https://doi.org/10.31891/2307-5732-2025-357-11
УДК 004.8

ГОЗАК ЯРОСЛАВ
Київський національний університет імені Тараса Шевченка
https://orcid.org/0009-0008-2963-7204
e-mail: hozakya@fit.knu.ua
ПАЛІЙ СЕРГІЙ
Київський національний університет імені Тараса Шевченка
https://orcid.org/0000-0001-9742-1116
e-mail: paliy@fit.knu.ua

ПОРІВНЯЛЬНИЙ АНАЛІЗ НЕЙРОННИХ МЕРЕЖ РОЗПІЗНАВАННЯ ОБ'ЄКТІВ НА ЗОБРАЖЕННЯХ

В роботі проведено аналіз ключових етапів розвитку нейронних мереж для розпізнавання об'єктів — від ранніх двоетапних архітектур (R-CNN, Fast R-CNN) до сучасних одноетапних моделей родини YOLO. Детально досліджено революційний перехід від пропозицій регіонів до end-to-end детекції, зокрема внесок YOLO в обробку зображень у реальному часі. Особливу увагу приділено еволюції архітектур YOLO: вдосконаленню точності, швидкодії та адаптації до обмежених ресурсів (від v1 до v8). Показано, як інновації — якірні рамки, FPN/PAN, CSPBackbone, квантування — вирішували проблеми попередників (недостатня швидкість, низька точність дрібних об'єктів, висока ресурсомісткість). Підкреслено практичну значущість сучасних YOLO-моделей у задачах автономного транспорту, систем безпеки та промислової автоматизації.

***Ключові слова**: R-CNN, YOLO, розпізнавання об'єктів, комп'ютерний зір*

HOZAK YAROSLAV
PALIY SERGIY
Taras Shevchenko National University of Kyiv

NEURAL NETWORKS FOR OBJECT RECOGNITION IN IMAGES COMPARATIVE ANALYSIS

This article presents a systematic analysis of the architectural evolution in object detection networks, tracing pivotal advancements from the pioneering Region-based Convolutional Neural Network (R-CNN) framework to contemporary YOLOv8 implementations. Early two-stage methodologies, including Fast R-CNN and Faster R-CNN, established foundational region-proposal paradigms but faced inherent computational inefficiencies. The paradigm shift toward unified, single-shot detection was catalyzed by the Single Shot Detector (SSD) and You Only Look Once (YOLO) architecture, which enabled real-time inference through end-to-end spatial grid processing.

Subsequent iterations of YOLO demonstrate progressive resolution of critical limitations. YOLOv1-v3 introduced multi-scale feature hierarchies via Feature Pyramid Networks (FPN) and anchor-based localization, enhancing small-object detection. YOLOv4 integrated cross-stage partial networks (CSPDarknet) and path aggregation (PANet) to optimize gradient flow, while advanced augmentation strategies improved robustness. YOLOv5/v6 refined these principles with hardware-aware optimizations, supporting streamlined deployment via ONNX and TensorRT runtimes.

Later innovations address persistent challenges: YOLOv7's extended efficient layer aggregation (E-ELAN) enhanced parameter utilization, whereas YOLOv8's anchor-free mechanism and Distribution Focal Loss (DFL) significantly improved bounding box precision. Throughout this evolution, techniques such as decoupled heads and post-training quantization have progressively mitigated constraints related to computational latency, model size, and edge-device compatibility.

Quantitative advances in mean average precision (mAP) and frames-per-second (FPS) metrics are contextualized against real-world applications in autonomous navigation, industrial automation, and surveillance. The study concludes by identifying emergent research trajectories, including lightweight model distillation, 3D scene understanding, and multimodal vision-language integration.

***Keywords:** R-CNN, YOLO, object recognition, computer vision.*

Стаття надійшла до редакції / Received 12.08.2025
Прийнята до друку / Accepted 286.08.2025

Вступ

Виявлення об'єктів є фундаментальним завданням комп'ютерного зору, яке поєднує локалізацію та класифікацію об'єктів на зображеннях. Протягом останнього десятиліття поширення методів глибинного навчання – зокрема згорткових нейронних мереж (ЗНМ) – суттєво підвищило рівень розвитку у цій сфері, забезпечивши стійку роботу в різноманітних і складних умовах.

Ця робота пропонує структурований огляд методів виявлення об'єктів, заснованих на нейронних мережах, з акцентом на методологічних інноваціях та емпіричній ефективності. Ми починаємо з контекстуалізації еволюції фреймворків виявлення об'єктів, простежуючи шлях від ранніх регіон-орієнтованих методів до уніфікованих детекторів реального часу. Особливу увагу приділено архітектурним рішенням, парадигмам навчання, залежності від наборів даних і обчислювальним компромісам, що характеризують кожне сімейство моделей.

Синтезуючи теоретичні висновки та практичні оцінки, цей огляд має на меті надати просунутий погляд на сучасні конвеєри виявлення об'єктів, що базуються на архітектурах нейронних мереж.

Моделі виявлення об’єктів на основі ЗНМ

R-CNN (Regions with Convolutional Neural Networks)

Згорткова нейронна мережа на основі регіонів (R-CNN) [1], запропонована Гіршиком та ін. (2014), представляє основоположний прогрес у виявленні об’єктів шляхом ефективної інтеграції механізмів пропозицій області з глибоким згортковим виділенням ознак. Цей метод значно перевершив традиційні детектори об’єктів завдяки використанню глибокого навчання для завдань представлення ознак і класифікації. R-CNN використовує багатоступеневий конвеєр виявлення (рис. 1).

Спочатку за допомогою алгоритму вибіркового пошуку створюється набір із приблизно 2000 пропозицій регіонів, незалежних від категорії. Ці пропозиції слугують обмежувальними рамками-кандидатами, які потенційно містять об’єкти.

Кожна пропозиція регіону змінюється до фіксованого розміру (наприклад, 227×227 пікселів) і незалежно проходить через згорткову нейронну мережу (зазвичай AlexNet), попередньо навчену на наборі даних ImageNet. ЗНМ витягує високовимірний вектор ознак (наприклад, повнозв’язаний шар розмірністю 4096) для кожного регіону.

Отримані ознаки для кожної пропозиції вводяться в набір специфічних для класу лінійних опорних векторних машин (SVM), які навчені призначати мітки класів і відрізняти категорії об’єктів від фону.

На окремій стадії регресії специфічний для класу лінійний регресор уточнює розташування кожної пропозиції регіону, навчаючись прогнозувати більш точний обмежувальний прямокутник, таким чином підвищуючи точність локалізації. Загальна схема роботи наведена на рисунку 1.

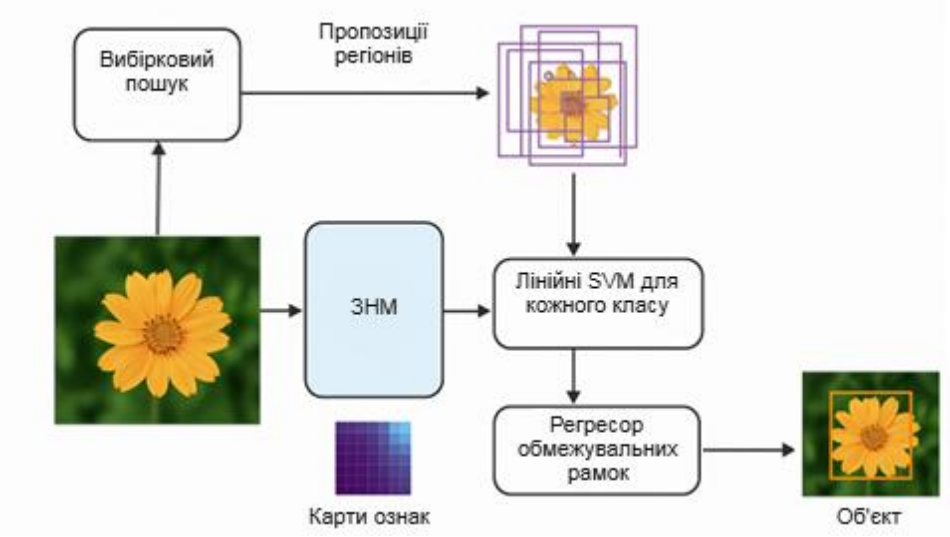

Рис. 1. Діаграма роботи R-CNN

Час обчислення для одного зображення становить приблизно від 18 до 47 секунд залежно від апаратного забезпечення. Отже, є багато можливостей для вдосконалення, враховуючи, що більшу частину часу займає стадія регіональної пропозиції.

Fast R-CNN

Fast R-CNN [2] — це регіональна згорткова нейронна мережа, яка покращує оригінальну R-CNN, збільшуючи швидкість і точність виявлення. Мережа обробляє все зображення за допомогою спільної згорткової мережі та застосовує об’єднання регіонів інтересу (RoI), щоб витягти об’єкти фіксованого розміру із запропонованих регіонів. Оригінальний підхід R-CNN вимагав окремого виділення ознак для кожної пропозиції регіону, що призводило до значної надмірності та високих обчислювальних витрат. Fast R-CNN інтегрує обробку пропозиції регіону в спільну карту ознак, забезпечуючи більш ефективне та наскрізне виявлення, яке можна навчити.

Як зазначено в статті [2], основними недоліками R-CNN є:

Багатоетапне тренування.

Навчання дороге по пам’яті та в часі. Для навчання SVM і регресора обмежувальної рамки ознаки витягуються з кожного об’єкта пропозиції для кожного зображення.

Виявлення об'єктів відбувається повільно. Оскільки ознаки витягуються з кожної пропозиції об’єкта в кожному тестовому зображенні, час виявлення за допомогою ЗНМ VGG16 займає 47 секунд на зображення (на GPU).

Замість того, щоб генерувати ознаки окремо для кожної пропозиції регіону, Fast R-CNN обробляє все зображення один раз через спільну згортку (наприклад, VGG16), створюючи карту ознак для всіх пропозицій.

Ключовим нововведенням є рівень RoI Pooling, який витягує карти ознак фіксованого розміру зі спільних карт, усуваючи зайві обчислення. Він використовує пропозиції зовнішніх функцій, згенеровані, як правило, алгоритмом вибіркового пошуку. Потім все вхідне зображення передається

через згортову мережу для обчислення спільної карти ознак. Пропозиції проєктуються на цю карту ознак з використанням відомого просторового масштабу. Рівень RoI Pooling виділяє об'єкти фіксованого розміру з кожного регіону на карті ознак. Нарешті, мережа розгалужується на два рівні вихідні шари:

- Класифікатор Softmax $P(c \mid r_i)$, по $K+1$ категорій (включаючи фон).
- Регресор обмежувальної рамки для кожного класу c, що виводить 4-вимірний вектор (1) що представляє параметризовані зсуви обмежувальної рамки.

$$t^c = (t_x^c, t_y^c, t_w^c, t_h^c) \quad (1)$$

На відміну від R-CNN, яка вимагає кілька етапів навчання, Fast R-CNN використовує багатозадачну функцію втрати для одночасної оптимізації класифікації та регресії обмежувальної рамки. Вона використовує багатозадачну функцію втрат L для спільної оптимізації класифікації та регресії обмежувальної рамки:

$$L(p, u, t^u, v) = L_{cls}(p, u) + \lambda \cdot [u \geq 1]\, L_{loc}(t^u, v) \quad (2)$$

Тут [u≥1] вказує на те, що втрати локалізації обчислюються лише для класів об'єктів (не фону). Ця умова дозволяє збалансувати кількість тренувань для фону і класів об'єктів.

Ефективність обчислень досягає ~2,3 секунди на зображення на тому самому графічному процесорі, що й для R-CNN, тому час обробки зображення під час тесту зменшується більш ніж у 140 разів.

Ми можемо зазначити такі обмеження Fast R-CNN:

1. Алгоритм покладається на зовнішні методи пропозиції регіонів (вибірковий пошук), які є повільними та не піддаються навчанню.
2. Регіональні пропозиції не оптимізовані спільно з мережею виявлення.

Faster R-CNN: на шляху до виявлення об'єктів у реальному часі за допомогою мереж регіональних пропозицій

Faster R-CNN [3] був запропонований для усунення основного вузького місця у продуктивності Fast R-CNN – повільного етапу пропозиції регіону, який до того не підлягає навчанню. Незважаючи на те, що Fast R-CNN значно покращує час роботи за рахунок спільного використання карти ознак між пропозиціями регіонів, створення цих пропозицій все ще залежить від зовнішніх методів, таких як вибірковий пошук. Ці алгоритми дорогі в обчислювальному плані (~2 с/зображення) і не оптимізовані як частина конвеєра виявлення.

Основною інновацією Faster R-CNN є запровадження мережі регіональних пропозицій (RPN), повністю згорткової мережі, яка спільно з мережею виявлення об'єктів використовує спільну карту ознак. Вона створює пропозиції щодо регіонів безпосередньо з карти ознак. RPN пересуває згорткове вікно розміром 3x3 над спільною картою ознак. У кожному просторовому місці (k) вона генерує:

- k показників об'єктності (передній план проти фону).
- $4k$ зміщення обмежувальної рамки (параметризовані t_x, t_y, t_w, t_h) відносно до k якорів (прив'язок).

Таким чином, результат RPN складається з $2k + 4k = 6k$ значень на кожне розташування, де $2k$ походить від окремих показників об'єктності для об'єкту та для фонового класу.

Іншим внеском є впровадження анкерних рамок. Мережа використовує попередньо визначений набір просторових прив'язок із різними масштабами та співвідношеннями сторін. RPN ковзає по спільних згорткових ознаках, щоб «передбачити» регресії обмежувальної рамки відносно попередньо визначених опорних рамок у кожному місці.

RPN навчається спільно з конвеєром виявлення об'єктів через багатозадачне навчання. Навчання включає в себе чотириетапне навчання, яке включає окреме навчання RPN і R-CNN на попередньо ініціалізованій ЗНМ з подальшим тонким налаштуванням RPN і R-CNN за допомогою спільних згорткових рівнів.

Продуктивність досягає ~0,3 секунди на зображення, що в 10 разів швидше, ніж Fast R-CNN.

Faster R-CNN представляє зміну парадигми виявлення об'єктів шляхом усунення зовнішніх методів пропозицій та об'єднання конвеєра виявлення в єдину структуру, яку можна навчити. Вона заклала фундаментальну архітектуру виявлення об'єктів на основі глибокого навчання.

SSD: Однопрохідний детектор (Single Shot MultiBox Detector)

SSD [4] було запропоновано, щоб ліквідувати розрив між високою точністю двоетапних детекторів (Fast R-CNN, Faster R-CNN) і потребою в роботі в реальному часі. Мета полягає в тому, щоб виконувати класифікацію та локалізацію об'єктів за один прямий прохід нейронної мережі, суттєво пришвидшуючи роботу без значної втрати точності .

Модель накладає фіксований набір рамок за замовченням (default bounding boxes) у кожній точці карт ознак. Для кожної карти передбачено рамки з п'ятьма різними співвідношеннями сторін {1, 2, 0.5, 3, 0.33} і п'ятьма масштабами {0.2, 0.375, 0.55, 0.725, 0.9}. Під час навчання мережа коригує ці рамки під реальні об'єкти.

Архітектурно SSD використовує згорткову "спину" (зазвичай VGG-16 або MobileNet) і додає чотири додаткові згорткові шари, щоб генерувати карти ознак різного масштабу (рис. 2). Це дає змогу визначати об'єкти різного розміру.

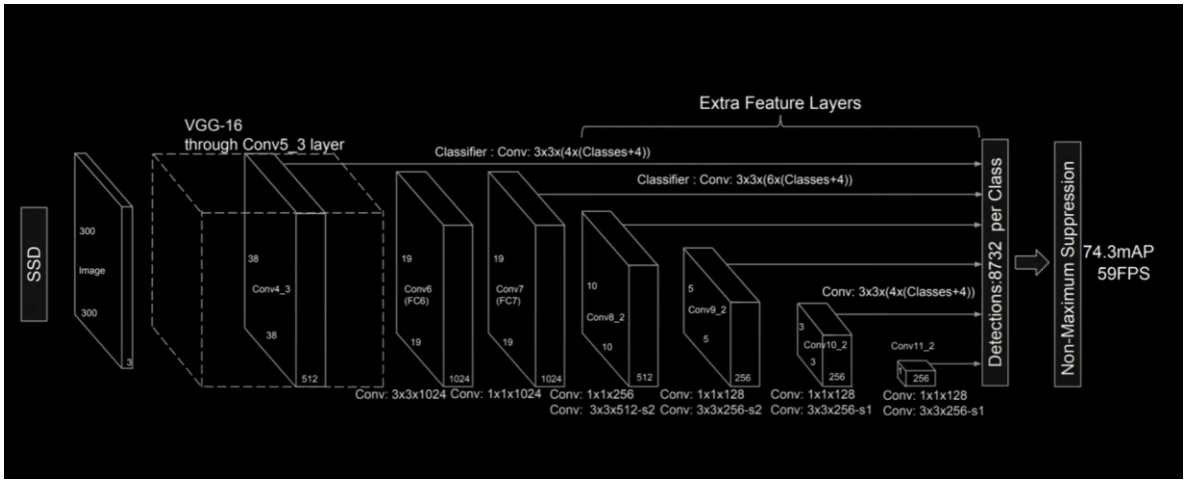

Рис. 2. Архітектура SSD [20]

Автори використовують жорсткий негативний аналіз (hard negative mining) і доповнення даних, щоб зробити детектор надійним.

Для навчання SSD використовує багатозадачну втрату, що поєднує класифікацію та локалізацію:

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)) \quad (3)$$

SSD досягає продуктивності в режимі реального часу (25–46 FPS) залежно від роздільної здатності вхідного сигналу та ЗНМ. Немає потреби в стадії пропозиції регіонів чи об'єднанні ROI. SSD дещо втрачає в точності, проте значно збільшує швидкість.

SSD залишається одним з найвпливовіших одноступеневих детекторів, що служить основою для більш пізніх архітектур, таких як MobileNet-SSD, Tiny-SSD та інших. Однак SSD має такі обмеження:

1. Погана продуктивність на дрібних об'єктах. SSD використовує стандартні блоки, що застосовуються на грубих картах об'єктів (наприклад, 38×38, 19×19, ..., 1×1). Рецептивні поля цих карт великі і недостатньо точні для локалізації малих об'єктів.

2. Нерухомі анкерні рамки. SSD використовує попередньо визначений набір стандартних блоків із фіксованими співвідношеннями сторін та масштабами на кожному рівні карти об'єктів. Це робить мережу менш гнучкою при роботі з об'єктами з формами, які погано узгоджуються з попередньо визначеними конфігураціями.

3. Немає чіткої оцінки об'єктності. На відміну від RPN (у Faster R-CNN), SSD явно не передбачає ймовірність того, що рамка містить саме об'єкт, а не фон. Він безпосередньо класифікує кожне поле за замовчуванням до одного з класів об'єктів C або фону, що робить його більш сприйнятливим до помилкових спрацьовувань, особливо в захаращених сценах з неоднозначним(неоднорідним) фоном.

You Only Look Once: Уніфіковане виявлення об'єктів у реальному часі

YOLO також був представлений у 2016 році P. Гіршиком [5]. Фреймворк був запропонований для переформулювання виявлення об'єктів як єдиної регресійної задачі, що дозволяє виявляти об'єкти в реальному часі шляхом безпосереднього прогнозування обмежувальних рамок і ймовірностей класів з повних зображень за один прохід згорткової нейронної мережі.

Як і SSD, фреймворк YOLO виконує прогнозування на основі сітки. Вхідне зображення поділяється на сітку S×S, де S зазвичай вибирається як 7. Кожна клітинка сітки передбачає:

1. Визначену кількість обмежувальних рамок (автори використовували значення 2), кожна з яких містить:

Координати обмежувальної рамки відносно комірки сітки (x, y, w, h)

Оцінка впевненості (IoU між передбачуваною рамкою та реальною)

2. Умовні ймовірності класу

Кожна прогнозована обмежувальна рамка включає координати центру щодо комірки сітки (x, y), ширину і висоту відносно зображення (w, h), впевненість = $P(object) * IoU_{pred}^{truth}$

На відміну від інших моделей, описаних вище, YOLO базується на створеній авторами згортковій архітектурі на основі InceptionV1, що називається YOLOv1-CNN, яка включає 24 згорткових шари, за якими слідують 2 повнозв'язних шари.

Функція втрати враховує втрату помилок у квадраті суми, що враховує помилки класифікації, впевненості та локалізації координат, а також бере до уваги помилку визначення ширини/висоти для обмежувальної рамки, яка має найвищий IoU з анотованим об'єктом. Оскільки загальні втрати обчислюються по всіх комірках і більшість комірок не містять об'єктів, градієнт для комірок noobj

перевершує інші градієнти. З цієї причини автори вводять гіперпараметри λcoord та λnoobj щоб збалансувати локалізацію та втрату впевненості.

Таким чином, YOLO пропонує наступні інновації:

1. Єдиний конвеєр детектування. YOLO прогнозує як ймовірності класу, так і координати обмежувальної коробки на основі повних зображень за один прохід нейронної мережі, без механізмів регіональних пропозицій або ковзних вікон.

2. Міркування глобального контексту. Оскільки YOLO бачить зображення повністю під час тренування та висновків, він фіксує контекстуальну інформацію глобально, покращуючи узагальнення до незвичайних сцен.

3. Виявлення в реальному часі. Досягає хорошої швидкості (~45 fps на графічному процесорі), зберігаючи конкурентоспроможність.

4. Дана модель має ряд обмежень, найголовнішими з яких є помилка локалізації, оскільки YOLO важко надати точну локалізацію, особливо для невеликих об'єктів. Також обмеженість сітки через те що кожна клітинка сітки може виявляти лише один об'єкт, що робить її менш ефективною, коли кілька малих об'єктів потрапляють в одну клітинку.

YOLO9000: Краще, швидше, сильніше

YOLOv2 (YOLO9000) [6] було запропоновано для усунення цих обмежень шляхом покращення як точності, так і гнучкості конвеєра виявлення, зберігаючи при цьому можливість виводу в реальному часі. Крім того, YOLOv2 навчається на даних виявлення та класифікації одночасно, дозволяючи виявляти понад 9000 категорій об'єктів.

Для поліпшення mAP було використано кілька методів:

1. Пакетна нормалізація (batch normalization) у всіх згорткових шарах. Таке просте регулювання дозволило моделі підвищити точність на кілька відсотків.

2. Змінено вхідний розмір на 416, що призвело до того, що кінцевий згортковий шар мав розмір 13x13. Непарна кількість ознак гарантує, що об'єкт в центрі визначається тільки однією клітинкою. Інтуїція полягає в тому, що більшість зображень містять великий об'єкт приблизно в центрі, тому є причина оптимізувати мережу, щоб краще справлятися з таким сценарієм.

3. Були введені якірні рамки типу RPN. Однак автори провели дослідження кількості, розміру та співвідношення сторін рамок і використали алгоритм кластеризації *k-means* для визначення найкращих параметрів. Вони використали висоту та ширину зображень з датасету, щоб визначити 5 кластерів (5 найпоширеніших співвідношень сторін об'єктів), які присутні в датасеті.

4. Автори застосували обмеження на параметри перетворення обмежувальної рамки, щоб дещо спростити навчання для моделі.

5. Було введено наскрізний шар для перетворення карти ознак з високою роздільною здатністю в карту з нижчою роздільною здатністю без втрати точної просторової інформації.

Ще однією важливою зміною є використання Darknet-19 як базової мережі. Вона має кращу точність (91,2% точності топ-5 на відміну від 90% у VGG-16). Обчислювально Darknet-19 також у 6 разів швидша за VGG-16.

YOLOv2 включає в себе спільне навчання за допомогою єдиної класифікації та виявлення за допомогою ієрархії WordTree за допомогою багатозадачного навчання. Це дозволяє проводити одночасне навчання на наборах даних виявлення (наприклад, COCO) і наборах даних класифікації (наприклад, ImageNet). Якщо мітка об'єкта відсутня при виявленні, модель вчиться передбачати категорії вищого рівня.

Можна виділити наступні обмеження моделі:

- Все ще відносно груба локалізація. Незважаючи на те, що вона покращилася порівняно з YOLOv1, точність відстає від регіональних детекторів, особливо для невеликих об'єктів.
- Обмеження якірних рамок обмежують прогнозування об’єктів що перетинаються. Крім того, YOLOv2 все ще поступається Faster R-CNN у mAP на COCO.
- Складність у навчанні на WordTree. Ця техніка дозволила YOLOv2 набрати кілька відсотків у mAP, але вона вимагає ретельного налаштування для збалансування завдань класифікації та виявлення.

YOLOv3: Поступове покращення

Ґрунтуючись на архітектурах YOLOv1 та YOLOv2, YOLOv3 було розроблено для вирішення кількох ключових питань:

- Удосконалення виявлення малих і середніх об'єктів,
- Підвищення впевненості в класифікації та точності локалізації,
- Підтримка продуктивності в режимі реального часу при підвищенні надійності на складних наборах даних, таких як COCO.

YOLOv3 [7] фокусується на поступових, але важливих покращеннях в конструкції мережі і багатомасштабному прогнозуванні, в результаті чого виходить більш потужний і збалансований одноступеневий детектор.

Ключовими нововведеннями є:

1. Багатомасштабне виявлення. Прогнозування робиться з трьома різними роздільними здатностями карт ознак, що дозволяє виявляти малі, середні та великі об'єкти за допомогою мережі ознакових пірамід (FPN). Вона інтегрує низькорівневі та високорівневі об'єкти між шарами, об'єднуючи збільшені карти ознак з пізніших шарів (які мають нижчу роздільну здатність, але більше семантичної інформації) з більш ранніми шарами (які мають більш детальні ознаки, але не мають семантичної інформації), як показано на рисунку 3, покращуючи виявлення малих об'єктів.

2. Покращена базова мережа: Darknet-53. Складається з 53 згорткових шарів з пакетною нормалізацією та протікаючою ReLU (leaky ReLU). Використовує залишкові зв'язки для покращення навчання. Мережа робиться повністю згортковою шляхом заміни повнозв'язних шарів на шари 1x1 conv та заміни max pool шарів на згорткові шари з кроком (strided conv layers). Згорткові шари з кроком дозволяють зменшити вибірку карт об'єктів і дізнатися, як це робити найкраще.

3. Ця модель також покращує класифікацію та прогнозування обмежувальних рамок. Класифікація проводиться з сигмоїдною функцією замість Softmax. Це дозволяє використовувати набори даних з об'єктами з кількома мітками. Для обмежувальних рамок автори вибрали 9 стандартних прямокутників на комірку, подібно до того, як у Fast R-CNN, а оцінка об'єктності просто обчислюється як 1 або 0. Ці зміни покращують загальну точність складних наборів даних (COCO).

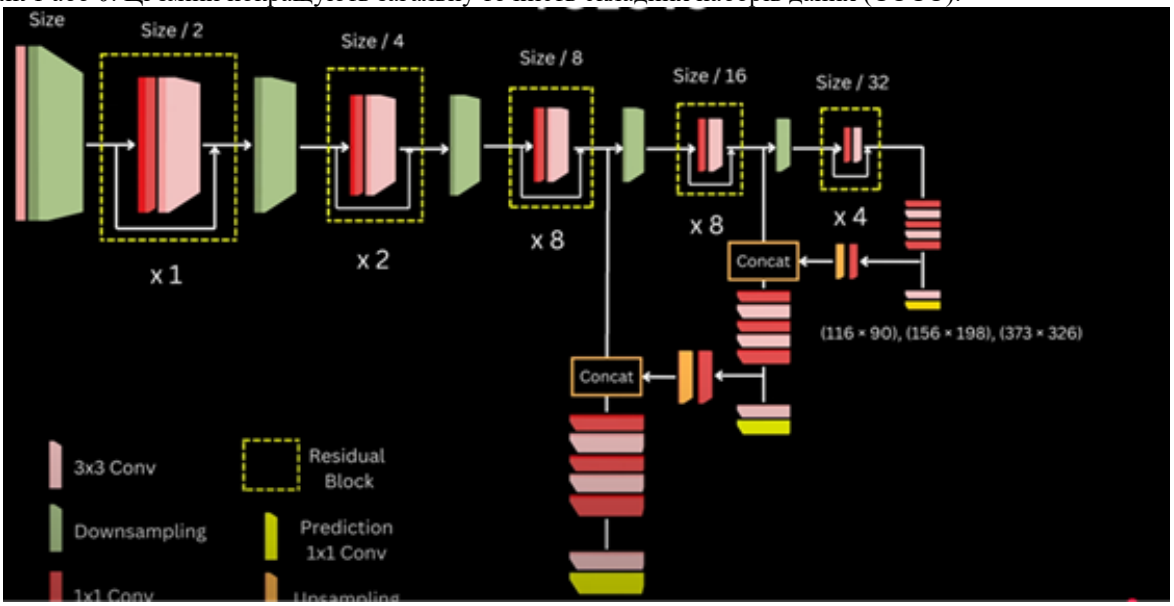

Рис. 3. Архітектура YOLO v3 [22]

YOLOv3 дотримується балансу між швидкістю та точністю, досягаючи роботи у реальному часі (30–45 FPS) із гарною точністю.

Серед обмежень все ще можна відзначити високу похибку локалізації та фіксовані якірні рамки. У YOLOv3 все ще відсутнє точне вирівнювання (наприклад, RoI Align [8]), що впливає на точність на основі IoU. Крім того, вона використовує важку модель (Darknet-53), яка хоч і швидша, ніж ResNet-101, все ще є складною з обчислювальної точки зору для периферійних пристроїв.

YOLOv4: Оптимальна швидкість та точність для розпізнавання об’єктів

YOLO v4 [9] базується на новій базовій мережі — CSPDarknet53, яка уніфікує підходи Cross-Stage Partial connections (CSP) із глибокими нейронними ResNet-подібними мережами, зменшуючи дублювання градієнтних потоків та обчислювальних витрат. Модель удосконалено приєднанням PANet (Path Aggregation Network) в останніх шарах для покращення багаторівневої агрегації ознак та Spatial Pyramid Pooling для розширення рецептивного поля без збільшення кількості обчислень.

Для підвищення загальної стійкості моделі застосовано набір методів «bag of freebies» і «bag of specials».

Bag of Freebies - це методи, що не збільшують витрат на інференс, але покращують збіжність і загальну стійкість під час тренування:

1. Мозаїчна аугментація. Поєднує чотири зображення в одну сітку, створюючи багатший контекст фону й об’єктів, що значно розширює варіативність даних.

2. MixUp. Лінійне змішування двох зображень та їхніх міток, що згладжує розподіл класів і запобігає надмірному пристосуванню.

3. CutMix. Вирізання області з одного зображення та вставка її в інше, з частковим змішуванням міток за площею, що стимулює модель фокусуватися на всіх об’єктах.

4. Self-Adversarial Training (SAT) На першому кроці мережа генерує власні «противні» спотворення зображень, потім навчається правильно їх класифікувати, що підвищує стійкість до шуму.

5. Регуляризація DropBlock. Це регуляризаційна техніка, яка випадково виключає цілі квадратні ділянки в карти ознак під час тренування, подібно до SpatialDropout, але груповими блоками.

6. Згладжування класових міток. Згладжування міток (наприклад, правильний клас отримує 0.9 замість 1.0, решта поділяє 0.1), що знижує надмірну впевненість і покращує узагальнення.

7. Горизонтальне та вертикальне перевернення, HSV-аугментація. Традиційна аугментація геометрією й кольором: випадкові віддзеркалення та зміни в просторі HSV для стійкості до освітлення.

Bag of Specials - це архітектурні рішення та зміни функції втрат, що трохи збільшують обчислювальні витрати під час тренування та/або інференсу, але дають помітний приріст точності:

1. Базова мережа CSPDarknet53 — це перехресно-етапна часткова мережа, що ділить блочні градієнтні шляхи, зменшує дублювання обчислень та пам'ять, зберігаючи глибину та широке рецептивне поле.

2. SPP (Spatial Pyramid Pooling). Пулінг із різними розмірами ядер (1×1, 5×5, 9×9, 13×13) останнього шару базової мережі для розширення рецептивного поля без додаткових згорток.

3. «Шия» PANet (Path Aggregation Network). Зворотне та пряме з'єднання між багаторівневими картами ознак для кращої передачі низькорівневих деталей у вищі шари.

4. SAM (Spatial Attention Module) — це механізм просторової уваги, що виділяє значущі ділянки на карті ознак перед детекцією.

5. Функція втрат через CIoU (Complete IoU) — вдосконалена функція регресії рамок, яка враховує збіг областей, відношення сторін і відстань між центрами, пришвидшуючи збіжність і підвищуючи точність локалізації.

6. Мозаїчний DropBlock (гібрид між gratis та specials), що поєднує мозаїчну аугментацію з DropBlock у картах ознак для додаткової регуляризації.

7. Тренування на вхідних даних різного розміру. Випадковий вибір роздільності (320, 416, 608) під час тренування, що робить модель адаптивною до різних вхідних розмірів без окремого доопрацювання.

8. CIoU-NMS (Complete IoU Non-Maximum Suppression) — версія NMS, що використовує Complete-IoU для вибору кращих кандидатних рамок, підвищуючи якість відсічення накладень.

Архітектура YOLOv4 внесла багато покращень, що дозволило досягти точності в 43.5 mAP на датасеті COCO зберігаючи високу швидкість у 65 кадрів на секунду. Проте архітектура має й недоліки.

По-перше, нововведення, такі як аугментація зображень, Self-Adversarial тренуванням або Complete IoU втрати створюють умови для складного процесу тренування, адже вимагають ретельного налаштування кроків навчання.

По-друге, статичні, хоча і обрані за допомогою алгоритму k-середніх під відповідний датасет, якірні рамки залишаються незмінними протягом навчання моделі. Динамічні якорі можуть надати кращий результат по точності.

По-третє, YOLOv4 все ще не може надати гарної точності для малих об'єктів, особливо у складних сценах.

Також, алгоритм Non-Maximum Suppression (NMS) впливає на ймовірність пропусків об'єктів у щільних сценах.

Ця модель, незважаючи на оптимізації, все ще залишається важкою для малопотужних пристроїв, таких як мобільні телефони або edge-пристрої.

Наступні версії YOLO

YOLOv5

Впровадила автоякорі та еволюційне налаштування гіперпараметрів. За допомогою техніки автоякірних рамок перед початком тренування автоматично генеруються оптимальні якірні рамки за допомогою IoU-орієнтованого алгоритму k-means. За необхідності, до визначених рамок застосовуються еволюційні алгоритми.

Також еволюційні алгоритми застосовуються для пошуку кращих гіперпараметрів, таких як: learning rate, momentum, тощо. На кожній ітерації береться найкраща лінія параметрів із попередніх поколінь і генерує нове покоління параметрів шляхом невеликих мутацій. Це дає можливість знайти оптимальні значення гіперпараметрів під конкретний датасет без ручного перебору. Еволюційні алгоритми та автоякірні рамки зменшуються кількість ручної роботи під час налаштування.

Найбільш значущою зміною в архітектурі моделі є впровадження фокус-шару першим шаром, що конкатенує чотири підпіксельні зсуви зображення у єдини канал, таким чином зменшуючи ширину та висоту карти ознак вдвічі, проте збільшуючи глибину в чотири рази. Це дозволяє мережі одразу отримати багатоканальну детальну інформацію, мінімізуючи втрати через звичайний pooling, і значно прискорює обчислення на ранніх шарах.

Використовуючи Soft NMS із гнучкішими параметрами (IoU-threshold, σ), знижує різкі відсічення рамок із нижчим IoU й піднімає recall у щільних сценах

Також для підтримки пристроїв із обмеженими потужностями модель YOLOv5 пропонує різні масштаби мережі, від v5-n до v5-x, що пропонують різний баланс між розміром мережі та точністю. Найменша v5-n може використовуватись edge-пристроями.

YOLOv6: Одноетапна платформа виявлення об'єктів для промислових застосувань.

Модель було представлено у серпні 2022 року [10] як рішення, що першочергово орієнтоване на промислові застосунки та поєднує високу швидкодію з точністю визначення об'єктів.

Основними відмінностями моделі є розділена голова на дві гілки: регресія координат та класифікація, що дозволило краще оптимізувати позиціювання рамок незалежно від оцінки ймовірності об'єкта.

Багато роботи було проведено для оптимізації роботи задля покращення ефективності на edge-пристроях. Модель використовує базову мережу EfficientRep, яка побудована навколо структурної репараметризації, яка дозволяє використовувати під час тренування складніші багатогілкові блоки зі змішаними ядрами згорток, а на етапі інференсу збирати їх у єдину оптимізовану структуру. Під час тренування кожен EfficientRep-модуль складається з трьох паралельних гілок:

- Згортки 3x3
- Згортки 1x1
- Ідентичне відображення (identity skip)

Під час тренування мережа здобуває переваги різних рецептивних полів (фільтр 1×1 виділяє локальні ознаки, 3×3 дає більшу контекстну інформацію), а також сигнал «ідентичності» для стабільнішого проходження градієнта. Після завершення фази тренування ваги цих трьох паралельних шляхів «згортаються» в одну єдину згортку 3×3. Алгоритм репараметризації обчислює еквівалентні фільтри й зміщення так, щоб результат застосування єдиної згортки 3×3 дорівнював сумі результатів оригінальних гілок.

Ця ідея застосована до всіх основних блоків базової мережі: замість послідовності простих згорток 3×3 використовуються блоки EfficientRep. Така побудова знижує кількість параметрів і FLOPs під час інференсу майже на 20–30% у порівнянні з класичними блоками Darknet чи CSP, але не погіршує якість детекції.

Також у YOLOv6 додано інструменти авто-квантизації до INT8 та INT4 задля зменшення виокристовуваної пам'яті.

YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors.

На відміну від YOLOv6, де архітектурна інновація зосереджувалася головним чином на структурній репараметризації і спрощеній стратегії аугментацій, у YOLOv7 [11] увага перемістилася на більш глибоку оптимізацію як самої мережі, так і послідовності тренування.

Замість базової мережі EfficientRep із паралельними гілками 1×1, 3×3 і skip-зв'язком, YOLOv7 впровадила розширені ефективні мережі агрегації шарів (Extended Efficient Layer Aggregation Networks (E-ELAN)), де розширено кількість шляхів агрегації ознак і збільшено ширину мережі без втрати глибини. Це дозволяє краще збирати контекстні та детальні ознаки на ранніх шарах, ніж YOLOv6. Також згортки 3x3 замінені на RepConv блоки, які багато в чому схожі на блоки з EfficientRep. Це дозволяє покращити як точність розпізнавання так і зменшити обчислювальні витрати.

Для збільшення ефективності розпізнавання дрібних об'єктів модель використовує мережу піраміди ознак агрегування шляхів (Path Aggregation Feature Pyramid Network (PAFPN)) що формує агрегацію ознак із вбудованими ваговими коефіцієнтами. Це дає змогу точніше поєднати деталі з нижчих рівнів й узагальнену інформацію з вищих. Також використовується Dynamic-NMS із використанням AutoAlign для кращого врахування варіацій співвідношення сторін, завдяки чому дрібні об'єкти у тіні великих рідше відсікаються.

Як і в YOLOv6 тут розвивається використання методу SimOTA для порівняння визначених рамок із істинними під час тренування. Замість використання фіксованого порогового значення IoU або наперед прорахованих кластерів якірних рамок використовується динамічна вибірка.— Спочатку для кожної якірної рамки SimOTA передбачає клас та координати обмежувальної рамки. Далі для кожної пари якірної рамки та істинних координат розраховується «ціна», яка включає правильність передбачення класу та точність передбаченої обмежувальної рамки. Коли матриця «цін» складена то для кожної істинної рамки кандидатні якірні рамки сортуються за збільшенням «ціни» і потім відбираються N рамок (або не більше k рамок, або коли кумулятивна «ціна» досягає певного порогового значення).

Оскільки це призначення перераховується на кожній міні-серії, коли базова мережа і голови вдосконалюють свої результати, «найкращі прив'язки» для кожної істинної рамки постійно коригуються.

У тренувальній стратегії слідом за YOLOv6 йде тенденція до спрощення тренування обмежуючи «bag of freebies» до абсолютно необхідних аугментацій. На різних етапах навчання змінюється сила мозаїчної аугментації, параметри MixUp/CutMix коригуються автоматично залежно від втрат і швидкості збіжності. Завдяки цьому мережа швидше адаптується до даних із мінімальною комбінацією аугментацій

YOLOv8.

Модель є найновішою ітерацією сімейства "You Only Look Once", що поєднує спадкоємність інженерних рішень із новими методами оптимізації.

Одна з ключових відмінностей полягає у спрощенні коду та уніфікації архітектурних блоків:

- E-ELAN блоки з репараметризацією (RepConv) та багаторівнева агрегація через PAFPN замінена CSP-Darknet зі спрощеними варіантами згорткових блоків.
- Велика кількість спеціальних блоків замінюється модульними блоками "C2f" (Cross Stage Partial "Fast")

- Тренування відбувається через бібліотеку
  - Налаштування аугментацій - замість ручного налаштування, використовується AutoAugment, яке автоматично обирає оптимальні аугментації під час навчання, враховуючи метрики збіжності;
  - Вибір гіперпараметрів;
  - Процедура експорту відбуваються без необхідності втручання;

Для механізму призначення якірних рамок замість SimOTA кожний батч застосовується динамічний підбір якорів (Dynamic Anchor Matching) де за базове "якірне" узгодження відповідає окрема мережа, що навчається одночасно з мережею визначення. Завдяки цьому дрібні об'єкти локалізуються точніше, а швидкість навчання залишається високою.

Продуктивність на edge-пристроях показує що YOLOv8-nano (найменша конфігурація) вагою близько 6 МБ при запуску на мобільному NPU тактової частоти 1 GHz досягає приблизно 35–45 FPS (в залежності від роздільності). У той самий час YOLOv7-tiny – біля 18 МБ і ≈ 30 FPS, а YOLOv4-tiny вагою понад 20 МБ лише ≈ 20 FPS.

YOLOv9: Learning what you want to learn using programmable gradient information

YOLOv9 [12, 13] запропонувала важливу технологію – програмовану градієнтну інформацію (Programmable Gradient Information (PGI)). Архітектура проекту на малюнку може покращити інтерпретабельність, надійність і універсальність моделі. Дизайн PGI полягає у використанні концепцій оборотної архітектури та багаторівневої інформації для максимізації вихідних даних, які може зберігати модель, та інформації, необхідної для виконання цільових завдань. YOLOv9 розширила E-ELAN до G-ELAN і використала його, щоб показати, як PGI може досягти відмінної точності, стабільності та швидкості логічного висновку на моделях з невеликою кількістю параметрів. Основні особливості YOLOv9 описані нижче:

- Допоміжна реверсивна гілка. PGI використовує властивості оборотної архітектури для вирішення проблеми інформаційних вузьких місць у глибоких нейронних мережах. Це повністю відрізняється від оборотної архітектури загального призначення, яка просто максимізує інформацію, яку потрібно зберегти. PGI використовує інформацію, що зберігається оборотною архітектурою, з основною гілкою у формі допоміжної інформації. За умови збереження інформації, необхідної для цільового завдання, вона зберігає якомога більше інформації з вихідних даних.
- Багаторівнева допоміжна інформація. PGI запропонувала концепцію багаторівневої допоміжної інформації, щоб кожен шар ознак з основної гілки зберігав інформацію, необхідну для всіх цілей завдання, наскільки це можливо. Це допомагає уникнути проблеми, коли попередні методи, як правило, втрачають важливу інформацію на перших шарах мережі, що, у свою чергу, призводить до неможливості отримати достатньо інформації у глибоких шарах.
- Узагальнення для подальших завдань. Оскільки PGI може максимізувати збереження вихідної інформації про дані, моделі, навчені за допомогою PGI, досягають більш надійної продуктивності на невеликих наборах даних, передачі навчання, багатозадачному навчанні та краще адаптують до нових завдань
- Узагальнення для різних архітектур. PGI також можна застосовувати до інших архітектур, таких як звичайні ЗНМ, глибинні ЗНМ, трансформатори, а також різні типи методів комп'ютерного зору, наприклад на основі прив'язки, без прив'язки, без постобробки тощо.

Незважаючи на значні архітектурні вдосконалення та гнучкість, YOLOv9 має кілька помітних обмежень. По-перше, модель усе ще потребує великого обсягу обчислювальних ресурсів під час тренування: програмована градієнтна інформація та механізми адаптивної втрати вимагають додаткових прошарків і метаданих у мережі, що збільшує потребу в пам'яті GPU і подовжує час епох. На слабших пристроях (edge-NPU або мобільних GPU) доводиться відмовлятися від частини функціональності YOLOv9 (наприклад, динамічного регулювання ваг втрат) або втрачати точність унаслідок агресивної квантизації.

По-друге, програмована градієнтна інформація, яка дозволяє моделі фокусуватися на вибраних контрастних прикладах, ускладнює інтерпретацію процесу навчання. Замість традиційних "зрозумілих" функцій втрат (CIoU, focal loss тощо) YOLOv9 вводить множину інтерактивних вагових коефіцієнтів, що автоматично корегуються під час тренування, але вже не дають простого способу зрозуміти, чому модель робить саме такі оновлення. Це ускладнює відтворюваність на інших наборах даних і стає перешкодою для глибокого аналізу помилок (наприклад, чому модель не бачить дуже дрібні або контрастно слабкі об'єкти).

Третя проблема стосується стабільності роботи з дуже нестандартними даними. У YOLOv9 суттєву роль відіграє попереднє "навчання" на "правильних" градієнтах, тому якщо датасет містить аномальні приклади (дуже засвічені кадри, сильні деформації або нетипові ракурси), адаптивний механізм втрат може концентруватися на тих патчах, які аж ніяк не покращують загальну роботу детектору. У таких випадках без дуже ретельного попереднього препроцесінгу даних точність різко падає.

Нарешті, хоча автори заявляють про можливість "програмувати" те, що модель має вивчати, це теж накладає додаткові зобов'язання на розробника: потрібно самостійно вирішувати, які градієнтні сигнали варто посилювати, а які — послаблювати, і як налаштувати їхню динаміку. Без глибокого розуміння внутрішніх механізмів YOLOv9 легко "перетренувати" модель на вузький клас об'єктів і

втратити здатність до узагальнення. Через це для багатьох прикладних задач, де немає команди дослідників зі спеціалізацією в програмованих градієнтах, YOLOv9 може виявитися надто складним інструментом.

YOLOv10: Real-time end-to-end object detection

Загальна архітектура YOLOv10 [13, 14] подібна до YOLOv6 3.0, але із додаванням модулю на основі трансформатора для покращення вилучення глобальних функцій. У моделі замінили подвійну голову на відповідності один-до-багатьох і один-до-одного. Ця зміна дозволяє YOLO обходитися без постобробки, як у методі на основі DETR[15], і може безпосередньо отримувати результати наскрізного виявлення об'єктів. Далі ми представимо деякі відмінні особливості YOLOv10.

Подвійне призначення міток. Використовує метод призначення міток, подібний до DATE [16] і додає операцію зупинки градієнта для гілки один-до-одного.

Виявлення об'єктів без NMS. Конструкція механізму відповідності один-до-одного дає змогу виконувати процес прогнозування не покладаючись на NMS для постобробки.

Розробка блоків, керована рангами. Запропоновано використовувати ранг, щоб визначити, на яких етапах використовується звичайна згортка, а на яких – поглиблена згортка.

Часткова самоувага. YOLOv10 поєднала CSPNet і Transformer і запропонувала модуль самоуваги.

Незважаючи на значні архітектурні вдосконалення, модель усе ще потребує потужного GPU з великою відеопам'яттю для повного конвеєра тренування, що ускладнює його застосування на пристроях з обмеженими ресурсами. Динамічні схеми призначення якірних рамок і програмовані градієнти додають непрозорості: користувачу складніше зрозуміти, які саме зміни у даних сприяють зростанню точності. Крім того, попри архітектурні зміни, які дають високу точність на великих об'єктах, виявлення дрібних чи малоконтрастних об'єктів у складних сценах залишається проблемою, що вимагає додаткових аугментацій або препроцесінгу. У непідходящих умовах використання (наприклад, швидко змінюване відео чи складні погодні умови) точність і швидкодія іноді різко падають. І хоча API зроблено зручним і на готових наборах даних модель налаштовується відносно швидко, проте для власних проєктів усе одно доведеться витратити багато часу на добір аугментацій і моніторинг навчання.YOLOv10 зберігає високий потенціал, але вимагає значних обчислювальних ресурсів для тренування і глибокого розуміння внутрішніх процесів щоб стабільно працювати на реальних, нетипових даних.

YOLOv11

Модель орієнтована на одночасне вирішення кількох задач (детекція, сегментація, оцінка поз та орієнтовані рамки) при мінімальному обчислювальному навантаженні. Ядро YOLOv11 складається з трьох ключових компонент: C3k2 (Cross Stage Partial with kernel size 2) block, SPPF (Spatial Pyramid Pooling - Fast), and C2PSA (Convolutional block with Parallel Spatial Attention).

C3k2 є еволюцією блоку C3 із Cross-Stage Partial (CSP), у якому замінено згортку 3×3 на 2×2. Під час тренування вхідні ознаки розділяються на дві гілки: одна проходить через послідовність згорток 2×2, інша—через пропуск (skip connection), а потім обидві об'єднуються. Завдяки цьому зменшується обчислювальна складність і кількість параметрів, але зберігається здатність моделі виявляти дрібні деталі. На етапі інференсу C3k2-блок повертає лише 2×2-згортку з об'єднаними вагами, що забезпечує високу швидкість без втрат у точності(mAP).

Модуль SPPF (Spatial Pyramid Pooling–Fast) поєднує кілька розмірів пулінгу (1×1, 5×5, 9×9) у каскадному режимі Fast MaxPool, замінивши громіздкі операції стандартного SPP. Це дозволяє суттєво розширити рецептивне поле для великих об'єктів, зменшивши обчислювальні затрати приблизно на 15–20 % порівняно з попередніми версіями, але зберігаючи аналогічну якість локалізації.

C2PSA (Convolutional Block with Parallel Spatial Attention) об'єднує звичайну згортку 3×3 з просторовим блоком уваги (Spatial Attention Map). Спочатку за допомогою глобального середнього пулінгу й згортки 1×1 генерується карта важливості пікселів, яка множиться на вихідні ознаки, після чого застосовується згортка 3×3. Це сприяє фокусуванню мережі на релевантних регіонах і підсилює здатність виявляти дрібні чи малоконтрастні об'єкти без значного збільшення у FLOPs.

У Head-частині YOLOv11 розгалужено декілька шляхів: окремий для регресії координат (включно з орієнтацією рамок), окремий для сегментації й ще один для оцінки поз (keypoint estimation). При цьому кожен рівень масштабованих карт P3–P5 обробляється C3k2-блоками, що відповідають роздільності 1/8, 1/16, 1/32 від оригіналу. Завдяки цьому модель одночасно виявляє дрібні об'єкти і великі контури.

У порівнянні з YOLOv10, яка мала більшу кількість параметрів для схожого mAP, YOLOv11 демонструє суттєве зменшення FLOPs (до 20 % для medium), що важливо для edge-застосунків. Наприклад, версія "nano" із приблизно 3 млн параметрів досягає ≈48 % mAP при ~150 FPS (Tesla V100, "medium" (≈20 млн) — ≈53 % mAP при ~80 FPS, а "extra-large" (≈100 млн) — ≈56,5 % mAP при ~30 FPS.

Порівняння з попередніми версіями вказує на те, що, на відміну від YOLOv10, що фокусувався на детекції рамок і зберігав традиційні C3-блоки та SPP, YOLOv11 уніфікує кілька задач у єдиному конвеєрі та впроваджує нові ефективні блоки, що суттєво знижують параметри та затрати обчислень без втрати точності.

Порівняння моделей

Було виділено такі характеристики моделей та їх роботи для порівняння: точність визначення об'єктів (у mAP) та швидкість роботи (у GFLOPs як абсолютний показник та у кадрах на секунду (FPS) для кращого розуміння застосовності у прикладних сферах) і їх значення наведені у таблиці 1. Ми також наводимо застосовність моделі на пристроях з обмеженими ресурсами (edge-пристроях) для розуміння коли моделі досягли ефективності, достатньої для застосування у сфері IoT.

Потрібно зазначити що точність для старих моделей (R-CNN, Fast/Faster R-CNN, SSD) вказано як mAP@0.5 на наборі PASCAL VOC, для новіших – AP (COCO) при IoU=0.5:0.95, якщо не зазначено інше. Також FPS наведено орієнтовно для розміру 640×640 на сучасному GPU (якщо не уточнено). Реальна швидкість залежить від апаратного забезпечення та реалізації. Також FLOPs – кількість операцій на одне зображення. Для моделей з кількома варіантами вказано порядок величини для найбільшої конфігурації.

Таблиця 1

Характеристики моделей розпізнавання зображень

Модель	Точність (mAP)	FPS	GFLOPs (×10^9)	Параметри (млн)	Підтримка Edge-пристроїв
R-CNN (2014)	≈66% (VOC2007)	<1 FPS	Значно більше 1000	~60	Ні
Fast R-CNN (2015)	≈70% (VOC2007)	~2 FPS (GPU)	~60	~60	Ні
Faster R-CNN (2015)	≈73% (VOC2007)	~5 FPS (GPU)	~80	~50	Ні
SSD300 (2016)	~75% (VOC2007)	~46 FPS (GPU)	~100	~24	Частково (потрібна MobileNet)
YOLOv1 (2016)	63.4% (VOC2007)	~45 FPS (GPU)	~30	45	Частково (є Fast YOLO)
YOLOv2 (2017)	78.6% (VOC2007)	~67 FPS (416px)	~30	~30	Частково бажано на мобільному GPU)
YOLOv3 (2018)	33.0% (COCO)	~30 FPS (416px)	~70	~62	Частково (є Tiny версія)
YOLOv4 (2020)	43.5% (COCO)	~65 FPS (V100)	~50	~64	Частково (є Tiny версія)
YOLOv5 (2020)	50.7% (COCO)	~200 FPS (V100)	~206	86.7	Так (Nano/Small моделі)
YOLOv6 (2022)	43.1% (COCO)	~50+ FPS (T4 GPU)	~44	17.2	Так (Nano модель для edge)
YOLOv7 (2022)	55.9% (COCO)	5–160 FPS (від E6 до Tiny)	~105	36.9	Частково (Tiny версія)
YOLOv8 (2023)	53.9% (COCO)	~280 FPS (A100)	~258	68.2	Так (налаштування. під обмежені ресурси)
YOLOv9 (2024)	~55.6% (COCO)	~60 FPS (T4)	~189	57.3	Так (орієнтована і на використання з обмеженими ресурсами)
YOLOv10 (2024)	≈54% (COCO)	~52 FPS (GPU)	~80 (без NMS)	~20	Так (оптимізована, без NMS)
YOLOv11 (2024)	~54.7% (COCO)	~74 FPS (GPU)	~195	56.9	Так (підтримка Jetson/RPi)

Із таблиці та огляду вище видно трендову еволюцію: від повільних двоетапних підходів (R-CNN сімейство) до швидших одноетапних моделей (SSD, YOLO), а останнім часом – до їх удосконалених версій, що поєднують високу точність рівня SOTA із можливістю роботи в реальному часі. Двоетапні моделі, хоча і забезпечують гарну точність, є занадто вимогливими до ресурсів. Тільки одноетапні моделі здатні забезпечити інференс у режимі реального часу із забезпеченням високої точності (> 40% AP на датасеті COCO), використовуючі потужні GPU, такі як Tesla A100, Tesla T4, Tesla V100, тощо.

Також одноетапні моделі більш придатні для використання на edge-пристроях, таких як Nvidia Jetson чи Raspberry PI 3B+. Це обумовлено меншою кількістю пам'яті, яка використовується одноетапними моделями. Також, останні версії YOLO надають варіативність розмірів (від nano до extra-large), де найменші версії розроблені для використання на edge-пристроях із невеликою втратою у точності.

Висновки

Ми проаналізували найбільш поширені двоетапні та одноетапні моделі нейронних мереж розпізнавання об'єктів від R-CNN до найновіших представників серії YOLO, розглянули ці технології з точки зору сучасних технологій виявлення об'єктів і вказали на ключовий внесок, який вони зробили на кожному етапі.

Одноетапні моделі демонструють найкращу швидкість інференсу та малий розмір, що дозволяє використовувати їх на пристроях з обмеженими ресурсами, в той же час вони є складними для модифікації або внесення додаткових блоків, оскільки це вимагає втручання у структуру нейронних шарів моделі. Однією з особливостей останніх версій YOLO (v9-11) є спрощення конвеєру тренування моделі, що полегшує їх використання кінцевими користувачами. Це також демонструє загальний напрямок розвитку нейронних мереж в напрямку полегшення використання в індустріальних застосунках. Двоетапні моделі є повільнішими, хоча деякі з них надають гарні результати в точності. Також деякі модифікації набагато легше проводити, оскільки є чітка границя між пропозицією регіонів та регресором обмежувальних рамок та класифікатором.

Варто відзначити що подальші кроки можуть бути пов'язані з автоматизацією підбору аугментацій (AutoAugment), інтеграцією часових модулів для відеодетекції й дослідженням вдосконалених механізмів уваги для більшої стійкості у складних умовах (шум, низька освітленість, аномальні кути огляду, слабка контрастність).

Література

1. Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 580–587. https://doi.org/10.1109/CVPR.2014.81
2. Girshick, R. (2015). Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV). https://doi.org/10.1109/ICCV.2015.169
3. S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.
4. W. Liu et al., "SSD: Single Shot MultiBox Detector" in Proc. Eur. Conf. Comput. Vis. (ECCV), 2016, pp. 21–37.
5. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788. [Online]. Available: https://arxiv.org/abs/1506.02640
6. J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 7263–7271. [Online]. Available: https://arxiv.org/abs/1612.08242
7. J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv preprint arXiv:1804.02767, Apr. 2018. [Online]. Available: https://arxiv.org/abs/1804.02767
8. K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2961–2969. doi: 10.1109/ICCV.2017.322.
9. Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934.
10. Li, C., Li, L., Geng, Y., Jiang, H., Cheng, M., Zhang, B., Ke, Z., Xu, X., Chu, X., & Wei, X. (2022). YOLOv6: A single-stage object detection framework for industrial applications (Preprint). arXiv. https://doi.org/10.48550/arXiv.2209.02976
11. Wang, C.-Y., Bochkovskiy, A., Liao, H.-Y. M. (2022). *YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors*. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops.
12. Wang, C.-Y., Yeh, I.-H., & Liao, H.-Y. M. (2024). YOLOv9: Learning what you want to learn using programmable gradient information. In Proceedings of the European Conference on Computer Vision (ECCV) 2024.
13. Wang, C.-Y., & Liao, H.-Y. M. (2024). YOLOv1 to YOLOv10: The fastest and most accurate real-time object detection systems. arXiv. https://doi.org/10.48550/arXiv.2408.09332
14. Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., & Ding, G. (2024). YOLOv10: Real-time end-to-end object detection. arXiv preprint arXiv:2405.14458.
15. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-end object detection with transformers (arXiv:2005.12872). arXiv. https://doi.org/10.48550/arXiv.2005.12872
16. Chen, Y., Chen, Q., Hu, Q., & Cheng, J. (2022). *DATE: Dual assignment for end-to-end fully convolutional object detection* (arXiv:2211.13859). arXiv. https://doi.org/10.48550/arXiv.2211.13859
17. Khanam, R., & Hussain, M. (2024). YOLOv11: An Overview of the Key Architectural Enhancements. arXiv:2410.17725. https://doi.org/10.48550/arXiv.2410.17725

18. Цюцюра С. В. Модифікація класичної нейронної мережі ймовірнісного типу для розпізнавання «ідеального співрозмовника» серед користувачів соціальних мереж / С. В. Цюцюра, І. А.Терейковський, С. В. Палій // Науково-технічний збірник «Управління розвитком складних систем» Київ. нац. ун-ту будівництва і архітектури, 2014, Вип. 19. С. 118-123.

19. Білощицький А. О. Способи пошуку неточних дублікатів зображень в наукових роботах. / А. О. Білощицький, О.В. Діхтяренко // Зб.наук. праць: Управління розвитком складних систем. – К.:КНУБА, 2015. – Вип. 21. – С. 149- 153.

20. Single Shot Multibox Detector | SSD Object Detection Explained and Implemented. YouTube. URL: https://www.youtube.com/watch?v=c_nEue9itwg&t=2489s&ab_channel=ExplainingAI (дата звернення: 06.05.2025).

21. Yaroslav Hozak, Sergiy Paliy. Modern small networks for image classification. feature analysis. Збірник наукових праць "Управління розвитком складних систем". Київ : КНУБА, 2024. №60. с. 221-229. https://doi.org/10.32347/2412-9933.2024.60.221-229

22. YOLOv2 (YOLO9000) and YOLOv3 Explained. YouTube. URL: https://www.youtube.com/watch?v=wYjkiI-Lm-8&t=2658s&ab_channel=ExplainingAI (дата звернення: 06.05.2025).

References

1. Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 580–587. https://doi.org/10.1109/CVPR.2014.81

2. Girshick, R. (2015). Fast R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV). https://doi.org/10.1109/ICCV.2015.169

3. Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6), 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031

4. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. In European Conference on Computer Vision (ECCV) (pp. 21–37). Springer. https://doi.org/10.1007/978-3-319-46448-0_2

5. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 779–788). https://arxiv.org/abs/1506.02640

6. Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (pp. 7263–7271). https://arxiv.org/abs/1612.08242

7. Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. arXiv preprint arXiv:1804.02767. https://arxiv.org/abs/1804.02767

8. He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. In Proceedings of the IEEE International Conference on Computer Vision (ICCV) (pp. 2961–2969). https://doi.org/10.1109/ICCV.2017.322

9. Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). YOLOv4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934. https://arxiv.org/abs/2004.10934

10. Li, C., Li, L., Geng, Y., Jiang, H., Cheng, M., Zhang, B., Ke, Z., Xu, X., Chu, X., & Wei, X. (2022). YOLOv6: A single-stage object detection framework for industrial applications. arXiv. https://doi.org/10.48550/arXiv.2209.02976

11. Wang, C.-Y., Bochkovskiy, A., & Liao, H.-Y. M. (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops. https://arxiv.org/abs/2207.02696

12. Wang, C.-Y., Yeh, I.-H., & Liao, H.-Y. M. (2024). YOLOv9: Learning what you want to learn using programmable gradient information. In Proceedings of the European Conference on Computer Vision (ECCV) 2024. https://doi.org/10.1007/978-3-031-72609-8_27

13. Wang, C.-Y., & Liao, H.-Y. M. (2024). YOLOv1 to YOLOv10: The fastest and most accurate real-time object detection systems. arXiv. https://doi.org/10.48550/arXiv.2408.09332

14. Wang, A., Chen, H., Liu, L., Chen, K., Lin, Z., Han, J., & Ding, G. (2024). YOLOv10: Real-time end-to-end object detection. arXiv preprint arXiv:2405.14458. https://doi.org/10.48550/arXiv.2405.14458

15. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-end object detection with transformers. arXiv. https://doi.org/10.48550/arXiv.2005.12872

16. Chen, Y., Chen, Q., Hu, Q., & Cheng, J. (2022). DATE: Dual assignment for end-to-end fully convolutional object detection. arXiv. https://doi.org/10.48550/arXiv.2211.13859

17. Khanam, R., & Hussain, M. (2024). YOLOv11: An overview of the key architectural enhancements. arXiv. https://doi.org/10.48550/arXiv.2410.17725

18. Tsiutsiura, S. V., Tereikovskyi, I. A., & Palii, S. V. (2014). Modification of a classical probabilistic-type neural network for recognizing the “ideal interlocutor” among social network users. Management of the Development of Complex Systems, (19), 118–123.

19. Biloshytskyi, A. O., & Dikhtiarenko, O. V. (2015). Methods of searching for approximate image duplicates in scientific works. Management of the Development of Complex Systems, (21), 149–153.

20. Explaining AI. (2025, May 6). Single Shot Multibox Detector | SSD object detection explained and implemented [Video]. YouTube. https://www.youtube.com/watch?v=c_nEue9itwg&t=2489s&ab_channel=ExplainingAI

21. Hozak, Y., & Paliy, S. (2024). Modern small networks for image classification: Feature analysis. Management of the Development of Complex Systems, (60), 221–229. https://doi.org/10.32347/2412-9933.2024.60.221-229

22. Explaining AI. (2025, May 6). YOLOv2 (YOLO9000) and YOLOv3 explained [Video]. YouTube. https://www.youtube.com/watch?v=wYjkiI-Lm-8&t=2658s&ab_channel=ExplainingAI