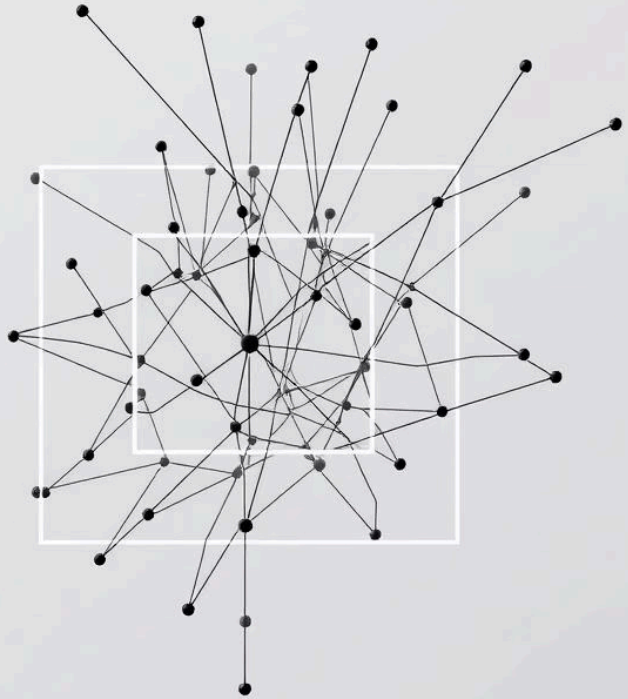
Повышение точности малых моделей детекции объектов

Плугин Олег 22503



Повышение точности малых моделей детекции объектов

Необходимо найти баланс между высокой точностью и эффективным использованием ресурсов

Постановка задачи: обнаружение объектов на ограниченных ресурсах

В задачах компьютерного зрения для мобильных и встроенных устройств, критически важно обеспечить высокую точность обнаружения объектов при сохранении скорости работы в реальном времени и минимальных вычислительных затратах.

Стандартные ограничения устройств

Процессор

4-ядерный, тактовая частота 2.2 ГГц.

Оперативная память

До 4 ГБ ОЗУ.

Графический процессор

Обработка изображений размером до 1920×1080.
Оптимизация под мобильные GPU-архитектуры.

Проблема и цель повышения точности

При приведенных аппаратных ограничениях использование крупных нейронных сетей (например, стандартных версий YOLO или Faster R-CNN) становится невозможным. Требуется упрощение архитектуры:

- Уменьшение числа слоев и их глубины.
- Сокращение количества фильтров (каналов).
- Снижение общего числа обучаемых параметров.

Основная проблема: Упрощение архитектуры нейронной сети приводит к значительному снижению точности (mAP) распознавания.

Ключевой целью является достижение точности, максимально приближенной к исходной, более крупной модели, при существенно меньшем размере панели и более высокой скорости работы.

Параметры задачи

Категория	Основные параметры	Влияет на
Архитектура сети	количество слоёв n, шаг карты признаков (stride feature map) $feat_{st}$, количество фильтров f	скорость работы (FPS) и точность (mAP)
Входные данные	размер входного изображения (W, H)	точность распознавания
Якори (anchors)	количество якорных рамок k и их размеры (w_a, h_a)	качество локализации объектов и IoU
Метрики оценки	mAP (mean Average Precision)	среднее значение точности по всем классам и порогам IoU. Основная метрика для сравнения моделей (ед.: $\%$)
	$Precision$	доля найденных моделью объектов действительно является корректной (ед.: $\%/0–1$)
	$Recall$	доля реальных объектов, найденная моделью (ед.: $\%/0–1$)
	IoU (Intersection over Union)	точность покрытия предсказанными областями исходных (ед.: $0–1$)
	FPS (frames per second)	количество обрабатываемых кадров в секунду (ед.: $\frac{\text{кадры}}{\text{сек}}$)
	T_1	время обработки одного кадра (ед.: мс)

Формулировка задачи оптимизации

Данную задачу можно представить в виде задачи оптимизации.

Пусть существуют параметры:

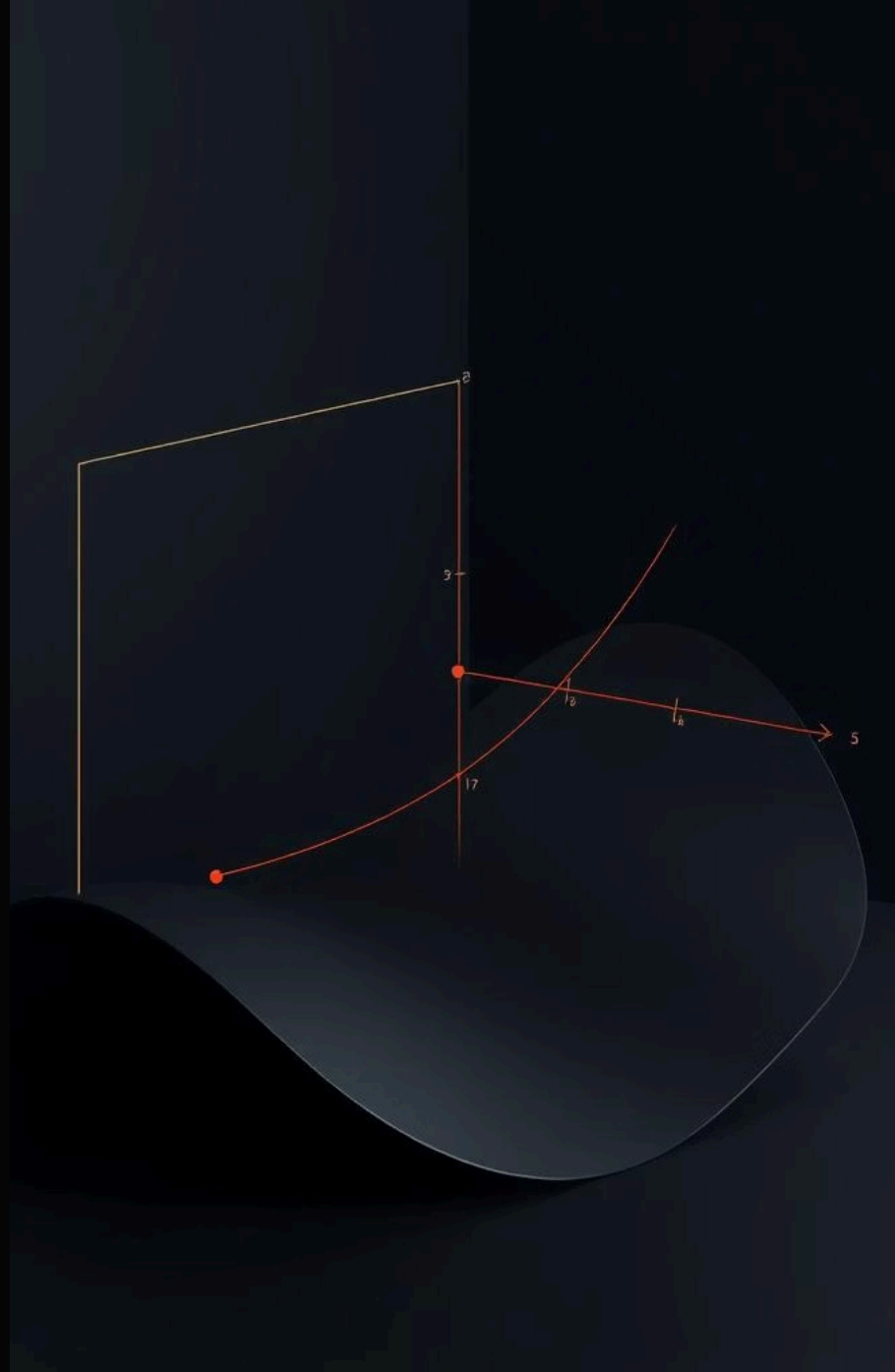
- W - ширина входного изображения (пиксели) [1]
- H - высота входного изображения (пиксели) [2]
- n - количество слоёв нейронной сети [3]
- k - количество якорных рамок [4]

Необходимо набрать такой набор параметров (W, H, n, k) при котором:

- точность $mAP \to \max$ [7]
- частота кадров $FPS \to \max$ (т.к. $FPS = \frac{1}{T_1}$, а $T_1 \to \min$) [8]

При выполнении следующих ограничений:

- $n < n_{max}$ [3]
- mAP_{raw} = mAP_{n_0} - Δ_{raw} [6] - это нижняя граница mAP при неоптимизированном наборе параметров (W, H, k), где Δ_{raw} - допустимое снижение точности
- $mAP(n) \geq mAP_{raw} + \Delta$ [7], где Δ - желаемый прирост точности в процентах (задается исходя из характеристик задачи)
- $FPS(W, H, n, k) \geq FPS_{min}$ [8] [5] (FPS_{min} задается исходя из задачи)



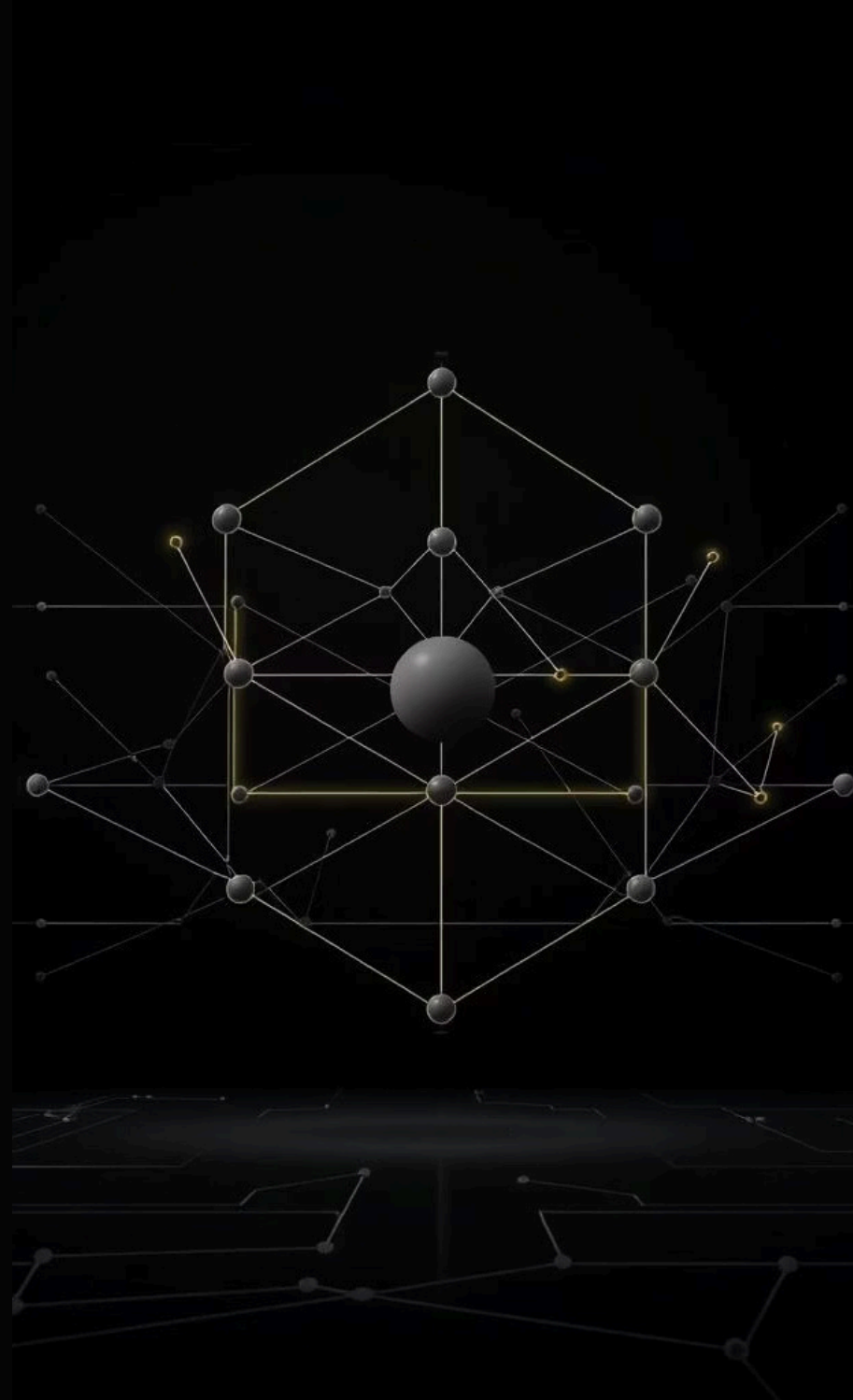
Алгоритм повышения точности сети: Архитектура и входные данные

Анализ архитектуры

1. Зафиксировать начальное количество слоев как n_0
2. Уменьшить количество слоев до n [3] $< n_0$ следующим образом:
   - Установить диапазон поиска $n \in [n_{min}; n_{max}]$ (изначально $n_{min} = 5, n_{max} = n_0 - 1$)
   - Вычислить $n_{mid} = (n_{min} + n_{max})/2$
   - Измерить метрики: $mAP(n_{mid})$, $FPS(n_{mid})$ [7][8]
   - При $FPS(n_{mid}) \geq FPS_{min}$ [5] и $mAP(n_{mid}) \geq mAP_{raw}$ [6] уменьшаем глубину модели $n_{max} = n_{mid}$
   - Иначе $n = n_{max}$ [3] (находим последнее удовлетворяющее значение)

Изменение входных данных

1. Изменить размер входного изображения $W_n \times H_n$ [1][2] по числам, кратным 32 (примеры: уменьшение $640 \to 416 \to 320$, увеличение $416 \to 512 \to 640$)
2. Оценить изменения mAP и FPS [7][8]
3. Отметить зависимость: $mAP \propto f_1(W_n, H_n)$, $FPS \propto f_2(W_n, H_n)$ - (время обработки обратно пропорционально частоте кадров). f_1, f_2 - функции, описывающие зависимость точности и количества кадров от разрешения входного изображения соответственно



Алгоритм повышения точности сети: Оптимизация якорных рамок

$$(w_r, h_r) = (W_n / W_i \times w_t, H_n / H_i \times h_t)$$

Формула обеспечивает пропорциональное масштабирование якорных рамок под новый размер.

$$(w_c, h_c) = k\text{-means}(w_r, h_r)$$

$$\bar{S}(k) = \frac{1}{n} \sum_{i=1}^{n} S(i)$$

$$S_k = (b - a) / \max(a, b)$$

1. Масштабировать рамки по формуле:
   - (w_t, h_t) - граничные рамки исходного значения (пиксели)
   - (w_r, h_r) - ремасштабированные рамки (пиксели)
   - (W_n, H_n) - исходное разрешение входа сети (пиксели)
   - (W_i, H_i) - ширина и высота входного изображения (пиксели)
2. Провести кластеризацию k-средних по ширине и высоте рамок:
   - (w_c, h_c) - k центров кластеров, соответствующие усредненным размерам объектов
3. Найти оптимальное количество кластеров k [4] с помощью силуэта средних коэффициентов:
   - Проэкспериментировать со значением k от 3 до 15
   - При каждой итерации вычислять средний силуэт по всем точкам:
     - n - общее количество элементов, участвующих в кластеризации
     - a - среднее внутрикластерное расстояние,
     - b - ближайшее межкластерное расстояние
   - Максимум среднего $\bar{S}(k)$ определит оптимальное значение k

Алгоритм повышения точности сети: Нормализация якорей и оценка результатов

Нормализация якорей

Нормализуем размеры якорных рамок

$$(w_a, h_a) = (w_c / feat_{st}, h_c / feat_{st})$$

- w_a, h_a - размеры рамок в масштабе feature map, то есть в тех единицах, с которыми работает последний слой модели
- $feat_{st}$ = 2^n - шаг карты признаков, где сеть содержит n слоев max pooling

Это необходимо, чтобы размер якорей соответствовал масштабу признаков на последнем уровне сети.

Оценка результатов

1. Измерить метрики $Precision$, $Recall$, mAP, FPS, IoU
2. Проанализировать зависимости параметров
   - увеличение W_n, H_n [1][2] $\rightarrow$ рост mAP, падение FPS
   - уменьшение n [3] $\rightarrow$ рост FPS, падение точности mAP
   - увеличение k [4] $\rightarrow$ рост IoU
3. Сделать выводы, удовлетворяют ли текущая конфигурация и метрики целевым ограничениям

Заключение

Представленный подход позволяет подбирать архитектурные и входные параметры модели под доступные вычислительные ресурсы.

Сопоставимые показатели

Достижение показателей, сопоставимых с исходной моделью при оптимизации архитектуры.

Эффективность ресурсов

Значительно меньшая сложность модели, что важно для систем компьютерного зрения, работающих на мобильных и малопроизводительных устройствах.

Пример результатов работы алгоритма

Модель	n [3]	k [4]	W, H (px) [1][2]	Recall (%)	Precision (%)	Avg. IoU (%)	mAP (%) [7]	FPS (Кадры/сек) [8]
Базовая модель	24	3	416	80	78	75	72	5–20
Оптимизированная модель	9	9	608	85	84	82	82.21	50–100

Источники

1. Tarek Teama, Hongbin Ma, Ali Maher, and Mohamed A. Kassab. 2019. Real Time Object Detection Based on Deep Neural Network. In Intelligent Robotics and Applications: 12th International Conference, ICIRA 2019, Shenyang, China, August 8–11, 2019, Proceedings, Part IV. Springer-Verlag, Berlin, Heidelberg, 493–504.