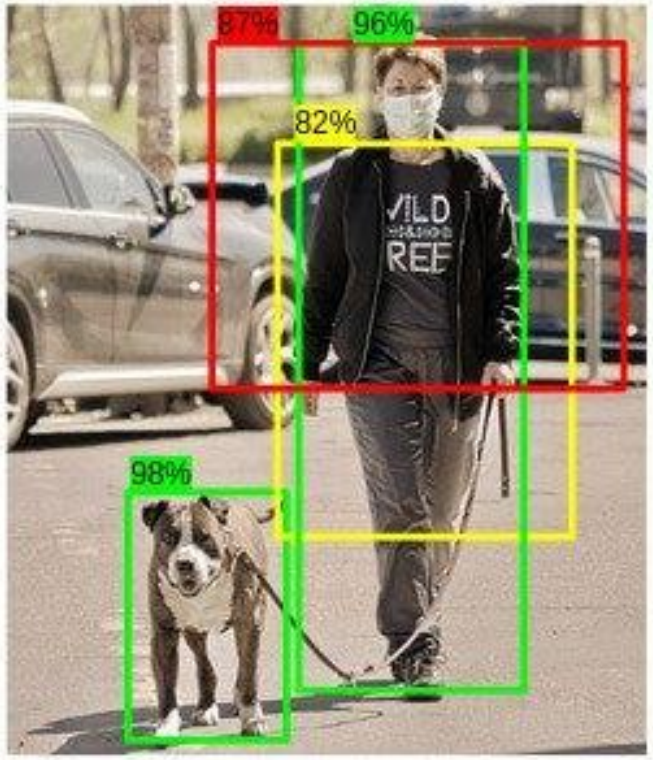
Инференс

3. Объект может не помещаться целиком только в одну из клеток. Тогда два, и даже три б-бокса из разных ячеек могут быть на самом деле единым б-боксом для одного и того же предмета.

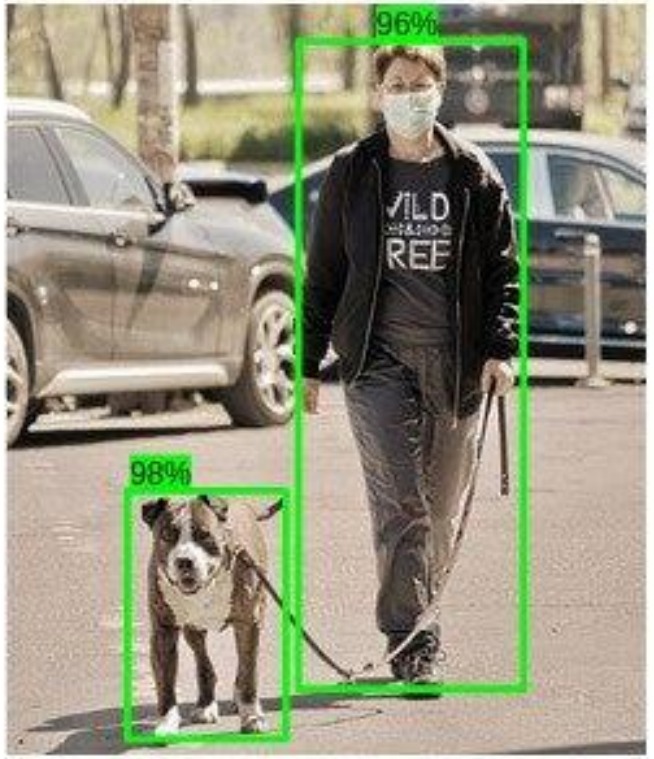
   На помощь приходит алгоритм **Non-maximum Suppression**.
   - Сначала мы берём список б-боксов, которые остались после пункта 2 и сортируем его по убыванию IoU, так, чтобы в начале были те б-боксы, в которых, как думает модель, расположены центры объектов.
   - Затем будем по очереди брать самых вероятных кандидатов и находить все б-боксы, которые пересекаются с ними настолько, что IoU этого пересечения больше некоторого порога.
   - Все такие б-боксы мы будем удалять и вычеркивать из исходного списка, и так до тех пор, пока список не опустеет, а у нас не появится набор финальных отфильтрованных б-боксов.



Step 1: Selecting Bounding box with highest score



Step 3: Delete Bounding box with high overlap



Step 5: Final Output

Изменения в архитектуре и не только

1. Удалили полносвязные детекционные слои в конце. На их место пришли свертки. Кроме того, из архитектуры убрали dropout. Вместо этого добавили batch normalization, который на тот момент показал себя хорошим инструментом для повышения сходимости и скорости обучения модели.

2. Саму базовую архитектуру тоже поменяли с GoogLeNet на Darknet-19. Эта сеть состоит из меньшего количества слоев: 19 сверточных слоев против 22 у GoogLeNet. За счет такой подмены модель не просела на задачах в реальном времени.

3. Если в предыдущей версии дообучение первых 20 классификационных слоев происходило на изображениях размером 224x224 из ImageNet, то вторая версия училась на изображениях более высокого качества: 448x448.

4. При этом во время обучения архитектуры целиком разрешение входной картинки снизили до 416x416. В этом случае изображение можно разделить на нечётное количество ячеек: тогда в середине изображения, где вероятность появления объекта больше всего, будет одна ячейка, а не четыре.

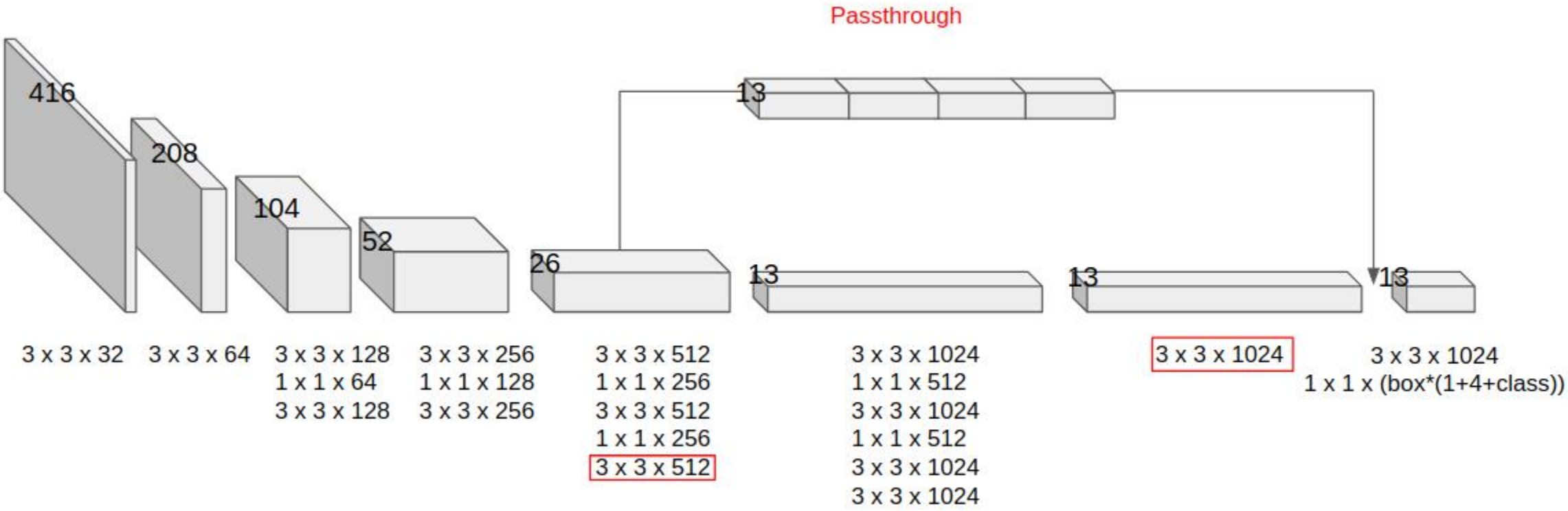
Изменения в архитектуре и не только

5. Само количество ячеек тоже увеличили. Если в YOLO1 сетке 7x7, то теперь сетка 13x13: удалили один слой пуллинга, благодаря чему размер выходного тензора получился 13x13x125.

6. Почему 125? В YOLO2 предсказываем метки классов уже не для всех боксов ячейки одновременно: теперь для каждого б-бокса вектор таких вероятностей свой.
   К тому же, теперь б-боксов на одну ячейку стало больше: раньше их было два, а теперь число увеличилось до пяти.
   Итого 5 б-боксов, для каждого координаты центра, ширина, высота, IoU и 20 меток классов: всего 125.

7. Появились слои skip connection, которые предотвращают переобучение модели и делают ее более стабильной.

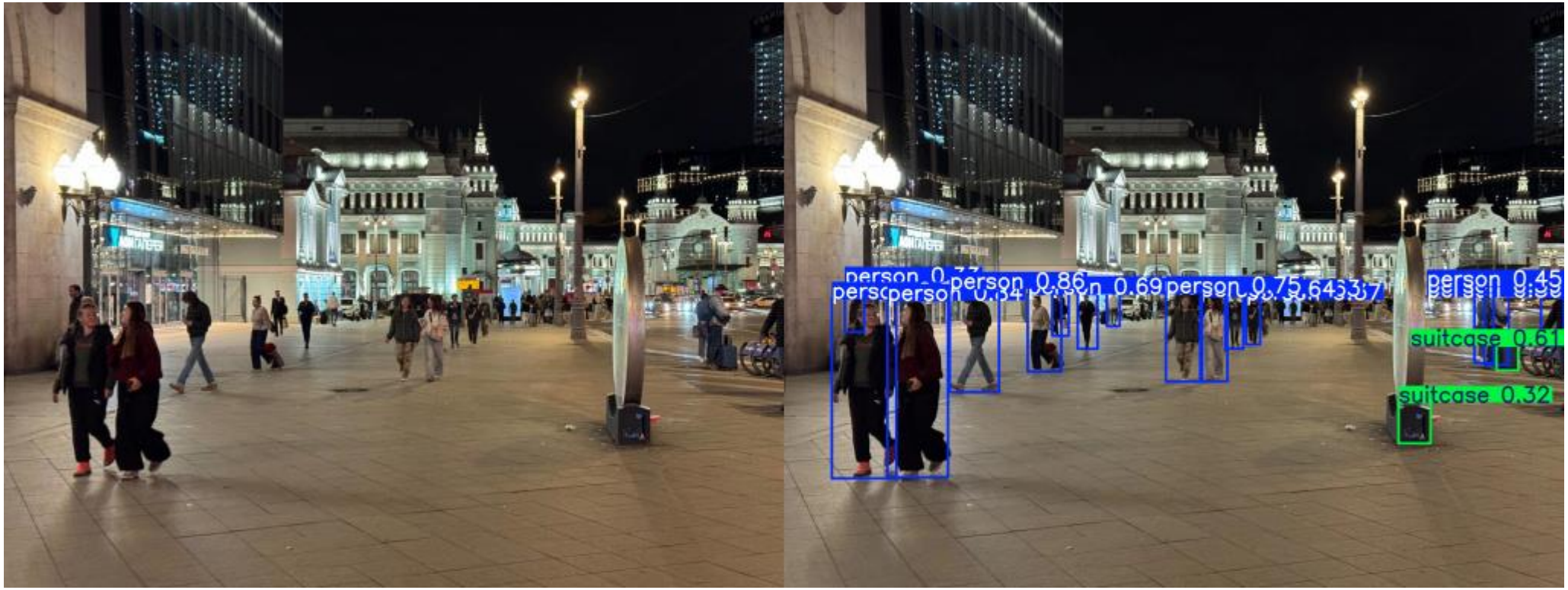
YOLOv2



В YOLO 2 изменилась сама идея подхода к детекции объектов. Вместо традиционных б-боксов исследователи подчерпнули из Faster R-CNN задумку предсказывать anchor boxes (или якори).

Версия YOLO	Статус в коммерческих проектах	Комментарий
YOLOv1-v4	❌ Устарели	Исторически важные версии, сегодня почти не применяются
YOLOv5	✅ Используется в продакшене	До сих пор массовый стандарт: много документации, обучающих материалов и готовых весов. Лёгкий старт для бизнеса
YOLOv6	⚠️ Ограничено используется	Разработка Meituan для внутренних индустриальных задач. Популярен меньше, чем v5-v8
YOLOv7	⚠️ Ограничено используется	Встречается в старых системах, но постепенно уступает место YOLOv8 и YOLOv11
YOLOv8	✅ Используется в продакшене	Универсальный инструмент: поддерживает детекцию, сегментацию, классификацию и позу. Оптимальный выбор для большинства задач
YOLOv9	⚠️ R&D / эксперименты	Интересен исследователям (PGI, GELAN), но не стал массовым стандартом
YOLOv10	⚠️ R&D / эксперименты	Сильный акцент на скорость и эффективность (NMS-free), но ограниченная экосистема
YOLOv11	✅ Новый флагман	Поддерживается Ultralytics, активно продвигается. Высокая точность и удобный туллинг
YOLOv12	⭐ Перспективно	Новый релиз (2025). Пока мало внедрений, но стоит следить: attention-архитектура и высокий потенциал

Недостатки



YOLO на лету определяет количество и тип объектов, всего за один проход.
Расплата за это — сниженная точность распознавания.
Конкретно в данном случае YOLO верно определила количество людей в кадре, но ошибочно приняла за чемодан скамейку и скульптуру.

Перспективы развития YOLO

Появились проекты типа:

- YOLO-NAS, где архитектура ищется автоматически с помощью нейроэволюции

- YOLO-World, поддерживающий open-vocabulary detection
  Это означает, что модель способна находить объекты, на которых её не обучали явно, используя текстовые подсказки и знания из языковых моделей

- YOLO-Nano (Edge-устройства и мобильный AI)
  перенос моделей на маломощные устройства.
  Сюда относятся дроны, камеры видеонаблюдения, умные колонки и даже смартфоны.
  Здесь критична энергоэффективность и минимальные задержки.

Основные виды сегментации изображений

1. **Бинарная сегментация (Binary Segmentation)**
   Это самый простой вид сегментации, в котором изображение делится на два класса — фон и объект.

2. **Мультиклассовая сегментация (Multi-Class Segmentation)**
   В мультиклассовой сегментации каждому пикселю изображения присваивается один из нескольких классов, которые являются взаимно исключающими.

3. **Семантическая сегментация (Semantic Segmentation)**
   Процесс классификации каждого пикселя изображения в один из предопределенных классов.

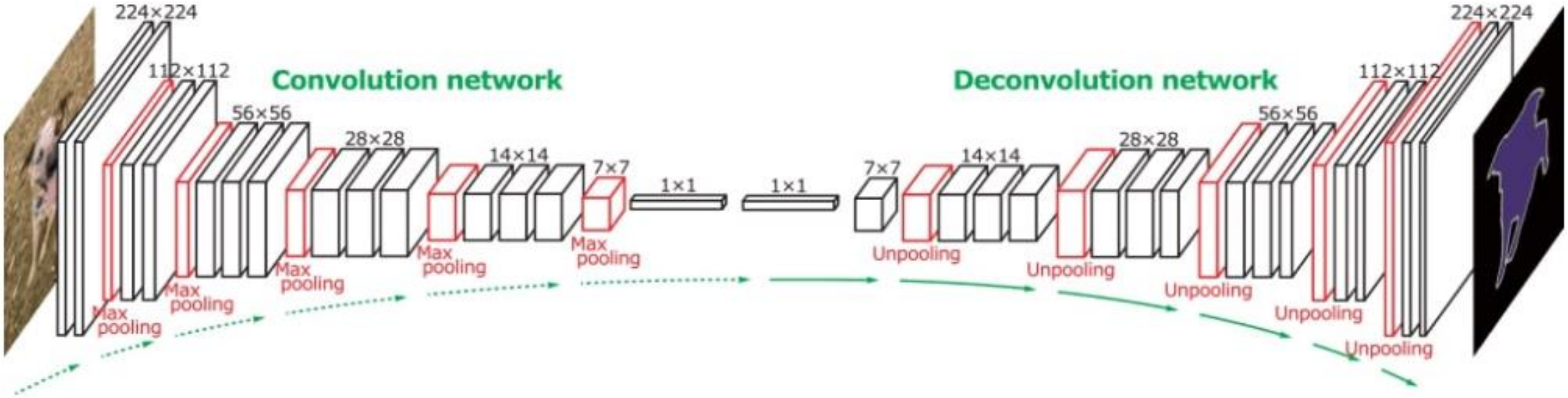
4. **Инстанс-сегментация (Instance Segmentation)**
   Сегментация экземпляров. Более продвинутая форма семантической сегментации, которая не только классифицирует каждый пиксель, но и разделяет разные экземпляры объектов одного класса.

5. **Паноптическая сегментация (Panoptic Segmentation)**
   Объединяет семантическую и инстанс-сегментацию, где каждый пиксель изображения получает либо метку экземпляра (instance), либо метку класса (семантический фон).

Сводная таблица видов сегментации

Вид сегментации	Классификация пикселей	Разделение экземпляров одного класса	Применение
Бинарная сегментация	Два класса (объект и фон)	Нет	Простые задачи выделения объектов
Мультиклассовая сегментация	Несколько классов	Нет	Распознавание нескольких объектов
Семантическая сегментация	Несколько классов	Нет	Задачи классификации на уровне классов
Инстанс-сегментация	Несколько классов	Да	Распознавание и разделение объектов
Паноптическая сегментация	Несколько классов и экземпляров	Да	Сложные сцены с объектами и фоном

Fully Convolutional Networks, FCN (2015)



Предложена для решения задачи семантической сегментации изображений.
Основная идея FCN заключается в том, чтобы **заменить полносвязные** (fully connected) слои в традиционных нейронных сетях (используемых для классификации изображений) **на сверточные** (convolutional) слои, что позволяет модели принимать на вход **изображения любого размера** и создавать выходные карты сегментации для всего изображения.

Ключевые особенности FCN

Полностью свёрточная архитектура:
в отличие от традиционных нейронных сетей, которые содержат полносвязные слои в конце сети (например, AlexNet, VGG), FCN состоит только из свёрточных слоев.
Это делает сеть более гибкой и позволяет работать с изображениями любого размера, сохраняя пространственные связи между объектами на изображении.

Обработка изображений любого размера:
полносвязные слои фиксируют размер входного изображения, но FCN заменяет их свёрточными слоями, что позволяет сети получать на вход изображения произвольных размеров.
Выходом модели является карта классов, которая имеет такое же пространственное разрешение, как и входное изображение, только с уменьшенным размером из-за операций свёртки и пулинга.

Карта сегментации:
вместо одного вектора предсказаний для всего изображения, как в задачах классификации, FCN предсказывает карту сегментации, где каждый пиксель изображения относится к определённому классу (например, «автомобиль», «дорога», «человек»).

Ключевые особенности FCN

Слои деконволюции (Upsampling):
В результате использования свёрток и пуллинга разрешение изображения уменьшается на промежуточных этапах.
Чтобы восстановить пространственную информацию до исходного размера изображения, FCN использует слои деконволюции для увеличения разрешения выходных карт.
Эти слои восстанавливают исходный размер, позволяя модели предсказывать классы для каждого пикселя.

Скип-соединения (Skip Connections):
Чтобы улучшить качество сегментации, FCN использует скип-соединения, которые позволяют объединять высокоуровневые признаки (из глубоких слоёв сети) с низкоуровневыми (из ранних слоев сети).
Это помогает сети учитывать как глобальные, так и локальные признаки при предсказании карты сегментации.

Недостатки FCN

Ограниченность в детализированной сегментации:
Хотя скип-соединения помогают восстанавливать детали, FCN может упускать мелкие объекты на изображении из-за многократных операций пулинга и сверток.

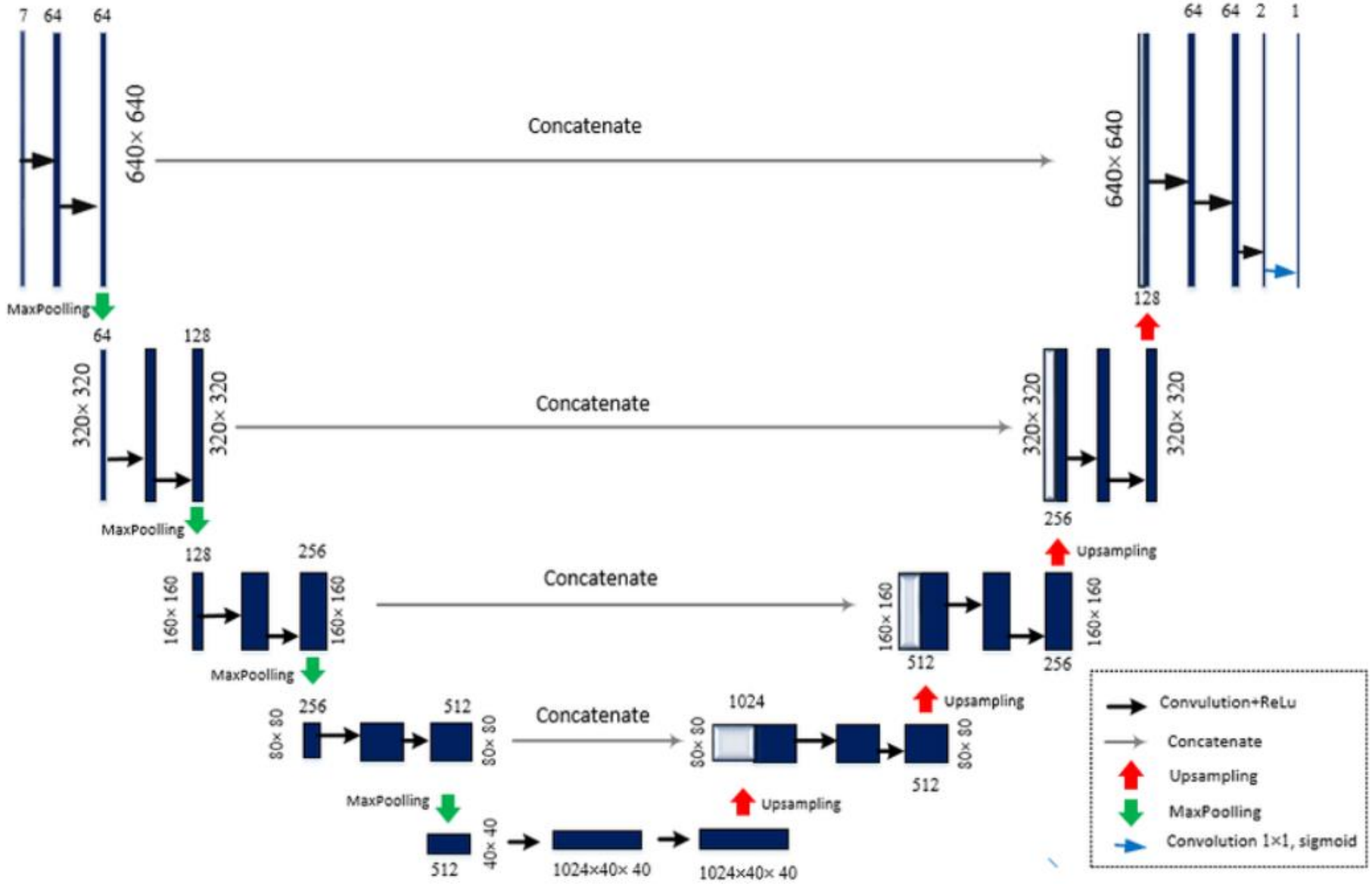
Ограниченные возможности для многоклассовой сегментации:
Сложные сцены с множеством объектов и классов могут потребовать более мощных архитектур, таких как U-Net или DeepLab, для более точного распознавания объектов.

U-Net (2015)

Предложена для решения задач сегментации изображений, в первую очередь, для медицинской визуализации.

Основной особенностью U-Net является ее симметричная структура, где на этапе «**кодирования**» свёртки постепенно уменьшают размер изображения, а на этапе «**декодирования**» применяется процесс восстановления исходного разрешения изображения, что позволяет сети эффективно восстанавливать пространственную информацию.

U-Net получила название благодаря своей U-образной архитектуре. В отличие от FCN, U-Net активно использует **скип-соединения** (skip connections), которые помогают передавать информацию из ранних слоев к более глубоким, чтобы улучшить качество сегментации, особенно для небольших объектов и мелких деталей.


7 64 64
640× 640
Concatenate
64 64 2 1
640× 640
MaxPoolling
64
128
320× 320
320× 320
Concatenate
320× 320
320× 320
128
256
MaxPoolling
Upsampling
128
256
160× 160
160× 160
Concatenate
160× 160
160× 160
512
256
MaxPoolling
Upsampling
256
512
80× 80
80× 80
Concatenate
1024
80× 80
80× 80
512
MaxPoolling
Upsampling
512
40× 40
1024×40× 40
1024×40× 40
Convulution+ReLu
Concatenate
Upsampling
MaxPoolling
Convolution 1×1, sigmoid

Основная структура U-Net

Энкодер (Contracting Path)
Выполняет роль свёрточной сети, которая извлекает признаки из изображения и постепенно уменьшает пространственное разрешение с помощью операций свёртки и пулинга.

Каждый блок энкодера состоит из:
- Двух последовательных свёрточных слоев с нелинейной функцией активации (например, ReLU).
- Max Pooling для уменьшения пространственного разрешения.

Количество фильтров увеличивается на каждом уровне по мере сжатия.

Этот этап «сжимает» изображение, извлекая более высокоуровневые абстрактные признаки, подобно классическим свёрточным нейронным сетям.

Основная структура U-Net

Декодер (Expanding Path)
Восстанавливает пространственное разрешение изображения, используя деконволюционные операции (или транспонированные свёртки).

Каждый блок декодера включает:
- Транспонированную свертку (или upsampling), которая увеличивает разрешение изображения.
- Конкатенацию (skip connections): на этом этапе передаются признаки из соответствующих уровней энкодера, что позволяет модели восстанавливать мелкие детали.
- Две свертки с функцией активации (например, ReLU).

Основная цель декодера — постепенно восстанавливать размер исходного изображения и предсказать классы для каждого пикселя.

Основная структура U-Net

Skip Connections (Скип-соединения)

Скип-соединения соединяют соответствующие слои энкодера и декодера, что помогает сохранить информацию из начальных этапов свёртки, важную для точной локализации объектов.

Эти соединения позволяют передавать как высокоуровневые, так и низкоуровневые признаки, что особенно полезно для распознавания небольших объектов.

Ключевые особенности U-Net

Симметричная архитектура:
U-Net имеет U-образную архитектуру, где количество слоёв свертки и деконволюции сбалансировано, что обеспечивает восстановление пространственной информации.

Скип-соединения,
которые позволяют передавать признаки с высокоразрешенных уровней энкодера в декодер, помогая сети сохранять мелкие детали объектов и улучшать сегментацию.

Отличная производительность на малых данных:
одной из целей U-Net было использование в медицине, где доступные наборы данных могут быть небольшими.
U-Net достигает высоких результатов даже при наличии ограниченных данных благодаря эффективному использованию данных через скип-соединения и симметричную структуру.

Способность обрабатывать изображения произвольного размера:
U-Net не требует фиксированного размера изображений, что делает её гибкой для различных применений.

Недостатки U-Net

Высокие вычислительные затраты:
для обучения U-Net могут требоваться значительные вычислительные ресурсы, особенно при работе с большими изображениями и сложными задачами.

Сложность восстановления мелких деталей:
хотя U-Net хорошо справляется с сегментацией, для очень мелких объектов или объектов, сильно перекрывающихся с другими, сеть может сталкиваться с трудностями.

Модификации U-Net

После разработки оригинальной U-Net было предложено множество модификаций, направленных на улучшение точности и производительности:

- **3D U-Net**: применяется для сегментации трёхмерных данных, таких как медицинские изображения с несколькими срезами (3D-томография).
- **Attention U-Net**: включает механизмы внимания (attention), чтобы улучшить сегментацию областей с важной информацией.
- **ResU-Net**: использует остаточные блоки (ResNet) для улучшения градиентного потока в сети и увеличения точности сегментации.

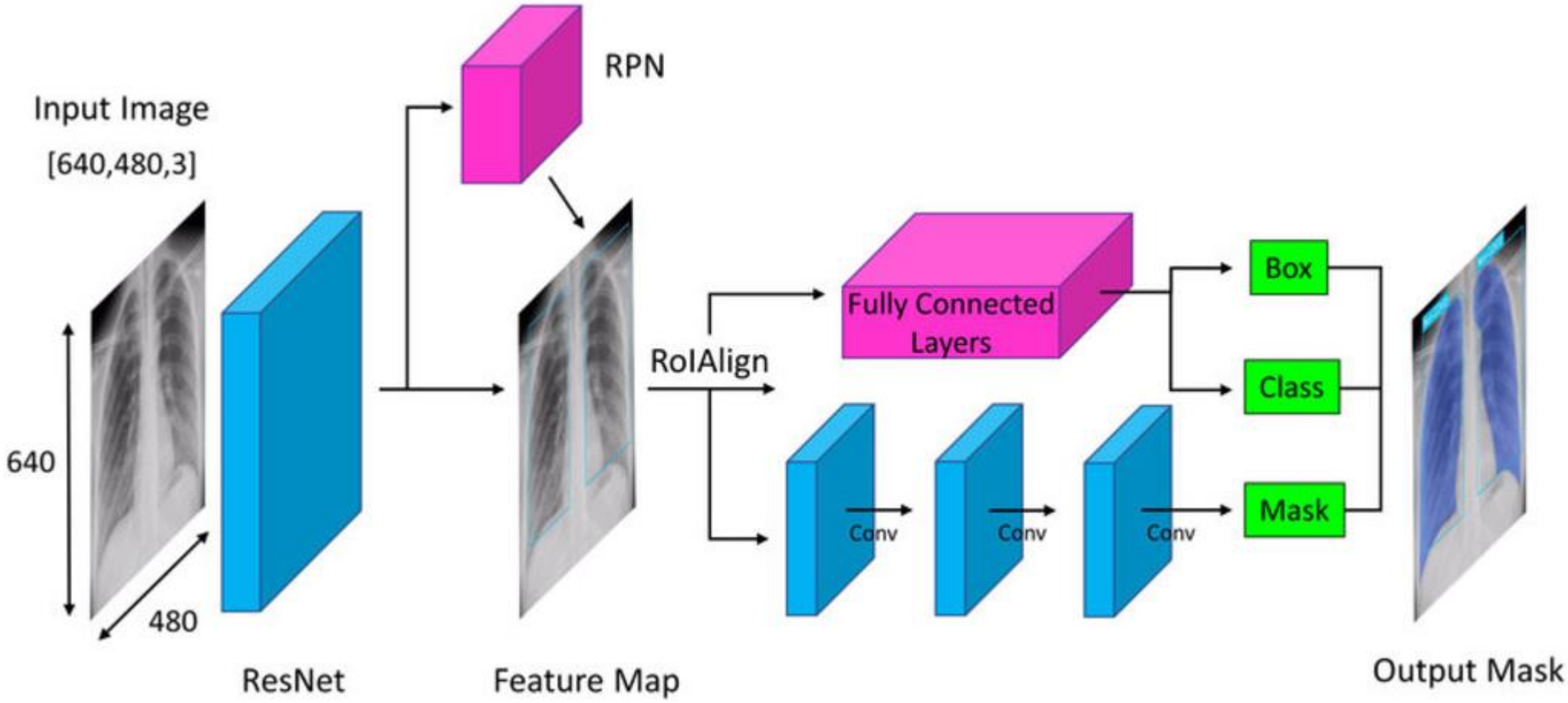
Mask R-CNN (2017)

Архитектура нейронной сети для задач детекции объектов и сегментации на уровне пикселей.

Mask R-CNN является расширением Faster R-CNN, который изначально предназначен для детекции объектов.

Основное новшество Mask R-CNN заключается в добавлении механизма сегментации, который позволяет точно определять контуры объектов, обеспечивая возможность выделения **не только классов объектов, но и их формы**.

Mask R-CNN (2017)



Основные компоненты Mask R-CNN

Backbone сеть:
Mask R-CNN использует предобученные модели, такие как ResNet-50 или ResNet-101, в сочетании с Feature Pyramid Network (FPN).

ResNet отвечает за извлечение признаков из изображения, представляя их в компактной и информативной форме.

FPN помогает эффективно работать с объектами разных масштабов, что особенно важно для обработки изображений, где присутствуют как мелкие, так и крупные объекты.

Backbone отвечает за извлечение характеристик изображения, которые затем используются для детекции и сегментации.

Backbone-сеть

Основная сеть извлечения признаков — это компонент модели глубокого обучения, особенно в компьютерном зрении.

Он получает исходные данные, например изображение, и преобразует их в набор высокоуровневых характеристик. Эти карты признаков отражают такие важные детали, как края, текстуры и формы.

Как правило, в качестве backbone используется **глубокая конволюционная нейронная сеть (CNN)**, которая была предварительно обучена на крупномасштабном наборе данных классификации, например **ImageNet**.

Такое предварительное обучение (трансферное обучение) позволяет сети изучить обширную библиотеку общих визуальных признаков.

Некоторые архитектуры backbone-сетей

ResNet — используют «пропускные связи», позволяющие обучать более глубокие сети за счёт смягчения проблемы исчезающего градиента.

EfficientNet — семейство моделей, которое использует метод комбинированного масштабирования, который равномерно балансирует глубину, ширину и разрешение сети.

Vision Transformer (ViT) — архитектура, которая адаптирует модель трансформера из обработки естественного языка (NLP) для задач зрения. ViT обрабатывают изображения как последовательности патчей и используют самовнимание для захвата глобального контекста.

CSPNet (Cross Stage Partial Network) — архитектура, которая повышает эффективность обучения за счёт разделения карт признаков для уменьшения узких мест в вычислениях.

Основные компоненты Mask R-CNN

Region Proposal Network (RPN):
генерирует кандидатные области (region proposals) для объектов на изображении.

Она использует карты признаков от backbone-сети и предсказывает, где могут находиться объекты.

RPN производит набор прямоугольников, которые потенциально могут содержать объекты, и помогает уменьшить количество областей, которые нужно анализировать на последующих этапах.

ROI Align:
данный механизм улучшает точность сегментации и позволяет извлекать фиксированные размеры признаков из карт, соответствующих кандидатурам, с использованием интерполяции.

В отличие от ROI Pooling, который округляет координаты, ROI Align **сохраняет точные координаты** пикселей, что критично для задач сегментации.

Основные компоненты Mask R-CNN

Mask Prediction Branch:
Mask R-CNN добавляет дополнительную ветвь для предсказания масок сегментации для каждого из объектов в области интереса (ROI).

Эта ветвь генерирует бинарные маски для каждого объекта, указывая, какие пиксели принадлежат каждому классу объекта.
Маски имеют размер 28x28 пикселей (или другой фиксированный размер) и интерполируются до размеров соответствующих ROI.

Bounding Box Regression:
в дополнение к сегментации, Mask R-CNN также предсказывает ограничительные рамки (bounding boxes) для объектов, аналогично Faster R-CNN.

Эти рамки помогают точно определить расположение объектов на изображении.

Достоинства Mask R-CNN

Точная сегментация объектов на уровне пикселей, что позволяет выявлять детали, которые важны для анализа.

Универсальность: можно использовать для множества задач, включая детекцию объектов, семантическую сегментацию и инстанс-сегментацию, что делает её универсальным инструментом в компьютерном зрении.

ROI Align: улучшает **точность предсказаний**, особенно в задачах, где важна **детальная** сегментация.

Быстрая и эффективная работа, что позволяет использовать ее в **реальном времени** в различных приложениях, включая видеоаналитику и автономные системы.

Недостатки Mask R-CNN

Требует значительных вычислительных ресурсов, особенно при использовании глубоких архитектур в качестве backbone, что может быть проблемой для приложений с ограниченными ресурсами.

Архитектура Mask R-CNN **требует тонкой настройки гиперпараметров** и может быть **чувствительна к выбору предобученных** весов, что может усложнить процесс обучения.

Проблемы с малым количеством объектов: в случаях, когда на изображении много объектов одного класса или объекты сильно перекрываются, Mask R-CNN может испытывать трудности с правильным разделением их на инстансы.