Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Детекція множин об'єктів на зображеннях з використанням YOLO
(тема)

Виконав:
студент 4 курсу, групи ІТІНФ-20-2
Кізіма Я.А.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник проф. Машталір В.П.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Кізімі Ярославу Андрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Детекція множин об'єктів на зображеннях з використанням YOLO

затверджена наказом університету від 20 травня 2024 року № 464 Ст

2. Термін подання студентом роботи до екзаменаційної комісії 3 червня 2024 р.

3. Вихідні дані до роботи науково-методична та науково-технічна література, матеріали конференцій, дані інтернет-мережі, моделі YOLO (You Only Look Once), датасети для навчання та тестування, метрики Mean Average Precision (mAP), Recall, Precision.

4. Перелік питань, що потрібно опрацювати в роботі

1. Огляд основних методів детекції множин об'єктів на зображеннях з використанням YOLO.
2. Математичні моделі YOLO.
3. Комп'ютерна модель YOLO.
4. Адаптація моделі YOLO для специфічних галузевих застосувань.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Актуальність проблеми детекції зображень, постановка задачі, тестові зображення, діаграми програмних застосунків.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	08.04.2024	
2	Аналіз завдання, підбір літератури	08.04.24-14.04.24	
3	Аналіз літератури з досліджуваної проблеми	16.04.24-18.04.24	
4	Аналіз технічних засобів	19.04.24-25.04.24	
5	Розробка алгоритмів	26.04.24-14.05.24	
6	Програмна реалізація	16.05.24-23.05.24	
7	Оформлення пояснювальної записки	24.05.24-30.05.24	
8	Перевірка на плагіат	01.06.24	
9	Рецензування	01.06.24	
10	Підготовка презентації та доповіді	29.05.24-08.06.24	
11	Занесення роботи в електронний архів	12.06.24	
12	Попередній захист кваліфікаційної роботи	12.06.24	

Дата видачі завдання 8 квітня 2024 р.

Студент _____
(підпис)

Керівник роботи _____ _____проф. Машталір В.П._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 70 с., 2 табл., 25 рис., 30 джерел.

YOLO, YOU ONLY LOOK ONCE, ДЕТЕКЦІЯ МНОЖИН, РОБОТА З ЗОБРАЖЕННЯМИ, ДЕТЕКЦІЯ ОБ’ЄКТІВ, ОЦІНКА ІНФОРМАЦІЇ.

Об'єктом роботи є алгоритм YOLO (You Only Look Once) для виявлення об'єктів на зображеннях.

Метою роботи є дослідження та аналіз принципів роботи алгоритму YOLO, його ефективності та можливостей у виявленні об'єктів на зображеннях. Також в рамках роботи планується розгляд впливу параметрів та конфігурацій алгоритму на його результативність. Крім того, метою є вивчення можливостей застосування YOLO у різних сферах, таких як комп'ютерно зорова навігація роботів, відеоспостереження, автоматичне водіння автомобілів та медична діагностика. На основі отриманих даних планується розробка та реалізація власного застосування YOLO з врахуванням конкретних вимог та обмежень вибраної сфери застосування.

YOLO, YOU ONLY LOOK ONCE, DETECTION OF SETS, WORKING WITH IMAGES, DETECTION OF OBJECTS, EVALUATION OF INFORMATION.

The object of the work is the YOLO (You Only Look Once) algorithm for detecting objects in images.

The purpose of the work is research and analysis of the principles of the YOLO algorithm, its effectiveness and capabilities in detecting objects in images. Also, as part of the work, it is planned to consider the influence of the parameters and configurations of the algorithm on its effectiveness. In addition, the goal is to study the possibilities of YOLO's application in various fields, such as computer vision navigation of robots, video surveillance, automatic driving of cars and medical diagnostics. Based on the received data, it is planned to develop and implement YOLO's own application, taking into account the specific requirements and limitations of the selected field of application.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

YOLO - You Only Look Once

NMS - Non-maximum Suppression

IoU - Intersection over Union

COCO - Common Objects in Context

PAN - Path Aggregation Network

RepVL-PAN - Vision-Language Path Aggregation Network

Leaky ReLU - Rectified Linear Unit

VGG - Visual Geometry Group

WRC - Weighted-Residual-Connections

CSP - Cross-Stage-Partial-connections

BoF - Bag of Freebies

SPP - Spatial Pyramid Pooling

PANet - Path Aggregation Network

PSA - Partial Self-Attention

ВСТУП

YOLO є абревіатурою від "You Only Look Once", що перекладається як "Ти дивишся лише один раз". Це популярний алгоритм для виявлення об'єктів на зображеннях та відео, який відрізняється високою швидкістю роботи та точністю результатів. Основна ідея YOLO полягає в тому, щоб виявляти об'єкти шляхом одноразового аналізу всього зображення, відтак значно прискорюючи процес порівняно з іншими алгоритмами.

Алгоритм YOLO широко використовується у різних областях, включаючи комп'ютерне зорове навігацію роботів, відеоспостереження, автоматичне водіння автомобілів, медичну діагностику та багато іншого. Його висока швидкість та точність робить його привабливим вибором для реалізації систем, які вимагають оперативного виявлення та відстеження об'єктів у реальному часі.

Основний принцип роботи YOLO полягає в тому, щоб аналізувати зображення лише один раз, відрізняючи його від більшості традиційних методів, які вимагають багаторазового сканування. Це досягається за рахунок використання нейронних мереж із зміщеними сітками (grid), які натреновані на виявлення об'єктів у реальному часі. Кожна клітина сітки прогнозує ймовірність присутності певного класу об'єктів та координати обмежувального прямокутника.

1 ОГЛЯД ОСНОВНИХ МЕТОДІВ ДЕТЕКЦІЇ МНОЖИН ОБ’ЄКТІВ НА ЗОБРАЖЕННЯХ З ВИКОРИСТАННЯМ YOLO

1.1 Згорткові мережі

Почнемо з Convolutional Neural Networks (CNNs) або згорткові нейронні мережі є однією з найважливіших архітектур у галузі глибокого навчання, спеціалізованих на обробці та аналізі зображень. Завдяки своїй здатності автоматично виділяти важливі ознаки зображень, CNNs досягли значних успіхів у різних задачах комп'ютерного зору, таких як класифікація зображень, виявлення об'єктів, сегментація зображень та інші.

Основні компоненти CNNs:

- згорткові шари (Convolutional Layers);
- шари активації (Activation Layers);
- шари субдискретизації (Pooling Layers);
- повнозв'язні шари (Fully Connected Layers);
- нормалізація та регуляризація.

Згорткові шари (Convolutional Layers) - основним будівельним блоком CNN є згортковий шар. Цей шар застосовує згортковий оператор до вхідного зображення, виділяючи локальні патерни та ознаки. Згортковий оператор або ядро (фільтр) рухається по всьому зображенню, обчислюючи скалярний добуток між ядром та відповідними частинами зображення. Вихід згорткового шару називається "картою ознак" (feature map), яка відображає присутність певних ознак у різних частинах вхідного зображення.

Шари активації (Activation Layers) - після кожного згорткового шару зазвичай йде шар активації, який додає нелінійність до моделі. Найпоширеніша функція активації — це ReLU (Rectified Linear Unit), Інші функції активації включають Sigmoid та Tanh.

Шари субдискретизації (Pooling Layers) - субдискретизація або пулинг (наприклад, max pooling або average pooling) зменшує розміри карти ознак, зберігаючи найважливіші інформації. Це допомагає зменшити кількість параметрів та обчислювальні витрати, а також запобігає перенавчанню. Max pooling вибирає максимальне значення з кожного підвікна, тоді як average pooling обчислює середнє значення.

Повнозв'язні шари (Fully Connected Layers) - наприкінці архітектури CNN зазвичай знаходяться один або кілька повнозв'язних шарів. Ці шари використовуються для об'єднання виділених ознак та прийняття остаточних рішень, таких як класифікація. Вихід з повнозв'язного шару — це вектор, який представляє ймовірності приналежності до різних класів.

Нормалізація та регуляризація. - нормалізація, як-от Batch Normalization, допомагає прискорити процес навчання та підвищити стабільність моделі. Регуляризація, як-от Dropout, запобігає перенавчанню, відключаючи випадкові нейрони під час навчання.

Архітектура CNN включає кілька згорткових і пулинг шарів, за якими слідують один або кілька повнозв'язних шарів. Типова архітектура може виглядати так:

- вхідне зображення;
- згортковий шар + ReLU;
- пулинг шар;
- повторення кроків 2-3 кілька разів;
- повнозв'язний шар;
- вихідний шар з функцією активації, яка відповідає завданню.

CNN широко використовуються в різних додатках, таких як:

- класифікація зображень;
- виявлення об'єктів;
- сегментація зображень (розділення зображення на смислові частини);
- розпізнавання обличчя;

– автоматичне керування транспортними засобами.

Популярні архітектури CNN для розпізнавання об'єктів:

– R-CNN;

– fast R-CNN;

– faster R-CNN;

– YOLO;

– SSD;

– Feature Pyramid Networks (FPN);

– RetinaNet.

R-CNN - можна сказати перша модель для вирішення даної задачі. Працює як звичайний класифікатор зображень. На вхід мережі подаються різні регіони зображення і для них робиться передбачення. Дуже повільна оскільки проганяє одне зображення декілька тисяч разів.

Fast R-CNN - покращена і швидша версія R-CNN, працює за схожим принципом, але спочатку всі зображення подаються на вхід CNN, потім з отриманого внутрішнього подання генеруються регіони. Але все ще досить повільна для задач реального часу.

Faster R-CNN - головна відмінність від попередніх у тому, що замість selective search алгоритму для вибору регіонів використовує нейронну мережу для їх навчання.

YOLO - зовсім інший принцип роботи порівняно з попередніми, не використовує регіони взагалі.

SSD - за принципом схожа на YOLO, але в якості мережі для вилучення ознак використовує VGG16. Теж доволі швидка і придатна для роботи в реальному часі.

Feature Pyramid Networks (FPN) - ще один різновид мережі типу Single Shot Detector, через особливості вилучення ознак краще ніж SSD розпізнає дрібні об'єкти.

RetinaNet - використовує комбінацію FPN+ResNet і завдяки спеціальній функції помилки (focal loss) дає вищу точність (ассигасу).

YOLO або Look You Only Once — це дуже популярна на поточний момент архітектура CNN, яка використовується для розпізнавання множини об'єктів на зображенні. Більш повну інформацію про неї можна отримати на офіційному сайті, там же можна знайти публікації в яких докладно описана теорія і математична складова цієї мережі, а так само розписаний процес її навчання. Головна особливість цієї архітектури порівняно з іншими полягає в тому, що більшість систем застосовують CNN кілька разів до різних регіонів зображення, YOLO CNN застосовується один раз до всього зображення відразу. Мережа ділить зображення на своєрідну сітку і пророкує bounding boxes і ймовірності того, що там є шуканий об'єкт для кожної ділянки.

Плюси даного підходу полягає в тому, що мережа дивиться на все зображення відразу і враховує контекст при детектуванні і розпізнавання об'єкта. Так само YOLO в 1000 разів швидше, ніж R-CNN і близько 100х швидше ніж Fast R-CNN.

YOLOv3 — це вдосконалена версія архітектури YOLO. Вона складається з 106-ти згорткових шарів і краще детектує невеликі об'єкти порівняно з її попереднецею YOLOv2. Основна особливість YOLOv3 полягає в тому, що на виході є три шари кожен з яких розрахований на виявлення об'єктів різного розміру. Наведено її схематичне влаштування на рисунку 1.1:

На рисунку також можна побачити, що YOLOv3 використовує методи прив’язки якорів (anchor boxes) для прогнозування розмірів об'єктів. В кожному з трьох вихідних шарів присутні кілька таких якірних блоків, що дозволяє моделі одночасно прогнозувати різні розміри об'єктів в межах однієї області. Загалом, удосконалення в YOLOv3 роблять її значно потужнішою та гнучкішою порівняно з попередніми версіями. На додаток до використання якорних блоків, YOLOv3 також використовує глибоку архітектуру мережі Darknet-53, яка складається з 53 згорткових шарів. Ця архітектура дозволяє моделі краще витягати ознаки зображень, що призводить до більш точного виявлення об'єктів. Ключовою особливістю YOLOv3 є те, що вона використовує концепцію багатомасштабного прогнозування.

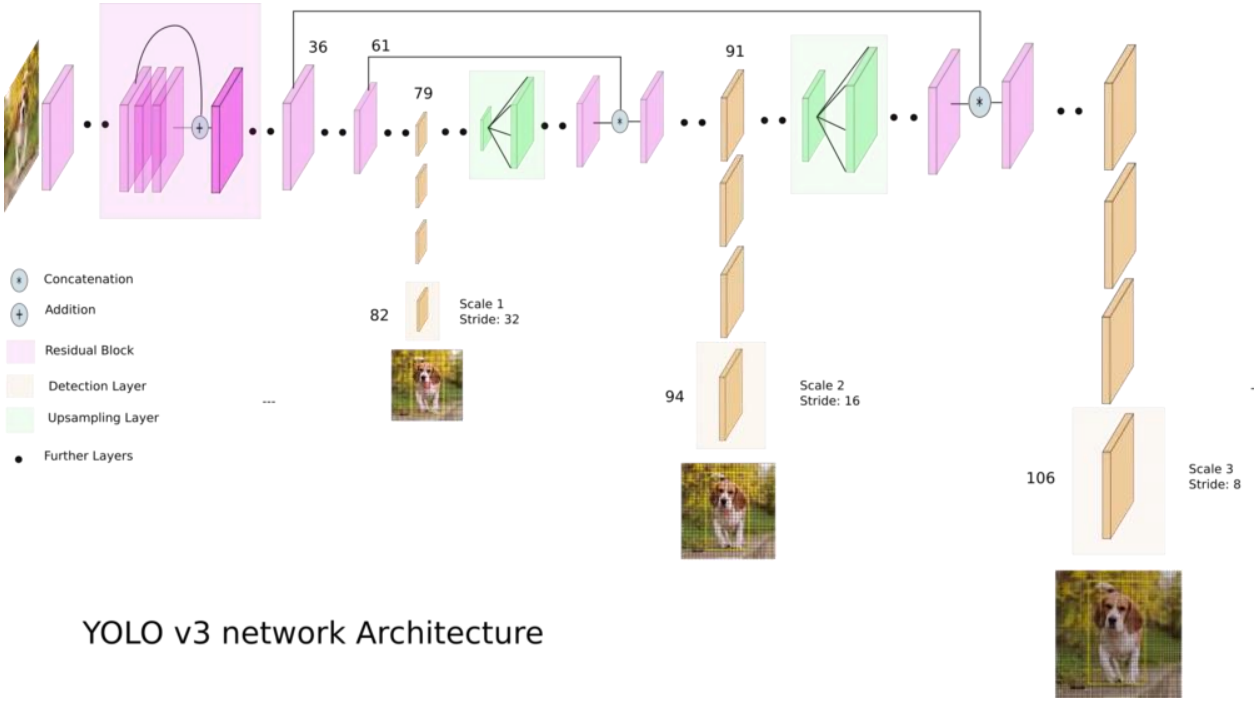

Рисунок 1.1 – Шари виявлення об’єктів різного розміру

YOLOv3-tiny — обрізана версія архітектури YOLOv3, складається з меншої кількості шарів (вихідних шарів всього 2). Вона гірше пророкує дрібні об'єкти і призначена для невеликих датасетів. Але, через спрощену будову, мережа займає невеликий обсяг пам'яті (~35 Мб) і вона видає вищий FPS. Тому така архітектура краще для використання на мобільному пристрої.

1.2 Перший метод Сітка (Grid)

Сітка (Grid) - це одна з ключових концепцій алгоритму YOLO. Вона використовується для розділення вхідного зображення на рівні області, що дозволяє алгоритму ефективно виявляти об'єкти на зображенні.

Сітка складається з фіксованої кількості клітин, розміщених в рядках і стовпцях. Кількість клітин у сітці зазвичай визначається під час попередньої настройки моделі YOLO і може бути налаштована в залежності від потреб застосування. Ці клітини використовуються для поділу зображення на частини, в яких модель YOLO виконує детекцію об'єктів. Кожна клітина сітки

відповідає певній області зображення і прогнозує об'єкти, які можуть знаходитися в цій області. Для кожної клітини сітки YOLO виводить набір прогнозів, який містить інформацію про ймовірності присутності різних класів об'єктів та координати обмежувального прямокутника, що містить ці об'єкти. Важливою перевагою використання сітки є можливість ефективно працювати з областями зображення різного розміру та пропорцій, а також виявляти об'єкти на різних частинах зображення. Крім того, сітка дозволяє алгоритму виявляти об'єкти різних розмірів та форм, забезпечуючи універсальність та адаптивність моделі до різних умов та сценаріїв.

Загалом, використання сітки у моделі YOLO дозволяє ефективно та точно виявляти об'єкти на зображеннях, забезпечуючи високу швидкість та надійність результатів.

Основні ідеї концепції сітки в YOLO:

- розділення зображення на сітку;
- відповідальність комірок;
- передбачення обмежувальних рамок (bounding boxes);
- прогнозування класів.

Розділення зображення на сітку - зображення поділяється на сітку рівних комірок. Кожна комірка відповідає за детекцію об'єктів, що знаходяться в межах цієї області. Зазвичай, розмір сітки може бути, наприклад, 7x7, 13x13 або 19x19 залежно від розміру вихідного зображення та вимог до точності.

Відповідальність комірок - кожна комірка відповідає за передбачення об'єктів, центри яких знаходяться всередині цієї комірки. Якщо об'єкт перекриває межі кількох комірок, відповідальною вважається та комірка, у якій знаходиться центр об'єкта.

Передбачення обмежувальних рамок (bounding boxes) - кожна комірка прогнозує кілька обмежувальних рамок (bounding boxes) та їхні відповідні довірчі значення (confidence scores). Довірче значення вказує на ймовірність того, що в цій рамці знаходиться об'єкт, а також на точність передбачених координат рамки.

Прогнозування класів - кожна комірка також прогнозує ймовірності класів для об'єктів, що можуть бути присутніми у відповідних обмежувальних рамках. Ці ймовірності використовуються для визначення, до якого класу належить передбачений об'єкт (наприклад, автомобіль, пішохід, велосипед тощо).

Процес детекції проходить згідно наступних етапів:

– вхідне зображення;
– прогнозування рамок і класів;
– підбірка та фільтрація.

Вхідне зображення - зображення подається на вхід моделі YOLO, яка його обробляє та розділяє на сітку комірок.

Прогнозування рамок і класів - для кожної комірки модель прогнозує кілька обмежувальних рамок, їхні довірчі значення та ймовірності класів об'єктів.

Підвибірка та фільтрація - після прогнозування, застосовується процедура фільтрації для видалення дубльованих рамок і залишення лише тих, що мають найвищі довірчі значення [11]. Цей процес називається Non-Maximum Suppression (NMS). Обираються найбільш ймовірні об'єкти з урахуванням передбачених класів і довірчих значень.

Переваги підходу сітки:

– одночасна детекція;
– швидкість;
– простота архітектури.

Одночасна детекція - завдяки сітці модель може одночасно передбачати кілька об'єктів у різних частинах зображення.

Швидкість - використання сітки дозволяє моделі працювати швидко, що є важливим для додатків у реальному часі, таких як відеоспостереження або автономне водіння.

Простота архітектури - концепція сітки спрощує архітектуру моделі, оскільки кожна комірка має чітку відповідальність за детекцію об'єктів у своїй області.

1.3 Прогнозування класу та локалізація

Прогнозування класу та локалізація є ключовими етапами алгоритму YOLO для виявлення об'єктів на зображеннях. Ці етапи визначають, як алгоритм ідентифікує та локалізує об'єкти різних класів на зображенні.

Прогнозування класу це на кожній клітині сітки алгоритм YOLO виробляє прогноз щодо наявності різних класів об'єктів. Для кожного класу створюється відповідна кількість вихідних каналів, які визначають ймовірність того, що даний об'єкт належить до кожного з класів. Наприклад, якщо на зображенні присутній автомобіль, собака та велосипед, то алгоритм згенерує відповідний вихідний вектор ймовірностей для кожного з цих класів.

Локалізація об'єктів це після прогнозування класів для кожної клітини сітки, YOLO визначає прямокутний прямокутник (bounding box), який точно обмежує кожен об'єкт на зображенні. Цей прямокутник задається координатами (x, y) верхнього лівого кута прямокутника та його шириною та висотою. Для кожної клітини сітки виходить декілька таких прямокутників, кожен з яких пов'язаний з певним класом.

Після цих двох етапів алгоритм YOLO застосовує фільтрацію Non-maximum Suppression (NMS), щоб вибрати найбільш ймовірні прямокутники для кожного класу та видалити непотрібні дублікати та перекриття. Прогнозування класу та локалізація є важливими складовими частинами YOLO, які дозволяють алгоритму точно та ефективно визначати та локалізувати об'єкти на зображенні. Таким чином, комбінуючи ці підходи, YOLO забезпечує високоефективне детектування об'єктів у реальному часі.

1.4 Підвибірка та фільтрація

Підвибірка та фільтрація (Non-maximum Suppression NMS) – це етап, що відбувається після прогнозування класів та локалізації об’єктів у алгоритмі YOLO. Цей етап спрямований на вибір найбільш ймовірних об’єктів для кожного класу та усунення непотрібних дублікатів та перекриття.

Процес підвибірки та фільтрації можна уявити як наступні кроки:

- одночасна детекція;
- сортування прогнозів;
- видалення непотрібних дублікатів;
- повторення процесу.

Сортування прогнозів - спочатку всі прогнози класів об'єктів сортуються за ймовірністю їх присутності. Об'єкти з більш високою ймовірністю перебувають на більш вищих позиціях у списку.

Видалення непотрібних дублікатів - починаючи з об'єкта з найвищою ймовірністю, видаляються всі інші об'єкти, які мають перекриття з ним на певному рівні перекриття (наприклад, визначеному за допомогою значення перекриття перекриття IoU - Intersection over Union). Це дозволяє вибрати лише найбільш точні та не перекриті об'єкти.

Повторення процесу- після видалення об'єктів з найвищою ймовірністю, процес повторюється для наступних об'єктів з найвищою ймовірністю. Це виконується до тих пір, поки всі об'єкти не будуть оцінені.

Фінальний результат - на виході отримується список об'єктів з найвищою ймовірністю, які не перекриваються одне з одним.

Процес підвибірки та фільтрації допомагає зменшити кількість непотрібних дублікатів та перекриття об'єктів, що забезпечує більш точні та зрозумілі результати виявлення об'єктів на зображеннях. Цей етап важливий для покращення якості виявлення об'єктів та для подальшого використання результатів у різних додатках і системах.

1.5 Функція втрат

Функція втрат (loss function) в алгоритмі YOLO грає ключову роль у процесі навчання моделі. Вона визначає, наскільки добре модель працює під час навчання, порівнюючи прогнози моделі з правильними мітками (ground truth) об'єктів на зображенні.

Втрати локалізації оцінюють точність передбачених координат обмежувальних рамок (bounding boxes). Ця частина функції втрат враховує різницю між передбаченими координатами центрів рамок, їх шириною та висотою і відповідними істинними значеннями.

Основна мета локалізаційних втрат – забезпечити, щоб передбачені рамки максимально точно відповідали реальним положенням і розмірам об'єктів у зображенні. Якщо модель неправильно передбачає положення або розмір об'єкта, то ці втрати зростають, що змушує модель покращувати свої передбачення під час навчання.

Втрати впевненості (Confidence Loss) - оцінюють впевненість моделі в тому, що в передбаченій рамці дійсно знаходиться об'єкт. Ця частина функції втрат включає дві складові:

- втрати для боксів, що містять об'єкти;
- втрати для боксів, що не містять об'єктів.

Втрати для боксів, що містять об'єкти - якщо в боксі знаходиться об'єкт, модель повинна передбачити високу впевненість. Втрати нараховуються, якщо модель не досить впевнена в наявності об'єкта у цьому боксі.

Втрати для боксів, що не містять об'єктів - якщо в боксі немає об'єкта, модель повинна передбачити низьку впевненість. Втрати нараховуються, якщо модель помилково передбачає високу впевненість у цьому боксі. Цей підхід допомагає моделі навчитися правильно визначати, де дійсно є об'єкти, і уникати помилкових передбачень.

Втрати класифікації (Classification Loss) оцінюють, наскільки точно модель передбачає клас об'єкта в кожному боксі. Ця частина функції втрат

враховує різницю між передбаченими ймовірностями класів об'єктів і фактичними класами.

Метою втрат класифікації є забезпечення того, щоб модель правильно ідентифікувала типи об'єктів у зображенні. Наприклад, якщо модель бачить автомобіль, вона повинна впевнено класифікувати його як автомобіль, а не як інший об'єкт, такий як автобус або мотоцикл.

Основна мета локалізаційних втрат – забезпечити, щоб передбачені рамки максимально точно відповідали реальним положенням і розмірам об'єктів у зображенні [12]. Якщо модель неправильно передбачає положення або розмір об'єкта, то ці втрати зростають, що змушує модель покращувати свої передбачення під час навчання.

Основні компоненти функції втрат в YOLO включають:

– точність класифікації;
– точність локалізації;
– вага об'єкта;
– необ'єктні прямокутники.

Точність класифікації (Classification Accuracy) - цей компонент вимірює, наскільки точно модель класифікує об'єкти. Він порівнює прогнозовані ймовірності класів з правильними мітками класів об'єктів на зображенні. Цей компонент функції втрат допомагає моделі відрегулювати ваги, щоб підвищити точність класифікації.

Точність локалізації (Localization Accuracy) - цей компонент вимірює, наскільки точно модель локалізує об'єкти на зображенні. Він порівнює прогнозовані координати прямокутників з правильними координатами об'єктів. Цей компонент допомагає моделі коректно розміщувати прямокутники вокруг виявлених об'єктів.

Вага об'єкта (Objectness Weight) - цей компонент враховує наявність об'єкта в клітці сітки. Якщо клітина містить об'єкт, цей компонент враховується в розрахунку функції втрат для відповідної клітини. Це

допомагає моделі зосередитися на виявленні об'єктів та зменшити вплив фонових частин зображення.

Необ'єктні прямокутники (Background Boxes) - цей компонент враховує вагу фонових прямокутників, тобто тих, які не містять об'єктів. Це допомагає моделі зосередитися на виявленні об'єктів та ігнорувати фонові елементи.

Втрати локалізації - використання квадратів різниць для координат та квадратних коренів для розмірів боксів забезпечує, що втрати мають меншу вагу для великих боксів і більшу вагу для малих боксів, що допомагає покращити стабільність навчання.

Втрати класифікації - квадрати різниць між істинними та передбаченими ймовірностями класів штрафують модель за неправильну класифікацію об'єктів.

Втрати впевненості: цей компонент відповідає за різницю між передбаченим значенням впевненості (probability) того, що рамка дійсно містить об'єкт, і фактичним станом. Впевненість визначає ймовірність того, що вказаний об'єкт дійсно присутній у межах рамки.

Функція втрат YOLO забезпечує ефективне навчання моделі, орієнтуючись на точність локалізації, надійність передбачень та правильну класифікацію об'єктів. Це дозволяє YOLO досягати високих показників продуктивності та точності у задачах детекції об'єктів.

Загальна функція втрат в алгоритмі YOLO об'єднує всі ці компоненти та використовується під час процесу зворотнього поширення помилки (backpropagation) для корекції параметрів моделі, щоб зменшити втрати та підвищити її ефективність. Використання правильно налаштованої функції втрат є ключовим для успішного навчання моделі YOLO та досягнення високої точності виявлення об'єктів на зображеннях.

Об'єднання цих компонентів у загальну функцію втрат дозволяє моделі одночасно навчатися точно визначати місця розташування, розміри та типи об'єктів на зображеннях, а також мінімізувати кількість хибнопозитивних результатів.

2 МАТЕМАТИЧНІ МОДЕЛІ YOLO

Основна ідея YOLO полягає у поділі зображення на сітку і передбаченні об'єктів та їх координат для кожної клітини цієї сітки. Математичні моделі, що лежать в основі YOLO, включають згорткові операції, функції активації, функцію втрат та інші важливі компоненти.

2.1 Основні компоненти математичних моделей YOLO

Поділ зображення на сітку - вхідне зображення розділяється на сітку розміром $S \times S$. Наприклад, для зображення 448x448 пікселів з сіткою 7x7, кожна клітина сітки буде мати розмір 64x64 пікселів.

Прогнозування для кожної клітини - для кожної клітини сітки модель передбачає B обмежувальних прямокутників (bounding boxes) і відповідні їм ймовірності приналежності до певних класів. Кожен прямокутник представлений п'ятьма параметрами.

Вихідні значення для кожної клітини сітки модель видає вектор розміром. Функція втрат в YOLO враховує як точність класифікації, так і точність локалізації об'єктів. Вона складається з декількох компонентів:

- втрата координат (Localization loss);
- втрата ймовірності об'єкта (Confidence loss);
- втрата класифікації (Classification loss).

Крім того, для покращення продуктивності та точності, YOLO використовує різні техніки попередньої обробки та постобробки даних. Наприклад, під час попередньої обробки зображення масштабується та нормалізується, щоб відповідати вимогам вхідного шару нейронної мережі. Постобробка включає такі етапи, як придушення непотрібних рамок (Non-Maximum Suppression, NMS), що дозволяє видаляти дублікати передбачених

меж об'єктів та залишати лише найбільш достовірні передбачення. Це суттєво знижує кількість хибно позитивних спрацьовувань та підвищує загальну точність системи.

Також важливо зазначити, що в останніх версіях YOLO були впроваджені покращення архітектури, такі як використання більш глибоких і складних моделей, які дозволяють досягати ще більшої точності та швидкості. Завдяки цим удосконаленням, YOLO залишається однією з провідних технологій для вирішення задач комп'ютерного зору в режимі реального часу.

2.2 Типи завдань детекції

Традиційний object detection. При такому підході модель здатна детектувати на зображеннях заздалегідь заданий список об'єктів, на якому вона навчалася (рис. 2.1). Модель може бути спеціально налаштована для виявлення об'єктів в конкретних умовах або сценаріях, що підвищує її ефективність у певних областях застосування, таких як медична діагностика або автоматизація промислових процесів. Для вирішення цих обмежень було розроблено більш сучасні підходи до детекції об'єктів, такі як алгоритми на основі глибокого навчання, зокрема методи, засновані на нейронних мережах, як-от YOLO (You Only Look Once), SSD (Single Shot MultiBox Detector) і Faster R-CNN. Ці методи можуть одночасно обробляти велику кількість об'єктів та адаптуватися до нових типів об'єктів і умов. Вони забезпечують більшу гнучкість і продуктивність, що робить їх придатними для широкого спектра застосувань, включаючи автономне водіння, відеоспостереження, та інші системи штучного інтелекту. Зокрема, у сфері автономного водіння YOLOv3 дозволяє транспортним засобам в реальному часі розпізнавати й уникати перешкод, розпізнавати дорожні знаки та світлофори, а також відстежувати рух інших учасників дорожнього руху. Це забезпечує більшу безпеку та ефективність під час руху. У галузі відеоспостереження, застосування

YOLOv3 дозволяє системам безпеки швидко і точно виявляти підозрілі об'єкти або поведінку, допомагаючи запобігати злочинам і оперативно реагувати на потенційні загрози.

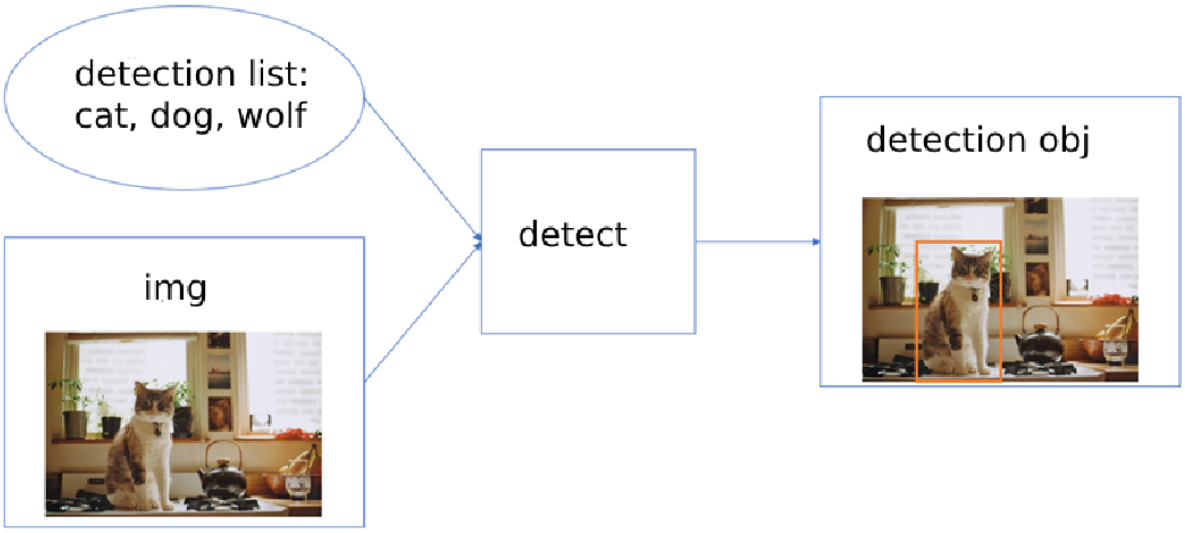

Рисунок 2.1 – Детекція з чітким переліком об’єктів

Основні етапи детекції об'єктів:

– попередня обробка зображення (масштабування, нормалізація та зміна розмірів зображення для відповідності вимогам моделі);

– техніки аугментації даних (наприклад, обертання, обтинання, зміна яскравості) для збільшення розміру тренувального набору даних;

– розділення зображення на сітку.

Зображення поділяється на сітку розміром $S \times S$. Кожна клітина сітки відповідає за виявлення об'єктів, чиї центри знаходяться в межах цієї клітини.

Прогнозування обмежувальних прямокутників та класів. Модель передбачає B обмежувальних прямокутників для кожної клітини сітки, а також ймовірності приналежності цих прямокутників до певних класів об'єктів. Кожен обмежувальний прямокутник представлений параметрами (x,y,w,h) та p_{object}, де (x,y) — координати центру, w і h — ширина і висота, p_{object} — ймовірність наявності об'єкта.

Після того, як модель детекції об'єктів (наприклад, YOLO) передбачає обмежувальні прямокутники (bounding boxes) і відповідні їм ймовірності класів, необхідно виконати постпроцесінг для покращення якості детекції. Основними етапами постпроцесінгу є підвибірка та фільтрація обмежувальних прямокутників за допомогою алгоритму Non-Maximum Suppression (NMS) і остаточний вибір найбільш імовірних об'єктів.

Постпроцесінг:

– підвибірка та фільтрація;

– остаточний вибір найбільш імовірних об'єктів.

Підвибірка та фільтрація (Non-maximum Suppression, NMS): Видаляються дублікати та перекриваючі прямокутники, залишаючи тільки ті, які мають найвищу ймовірність. Підвибірка та фільтрація (Non-maximum Suppression, NMS) Non-maximum Suppression (NMS) є ключовим методом для обробки результатів детекції. Основна мета NMS полягає у видаленні перекриваючих обмежувальних прямокутників, залишаючи лише ті, які мають найвищу ймовірність, щоб уникнути дублювання.

Кроки виконання NMS:

– функція втрат;

– сортування;

– основний цикл NMS.

Функція втрат - набір обмежувальних прямокутників з відповідними їм ймовірностями. Порогове значення для перекриття (Intersection over Union, IoU) $IoU_{threshold}$.

Сортування - сортування обмежувальні прямокутники за спаданням їхніх ймовірностей p_i.

Основний цикл NMS - ініціалізуйте порожній список для збереження кінцевих результатів $R = []$. Поки список обмежувальних прямокутників B не порожній. Виберіть обмежувальний прямокутник b_i з найвищою ймовірністю p_i та додайте його до результатів R, видаліть b_i з B. Видаліть з B усі обмежувальні прямокутники b_j для яких $IoU(b_i, b_j) > IoU_{threshold}$.

Вихідні дані - список R містить обмежувальні прямокутники після застосування NMS.

Обчислення IoU де $Area(b_i \cap b_j)$ — площа перетину двох прямокутників, а $Area(b_i \cup b_j)$ — площа їх об'єднання.

Остаточний вибір найбільш імовірних об'єктів - після виконання NMS отримується список обмежувальних прямокутників, які мають найвищу ймовірність і не перекриваються один з одним значним чином. Цей список представляє остаточний набір виявлених об'єктів.

Видалення малоймовірних детекцій виконується фільтрація обмежувальних прямокутників на основі їхніх ймовірностей. Якщо ймовірність об'єкта p_i менша за певний поріг confidence threshold, такий об'єкт видаляється.

Формування остаточного результату для кожного обмежувального прямокутника визначається його клас на основі найбільш імовірного класу серед прогнозів моделі. Результати містять координати обмежувальних прямокутників, ймовірності об'єктів та відповідні класи.

Переваги NMS:

– усунення дублікативних детекцій;
– підвищення точності.

Усунення дублікативних детекцій забезпечує, що кожен об'єкт на зображенні представлений лише одним обмежувальним прямокутником.

Підвищення точності залишаються лише найбільш надійні детекції, що зменшує кількість помилкових позитивних спрацьовувань.

Недоліки NMS:

– налаштування порогу IoU;
– обчислювальна складність.

Налаштування порогу IoU: Вибір оптимального значення порогу IoU є важливим для балансу між точністю і повнотою (recall) детекції.

Обчислювальна складність: NMS може бути обчислювально інтенсивним для великої кількості обмежувальних прямокутників, особливо в режимі реального часу.

Постпроцесінг є критичним етапом у детекції об'єктів, оскільки дозволяє отримати більш точні та надійні результати. Застосування алгоритму Non-maximum Suppression є стандартним методом для видалення дублікативних детекцій, що значно покращує якість кінцевого результату. Завдяки постпроцесінгу, сучасні моделі детекції об'єктів можуть забезпечувати високу точність навіть у складних умовах реального світу.

Методи детекції об'єктів:

- регіональні пропозиції (Region Proposals);
- одноетапні методи (Single Shot Detectors);
- архітектури на основі Attention та Transformers;
- застосування детекції об’єктів.

Регіональні пропозиції (Region Proposals) - R-CNN (Regions with CNN features), Fast R-CNN, Faster R-CNN,

R-CNN (Regions with CNN features) - використовує методи виділення регіонів пропозицій, після чого застосовується CNN для класифікації та локалізації об'єктів у кожному регіоні.

Fast R-CNN оптимізує R-CNN шляхом застосування CNN до всього зображення, а не до кожного регіону окремо, що значно підвищує швидкість.

Faster R-CNN додає мережу регіональних пропозицій (Region Proposal Network, RPN) для генерації пропозицій регіонів, що ще більше підвищує ефективність.

Одноетапні методи (Single Shot Detectors) YOLO (You Only Look Once) виконує детекцію об'єктів у реальному часі, поділяючи зображення на сітку та здійснюючи одночасне прогнозування обмежувальних прямокутників та класів об'єктів для кожної клітини.

SSD (Single Shot MultiBox Detector) подібний до YOLO, але використовує декілька розмірів сіток та масштаби для покращення виявлення об'єктів різних розмірів.

Архітектури на основі Attention та Transformers - DETR (DEtection TRansformers): Використовує архітектуру трансформерів для детекції об'єктів, що дозволяє моделі вчитися взаємодії між різними частинами зображення.

Застосування детекції об'єктів складається з:

– автономні транспортні засоби;
– відеоспостереження;
– медична діагностика;
– розумні міста;
– робототехніка.

Автономні транспортні засоби - виявлення пішоходів, автомобілів, дорожніх знаків для забезпечення безпеки на дорогах.

Відеоспостереження - виявлення підозрілих об'єктів або осіб у режимі реального часу.

Медична діагностика - автоматичне виявлення аномалій на медичних зображеннях, таких як рентген або МРТ.

Розумні міста - моніторинг трафіку, управління міськими ресурсами, забезпечення безпеки громадян.

Робототехніка - навігація та взаємодія роботів з навколишнім середовищем [4].

Іншим типом, який останніми роками набирає популярність через більшу гнучкість, а також найчастіше відмінної роботи без попереднього донавчання, є так званий open vocabulary object detection. Основна ідея детекторів, що вирішують завдання розпізнавання з "відкритим словником", є використання не самих цілісних міток класів, а ембедінгів (векторних уявлень) назв цих класів. Завдяки цьому вони можуть знаходити навіть ті класи, які не були заздалегідь задані, а також добре працювати з фразами. Наприклад, ми можемо змусити модель шукати не просто котів, а конкретну породу, навіть

якщо її не було у навчанні. Такі детектори здатні знаходити практично необмежену кількість класів об'єктів на зображенні (рис. 2.2):

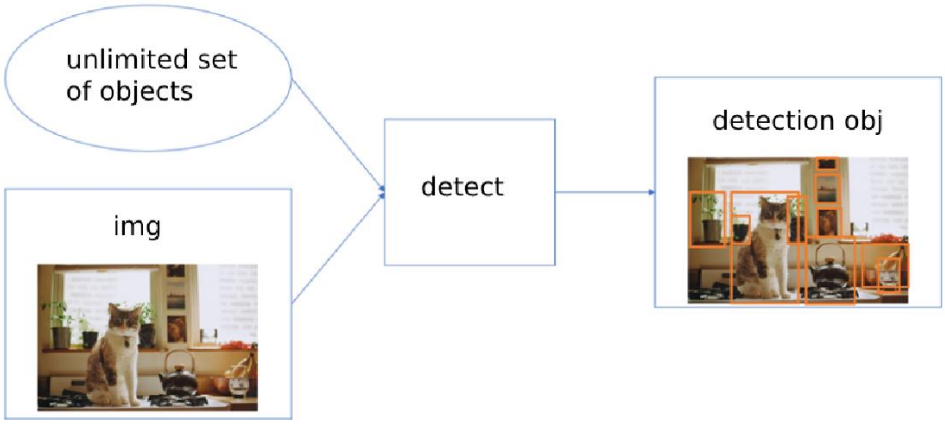

Рисунок 2.2 – Детекція з необмеженою кількістю об’єктів

Тут варто відзначити, що, кажучи необмежене, ми маємо на увазі, що людина в будь-якому випадку повинна визначити список об'єктів, які вона хоче розпізнавати, але цей список може бути як завгодно великим (фактично кінцевим). Наприклад, таким детекторам можна передати 21 тисячу найменувань класів з повного датасету ImageNet і вони справді намагатимуться розпізнати кожен із них.

Як уже було зазначено на початку, Yolo-world якраз відноситься до детекторів, здатних вирішувати більш цікаве друге завдання. Далі запропонували використати так званий prompt-then-detect підхід. Це означає, що якщо раніше ми один раз створювали великий список слів, що цікавлять нас (online vocabulary, вектори в якому статичні), то тепер ми формуємо так звані промти, які кодуються в offline vocabulary і вже ці ембеддінги йдуть далі пайплайном. Такий підхід дозволяє зменшити кількість обчислень для кожного введення, а також забезпечує гнучкість, коригуючи словник у міру потреби (рис 2.3).

Крім того, такий підхід значно знижує потребу в оновленні всього словника, що вимагає великих обчислювальних ресурсів та часу. Замість цього, зміни вносяться тільки в необхідні промти, що робить систему більш

гнучкою та швидкодіючою. Це особливо важливо в умовах, коли швидкість та точність обробки даних є критичними.

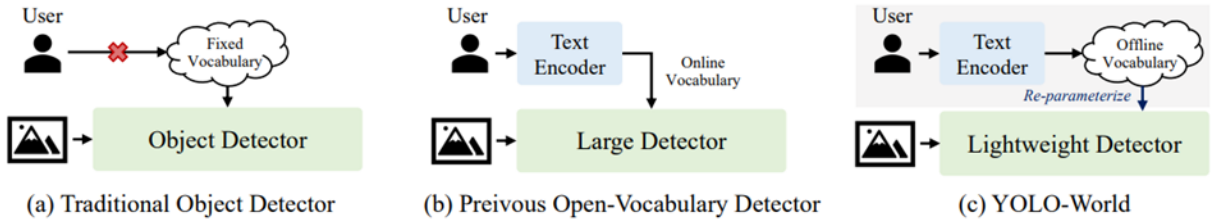

Рисунок 2.3 – Різниця між детекторами

Отже, усвідомивши потенціал нової моделі, потрібно перейти до розбору самої архітектури і зрозуміти, в чому її основні особливості та в чом вона демонструє ефективність [10].

Архітектуру YOLO-World як і багато сучасних нейромережевих архітектур, YOLO-World можна розбити на безліч окремих блоків. Давайте докладніше зупинимося на деяких із них (рис 2.4).

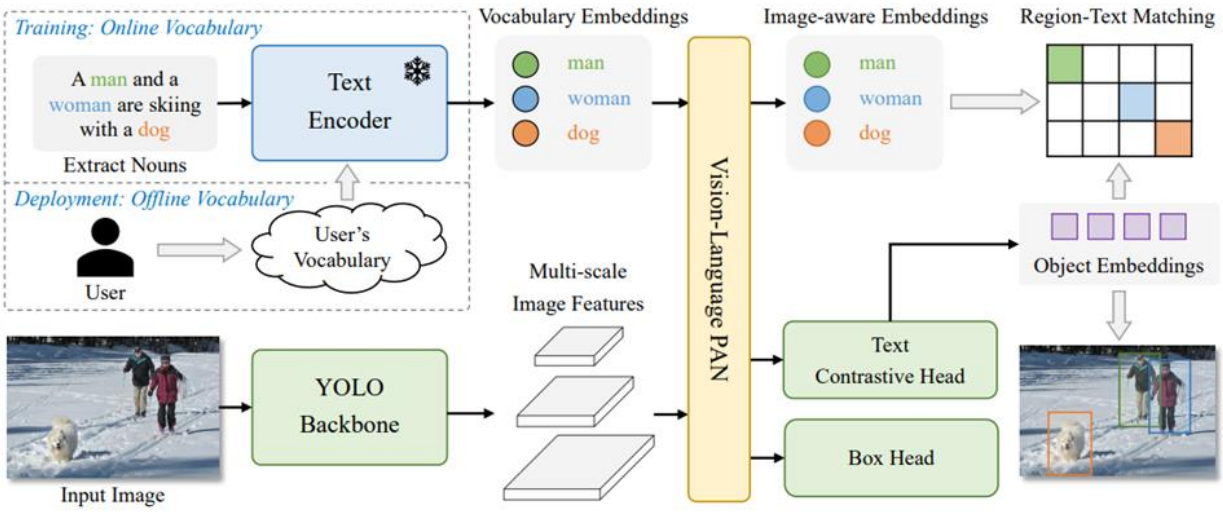

Рисунок 2.4 – Архітектура

YOLO Detector - для отримання фіч зображення використовується відносно нова YOLOv8. Вона ж у свою чергу містить backbone Darknet як кодувальник та мережу PAN для формування multi-scale фічів , що дозволяє

досягти високої точності та швидкості детекції. Text Encoder - для отримання ембеддингів тексту використовується CLIP, що вже добре зарекомендував себе, а саме його трансформер для кодування тексту. Кодувальник CLIP формує вектори текстів так, щоб їх можна було добре порівняти з відповідними векторними уявленнями зображень (висока косинусна подібність).

Re-parameterizable Vision-Language PAN один із основних та головних будівельних блоків усієї архітектури. Складається він з top-down та bottom-up частин, всередині яких проводиться зіставлення раніше вилучених текстових ембеддингів та multi-scale фічів зображення. Блок у собі включає 2 основних компоненти: CSPLayer і Image-Pooling Attention. Говорячи обивницькою мовою, перший намагається додати мовну інформацію до елементів зображення, а другий навпаки, закласти інформацію із зображення до текстових ембеддінгів (рис 2.5):

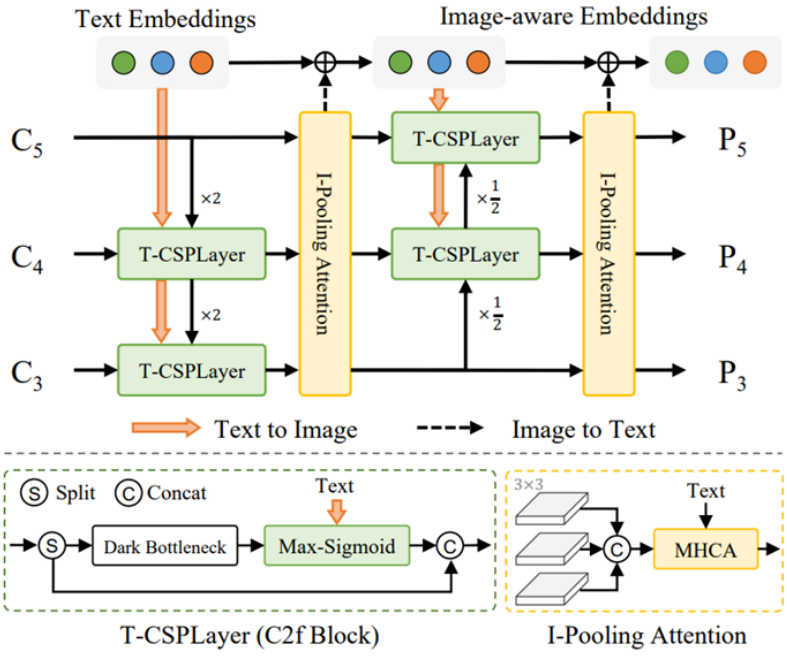

Рисунок 2.5 - Vision-Language PAN

Після блоку RepVL-PAN слідують модулі Box Head і Text Contrastive Head. Перший модуль відповідає за передбачення обмежувальних рамок для

об'єктів, тоді як другий прогнозує їх ембеддінги, базуючись на близькості між об'єктом і текстом.

Таким чином, наприкінці пайплайну ми отримуємо обмежувальні рамки для об'єктів, ембеддінги цих об'єктів на зображенні та вектори текстових описів класів, які потрібно виявити. За допомогою методу порівняння попарних близькостей векторів об'єктів і текстів у рамках боксів, на виході отримуємо список знайдених класів із відповідними ймовірностями (при заданому порозі близькості).

Це загальний огляд моделі без заглиблення в математичні деталі та формули. Детальніше про процес навчання з використанням Region-Text Contrastive Loss, а також про численні експерименти з навчання для різних завдань на різних датасетах і порівняння з попередніми підходами можна знайти в додаткових матеріалах.

Звісно, архітектура важлива, але які метрики демонструє нова передова модель (SOTA) у розпізнаванні об'єктів? Автори надають графік "Швидкість-Точність" (Speed-Accuracy), який показує, що продуктивність моделі зросла в 20 разів при збереженні або навіть підвищенні точності (mAP), виміряної на датасеті LVIS. (рис 2.6). Цей значний приріст продуктивності обумовлений вдосконаленням архітектури YOLOv8, зокрема інтеграцією більш ефективних методів обробки зображень та оптимізацією обчислювальних процесів. Завдяки цьому нововведенню, модель здатна швидше обробляти великі обсяги даних, забезпечуючи високу точність розпізнавання об'єктів, що особливо важливо для застосувань в реальному часі. графік демонструє, що YOLOv8 не тільки перевершує попередні версії YOLO та інші моделі в плані швидкості, але й забезпечує кращу узгодженість результатів при обробці зображень різних розмірів і складностей. Це робить її універсальним інструментом для широкого спектра завдань, від автономних транспортних засобів до систем відеоспостереження і мобільних додатків.

Завдяки підвищеній швидкості та точності, YOLOv8 може ефективно використовуватись у системах автономного водіння, де швидке та точне

розпізнавання об'єктів є критичним для безпеки руху. Модель може швидко адаптуватись до змін в оточенні, забезпечуючи своєчасну реакцію на появу перешкод або зміну дорожньої обстановки.

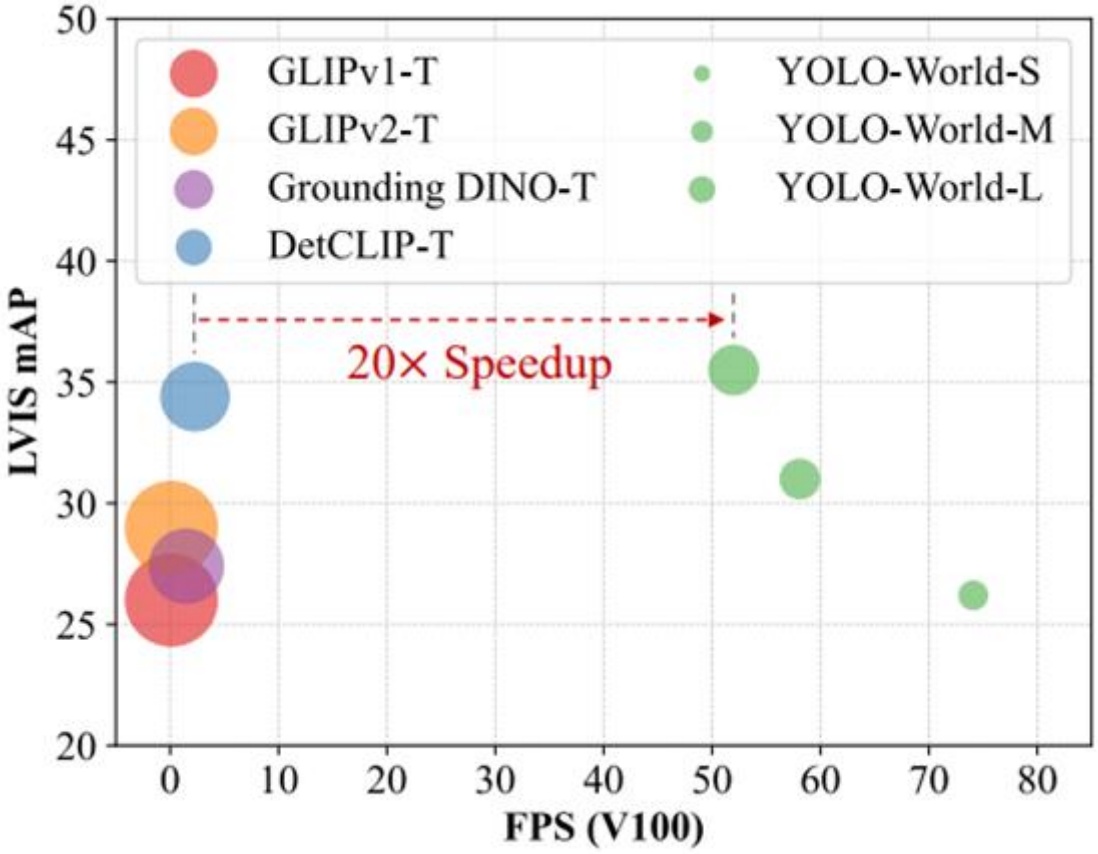

Рисунок 2.6 – SOTA

А моделі є важливим аспектом, але ключовими показниками є точність і швидкість її роботи. Автори дослідження надають графік "Швидкість-Точність" (Speed-Accuracy), який демонструє, що нова модель досягає значного покращення продуктивності. Зокрема, продуктивність збільшилася в 20 разів при збереженні або навіть підвищенні точності (mAP), виміряної на датасеті LVIS. Це суттєве покращення робить модель особливо корисною для застосувань, де критично важливі швидкість і точність, таких як реального часу системи відеоспостереження, автономні транспортні засоби та інші.

Детальний аналіз показників ефективності моделі показує, що вона здатна забезпечувати високу точність розпізнавання об'єктів навіть при значно

швидшій обробці зображень. Цей баланс між швидкістю і точністю досягається завдяки оптимізаціям, впровадженим у її архітектуру, включаючи використання легкого backbone і спеціалізованих методів постобробки, таких як Non-Maximum Suppression (NMS).

Для досягнення таких високих показників, модель навчалась з використанням Region-Text Contrastive Loss, що дозволяє ефективно співвідносити текстові описи з візуальними об'єктами на зображеннях [9]. Крім того, було проведено численні експерименти на різних датасетах для адаптації моделі до конкретних завдань і для порівняння її з попередніми рішеннями. Це дозволило підтвердити переваги нової архітектури в різних сценаріях застосування.

Завдяки своїй інноваційній архітектурі та оптимізаціям, нова модель YOLO демонструє значні покращення у швидкості та точності розпізнавання об'єктів. Використання легкого backbone, ефективних методів постобробки та продуманих підходів до навчання робить її передовим інструментом у сфері комп'ютерного зору. Ці вдосконалення забезпечують модель здатністю працювати в реальному часі та з високою точністю, що є критично важливим для широкого спектру практичних застосувань.

Таке значне покращення продуктивності головним чином пов'язане з використанням легкого Yolo backbone для вилучення ознак зображення. У той час як попередні архітектури використовували більш важкі та обчислювально інтенсивні трансформери, такі як Swin, Yolo застосовує більш ефективну та менш ресурсоємну структуру. Це дозволяє досягати високої швидкості обробки та кращої продуктивності, зберігаючи при цьому точність детекції об'єктів. Додатково, оптимізація архітектури YOLO включає використання сучасних методів зменшення розмірності та підвищення обчислювальної ефективності, що ще більше сприяє покращенню загальної продуктивності системи.

Додатково до використання легкого backbone, в архітектурі YOLO впроваджено кілька інших оптимізацій, які сприяють підвищенню

продуктивності. Серед них можна виділити: використання глибоких згорткових шарів (deep convolutional layers) - ці шари забезпечують більш ефективне вилучення складних ознак зображення, що дозволяє моделі краще розпізнавати об'єкти різних розмірів та форм.

Механізми багатомасштабного виявлення (multi-scale detection) - YOLO здатна одночасно обробляти об'єкти різних масштабів, що дозволяє моделі бути більш універсальною та ефективною у різних умовах [19].

Швидка та ефективна постобробка (post-processing) - використання алгоритму Non-Maximum Suppression (NMS) для видалення дублікативних детекцій забезпечує високу точність виявлення об'єктів, зменшуючи кількість хибнопозитивних результатів.

Удосконалені методи регуляризації (regularization techniques) - такі методи, як Dropout та Batch Normalization, допомагають уникнути перенавчання та покращують узагальнюючу здатність моделі.

Оптимізація обчислювальних витрат (computational cost optimization) - завдяки ефективному розподілу обчислювальних ресурсів, YOLO може працювати в реальному часі навіть на пристроях з обмеженими ресурсами, таких як мобільні телефони та вбудовані системи.

Ці оптимізації не тільки підвищують швидкість обробки та точність виявлення об'єктів, але й роблять YOLO більш доступною для широкого спектру застосувань, від автономних транспортних засобів до систем відеоспостереження та медичних діагностичних інструментів. Завдяки цим вдосконаленням, YOLO залишається однією з найпопулярніших та найефективніших моделей для детекції об'єктів у комп'ютерному зорі.

Якість розпізнавання цілком зрозуміла головним блоком (RepVL-PAN), який використовує багаторівневе крос-модальне поєднання фічів (тексти та картинки).

Особливості нового YOLO-World детектора:

– здатність розпізнавання необмеженної кількості об’єктів;

– large модель показує real-time швидкість на інференсі (перша мережа з таким показником для завдання OVD);

– використовує як Yolov8 [25] CLIP, так і RepVL-PAN.

Оскільки YOLO потрібно лише один раз подивитися на зображення для виконання детекції, метод ковзного вікна [27] не підходить для цього підходу. Замість цього зображення ділиться на сітку з комірками розміром S на S. Кожна комірка може містити декілька різних об'єктів для розпізнавання.

По-перше, кожен осередок відповідає за прогнозування кількох обмежувальних прямокутників (bounding boxes). Також кожен осередок прогнозує значення довіри (confidence value) для кожного з цих обмежувальних прямокутників. Іншими словами, це значення вказує на ймовірність того, що у визначеній області знаходиться якийсь об'єкт. Тобто, якщо в якійсь комірці сітки немає певного об'єкта, важливо, щоб значення довіри для цієї області було низьким.

Крім цього, кожен осередок також прогнозує ймовірності класів для кожного з виявлених об'єктів. Це означає, що, маючи значення довіри та ймовірності класів, модель може визначити, який саме об'єкт знаходиться в межах кожного bounding box, і наскільки впевнена вона в цьому передбаченні. Це робить процес детекції об'єктів швидшим та ефективнішим, оскільки всі прогнози здійснюються одночасно в одному проході моделі по зображенню. Коли ми візуалізуємо всі передбачення, ми отримуємо карту об'єктів [1] та впорядкованих за довірчим значенням рамки (рис 2.7). Таке впорядкування дозволяє легко відфільтровувати менш надійні передбачення, залишаючи лише найбільш вірогідні, що покращує точність та надійність системи. Завдяки цим можливостям, модель YOLOv8 стає потужним інструментом для задач, що вимагають високої швидкості обробки та точної ідентифікації об'єктів у реальному часі. Це впорядкування дозволяє легко відфільтровувати менш надійні передбачення, залишаючи лише найбільш вірогідні, що покращує точність та надійність системи. Завдяки цим можливостям, модель

YOLOv8 стає потужним інструментом для задач, що вимагають високої швидкості обробки та точної ідентифікації об'єктів у реальному часі.

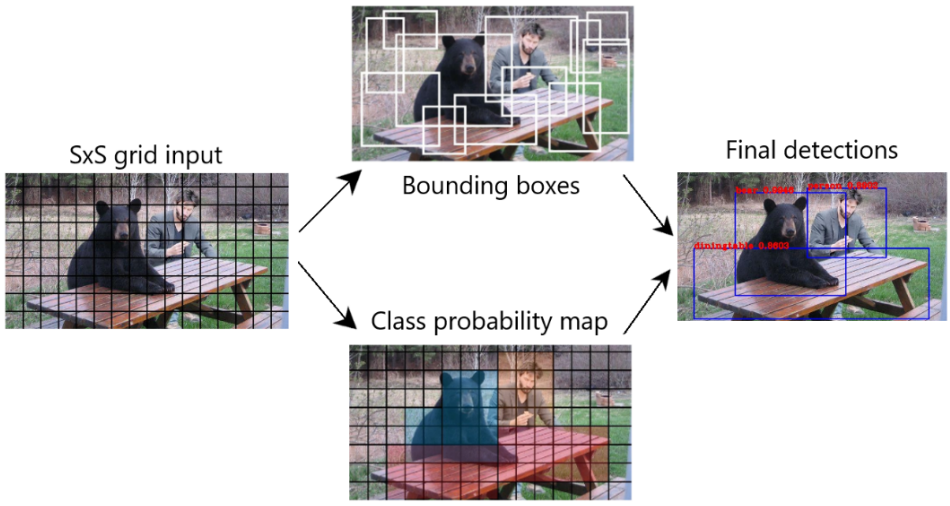

Рисунок 2.7 – Одночасні прогнози

По-друге, кожен осередок відповідає за передбачення ймовірностей класів. Це означає не те, що якийсь осередок обов'язково містить об'єкт, а лише вказує на ймовірність присутності об'єкта в ньому. Наприклад, якщо осередок передбачає автомобіль, це не гарантує, що автомобіль дійсно знаходиться в цьому осередку. Це просто означає, що якщо в осередку є об'єкт, то найбільш ймовірно, що це автомобіль.

У YOLO використовуються anchor boxes (якорні рамки або фіксовані рамки) для прогнозування обмежувальних прямокутників (bounding boxes). Ідея anchor boxes [26] полягає в попередньому визначенні кількох різних форм рамок. Таким чином, ми можемо зробити декілька передбачень для кожного осередку, використовуючи ці рамки як шаблони. Зазвичай використовуються два або більше anchor boxes, але їх кількість може бути збільшена в залежності від вимог задачі. Ці якорні рамки були розраховані за допомогою датасету COCO (Common Objects in Context) і кластеризації методом k-середніх (K-means clustering). Це дозволяє оптимально підбирати форми і розміри рамок для найбільш ефективного виявлення об'єктів різних типів та розмірів (рис 2.8).

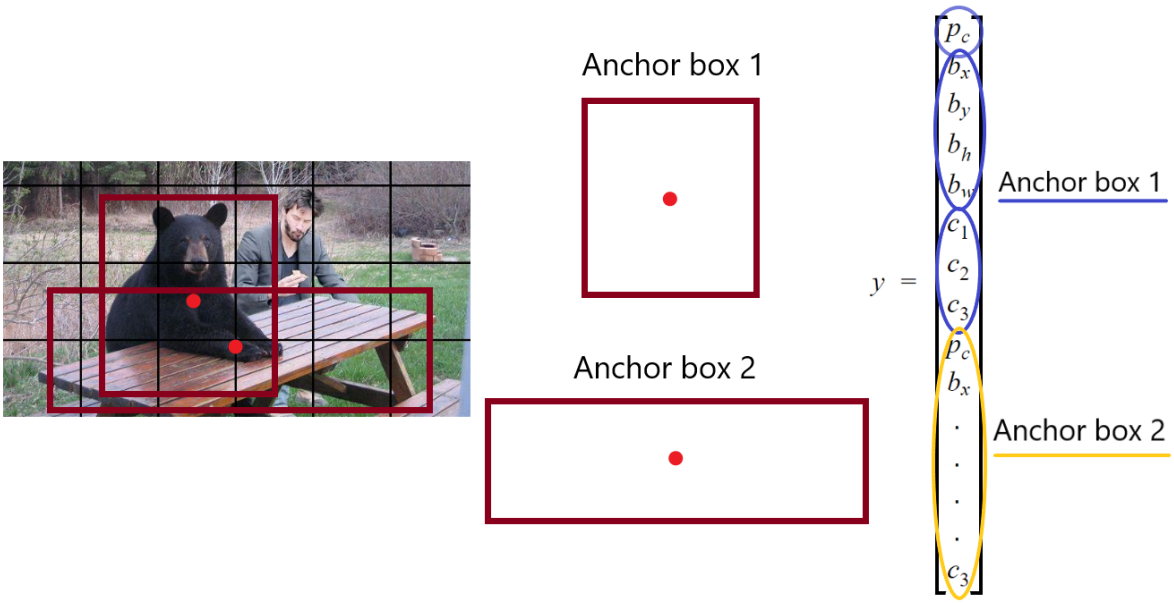

Рисунок 2.8 – Anchor boxes

Є сітка, де кожен осередок передбачає для кожного bounding box:

– 4 координати (t_x, t_y, t_w, t_h);

– 1 objectness error (помилка об'єктності), яка є показником впевненості у присутності того чи іншого об'єкта;

– декілька ймовірностей класів.

Якщо ж є деяке зміщення від верхнього лівого кута на c_x , c_y то прогнози будуть відповідати - p_w (ширина) і p_h (висота) відповідають ширині та висоті bounding box. Замість того, щоб передбачати зміщення як у минулій версії YOLOV2, автори прогнозують координати розташування щодо розташування комірки. Загалом 4 – для bounding box, 1 – для objectness prediction (прогнозування об'єктності). За один прохід ми можемо пройти від вхідного зображення до вихідного тензора, який відповідає виявленим об'єктам на зображенні. Також варто відзначити, що YOLOv3 прогнозує bounding box у трьох різних масштабах [21].

Тепер, якщо ми візьмемо ймовірність і помножимо їх на довірчі значення, ми отримаємо всі bounding box, зважені на ймовірність утримання

цього об'єкта. Просте знаходження порогового значення позбавить нас прогнозів з низьким довірчим значенням. Після цього можемо застосувати метод не-максимального придушення (NMS), щоб видалити зайві коробки, що перекриваються, залишивши тільки ті, що мають найвищі ймовірності. Це допоможе зменшити кількість помилкових спрацьовувань і підвищити точність нашої моделі.

Таким чином, комбінування ймовірностей, порогових значень та NMS дозволяє оптимально ідентифікувати об'єкти на зображенні, забезпечуючи високий рівень довіри до виявлених об'єктів.

Для наступного кроку важливо визначити метрику IoU (Intersection over Union/Перетин над об'єднанням). Ця метрика дорівнює співвідношенню площі областей, що перетинаються, до площі областей об'єднаних (рис 2.9):

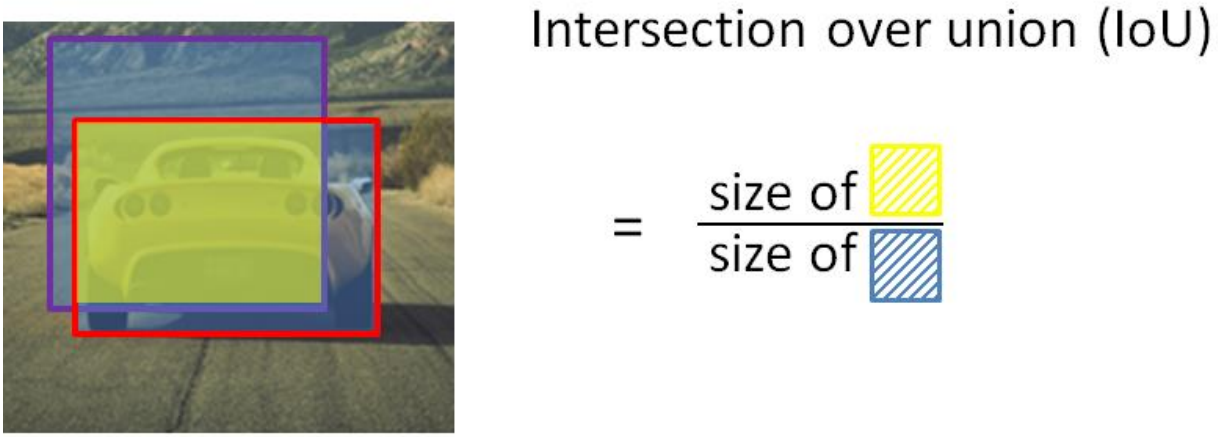

Рисунок 2.9 - Перетин над об'єднанням

Після цього все одно можуть залишитися дублікати, і щоб їх позбутися потрібно використовувати “пригнічення не-максимумів” (non-maximum suppression). Пригнічення не-максимумів полягає в наступному: алгоритм бере bounding box з найбільшою ймовірністю приналежності до об'єкта, потім, серед інших межують bounding box'ів з даної області [30], візьме один з найвищим IoU і пригнічує його.

Зважаючи на те, що все робиться за один прогін, ця модель працюватиме майже так само швидко, як і класифікація. До того ж, всі виявлення передбачаються одночасно, що означає, що модель неявно враховує глобальний контекст. Окрім цього, одночасне передбачення всіх виявлень дозволяє моделі більш ефективно використовувати обчислювальні ресурси та зменшує затримки, пов'язані з обробкою кожного об'єкта окремо. Завдяки цьому, модель може швидко адаптуватися до різноманітних сцен та об'єктів, забезпечуючи високу продуктивність навіть при обробці великої кількості даних. Отже, поєднання швидкості, глобального контексту та ефективного використання ресурсів робить цю модель надзвичайно потужною.

Простіше кажучи, модель може дізнатися, які об'єкти зазвичай зустрічаються разом, їх відносний розмір і розташування об'єктів і так далі (рис 2.10).

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Рисунок 2.10 - Зображення таблиці розширення мережи Darknet-53

Тепер настав час для реалізації YOLOV3 [14]. Ідея полягає в тому, щоб використовувати тільки згорткові шари. Так як їх тут 53, то найпростішим способом є створення функції, в яку передаватимемо важливі параметри, що змінюються від шару до шару (рис 2.11). Ця функція дозволить динамічно створювати шари з урахуванням параметрів, таких як кількість фільтрів, розмір ядра згортки, крок, методи нормалізації та активації.

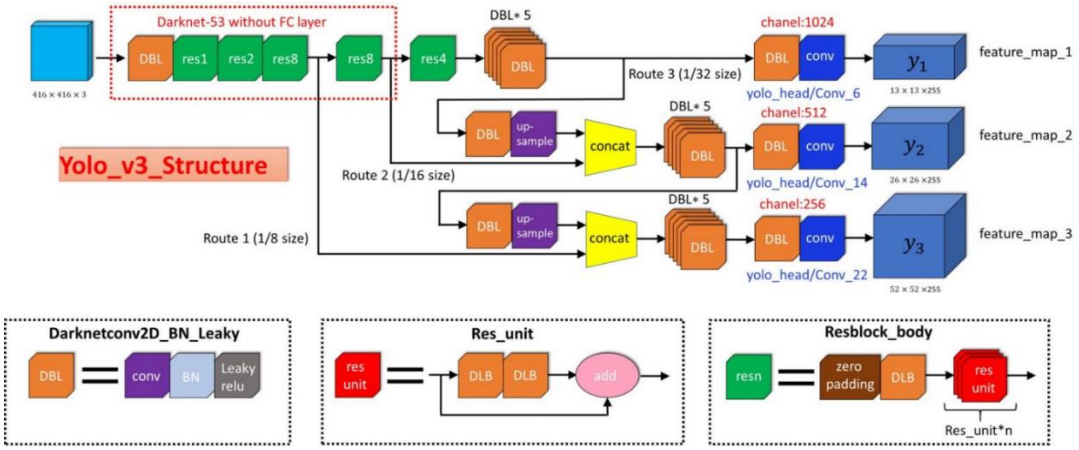

Рисунок 2.11 – Згорткові шари

Залишкові блоки (Residual Blocks) у діаграмі архітектури YOLOv3 застосовуються вивчення ознак [3]. Залишковий блок містить кілька згорткових шарів і додаткові зв'язки для обходу цих шарів (рис 2.12).

Залишковий блок складається з двох основних частин: основного шляху, де виконуються згорткові операції, і пропускного шляху (skip connection), який додає вхідні дані безпосередньо до виходу блоку. Окрім цього, YOLOv3 використовує багатошарові детектори (Multi-scale Detectors), які дозволяють виявляти об'єкти різних розмірів на різних рівнях мережі.

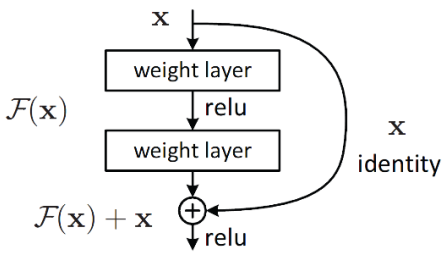

Рисунок 2.12 – Залишкові блоки

3 КОМП’ЮТЕРНІ МОДЕЛІ YOLO

YOLO (You Only Look Once) є однією з найпопулярніших моделей для детекції об'єктів на зображеннях і відео. З моменту її створення було розроблено кілька версій, кожна з яких пропонувала суттєві поліпшення щодо швидкості, точності та ефективності. Нижче наведено опис основних версій та вдосконалень YOLO.

3.1 YOLOv1

YOLOv1 (You Only Look Once, версія 1) була представлена Джозефом Редмоном у 2016 році. Це був революційний підхід до детекції об'єктів, який поєднував простоту і швидкість, дозволяючи досягти високої продуктивності в режимі реального часу. Розглянемо детальніше основні аспекти YOLOv1.

Концепція YOLOv1 представляє новий підхід до детекції об'єктів, відмінний від традиційних методів, які розбивали задачу на дві окремі частини: локалізацію об'єктів і їх класифікацію. Замість цього, YOLOv1 об'єднує ці два завдання, виконуючи їх одночасно в рамках єдиного проходу нейронної мережі.

Основні ідеї YOLOv1:

- єдине представлення (Single Pass);
- сітка комірок;
- bounding boxes і confidence scores;
- класифікація об’єктів.

Єдине Представлення (Single Pass) YOLOv1 розглядає всю задачу детекції об'єктів як єдину задачу регресії, перетворюючи вхідне зображення безпосередньо на вектори обмежувальних рамок та ймовірностей класів.

Модель потребує лише один погляд на зображення для передбачення як рамок, так і класів об'єктів.

Сітка комірок - вхідне зображення ділиться на сітку розміром S на S (наприклад, 7 на 7). Кожна комірка відповідає за виявлення об'єктів, центри яких знаходяться всередині цієї комірки. Це дозволяє моделі визначити розташування об'єктів відносно координат комірки.

Bounding Boxes і Confidence Scores кожна комірка передбачає два обмежувальні рамки та довірчі значення для кожної рамки. Довірче значення вказує на ймовірність того, що в цій рамці знаходиться об'єкт, а також на точність передбачених координат рамки. Якщо комірка не містить об'єкт, довірче значення повинно бути низьким.

Класифікація Об’єктів кожна комірка також прогнозує ймовірності класів для об'єктів, що можуть бути присутніми в передбачених рамках. Ці ймовірності використовуються для визначення класу об'єкта (наприклад, автомобіль, велосипед, пішохід тощо).

Архітектура YOLOv1 - була інноваційною для свого часу, запропонувавши новий спосіб детекції об'єктів на зображеннях. Давайте детально розглянемо кожен аспект цієї архітектури (рис 3.1).

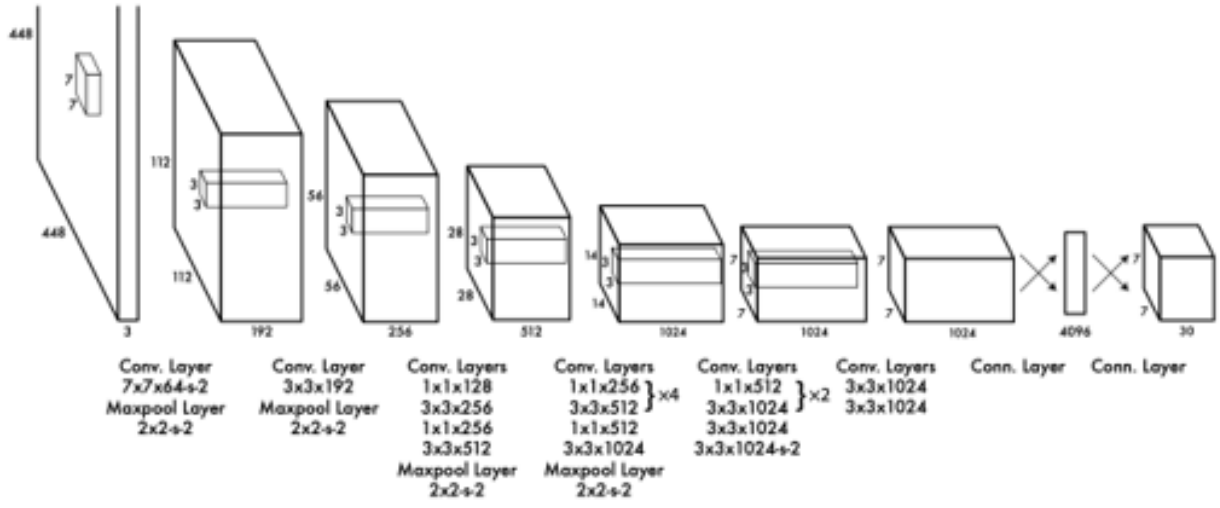

Рисунок 3.1 – Архітектура YOLOv1

Вхідні Дані - вхідне зображення має фіксований розмір (наприклад, 448x448 пікселів).

Нейронна Мережа - YOLOv1 використовує нейронну мережу з 24 згортковими шарами (convolutional layers) та 2 повнозв'язними шарами (fully connected layers). Конволюційні шари відповідають за вилучення ознак із зображення, а повнозв'язні шари здійснюють передбачення обмежувальних рамок і класів об'єктів [15].

Вихідний шар мережі представляє результати у вигляді тензора. Цей тензор потім обробляється для вилучення інформації про кожний передбачуваний об'єкт, включаючи координати bounding boxes, імовірність об'єкта та класові ймовірності. Модель використовує функцію softmax для передбачення класових ймовірностей та сигмоїдну функцію для передбачення ймовірності об'єкта і координат bounding boxes.

Резюмуючи, вихідний тензор містить повну інформацію про всі виявлені об'єкти в зображенні, включаючи їх позиції та класові ймовірності. Ця інформація потім проходить через процес післяобробки, такий як не-максимальне придушення (NMS), для вилучення остаточних передбачень і уникнення зайвих передбачень одного й того ж об'єкта. Це дозволяє моделі ефективно та точно виявляти і класифікувати об'єкти в реальному часі.

Процес передбачення в YOLOv1 складається з декількох кроків:

– поділ зображення на сітку;
– передбачення рамок і класів;
– фільтрація результатів.

Поділ зображення на сітку - вхідне зображення ділиться на S на S комірок.

Передбачення рамок і класів - кожна комірка передбачає обмежувальні рамки, довірчі значення і ймовірності класів [8].

Фільтрація результатів виконується за допомогою порогових значень для відсіювання неправдивих передбачень і видалення перекриттів.

YOLOv1 мала деякі обмеження:

– точність;

– роздільна здатність сітки;

– регресія на обмежувальні рамки.

Точність проблеми з детекцією дрібних об'єктів і об'єктів, що перекриваються [2].

Роздільна здатність сітки має фіксований розмір сітки (7 на 7) та може бути недостатнім для складних сцен з багатьма об'єктами.

Регресія на обмежувальні рамки - точність передбачення координат рамок може бути нижчою в порівнянні з іншими методами.

Для навчання YOLOv1 використовувалися великі анотовані датасети, такі як PASCAL VOC [16].

Функція Втрат включає три компоненти:

– втрати локалізації;

– втрати довірчих значень;

– втрати класифікації.

Втрати локалізації (розбіжності між передбаченими та істинними координатами рамок), втрати довірчих значень (розбіжності між передбаченими та істинними ймовірностями об'єктів), і втрати класифікації (розбіжності між передбаченими та істинними класами об'єктів).

Для оптимізації використовувався алгоритм стохастичного градієнтного спуску (SGD) з відповідними гіперпараметрами, такими як швидкість навчання, імпульс (momentum) і регуляризація [7].

Архітектура YOLOv1 стала важливим кроком у розвитку методів детекції об'єктів. Вона продемонструвала, що можна досягти високої продуктивності і швидкості за допомогою єдиного проходу нейронної мережі. Хоча ця версія мала свої обмеження, вона заклала основу для подальших удосконалень у наступних версіях YOLO, що призвело до створення потужних моделей для різних застосувань.

Після успіху YOLOv1, було розроблено YOLOv2, яка покращила точність та швидкість завдяки використанню таких технік, як Batch

Normalization, більш точних anchor boxes, і більшої кількості шарів у нейронній мережі. Ці вдосконалення дозволили моделі краще розрізняти об'єкти на різних масштабах і в різних умовах освітлення. Згодом з'явилася YOLOv3, яка додатково підвищила точність і здатність виявляти дрібні об'єкти завдяки впровадженню залишкових блоків (Residual Blocks) і багаторівневого виявлення. Залишкові блоки допомагають у глибокому навчанні, покращуючи передачу градієнтів і стабільність моделі. Багаторівневе виявлення дозволяє моделі передбачати об'єкти різних розмірів на різних масштабах, що значно підвищує ефективність і точність детекції.

3.2 YOLOv2 (YOLO9000)

YOLOv2, також відома як YOLO9000, була випущена в 2016 році Джозефом Редмоном та його командою. Ця версія значно покращила точність і продуктивність у порівнянні з YOLOv1 і представила декілька інноваційних ідей, далі розглянемо основні аспекти YOLOv2.

Покращення в YOLOv2:

- архітектурні зміни;
- анкори (Anchor Boxes);
- висока роздільна здатність;
- ймовірність класу.

Архітектурні зміни перш за все стосується Darknet-19, YOLOv2 використовує нову базову архітектуру нейронної мережі під назвою Darknet-19. Вона складається з 19 конволюційних шарів і 5 шарів максимального об’єднання (max-pooling). Це легка та ефективна мережа, яка дозволяє досягти високої продуктивності. Darknet 19 не тільки ефективний, але й демонструє відмінну продуктивність. У задачах класифікації зображень він успішно конкурує з провідними моделями, такими як VGG і ResNet, при цьому забезпечуючи високу швидкість роботи - 200 кадрів в секунду. Оригінальна

версія YOLOv1 вже була відома своєю швидкістю та точністю, і Darknet 19 продовжує цю традицію, покращуючи її ще більше (рис 3.2). Наступна версія, YOLOv3, базується на покращеній архітектурі під назвою Darknet-53. Darknet-53 використовує 53 згорткових шари і включає залишкові блоки (Residual Blocks), що дозволяють ефективніше навчати глибокі мережі.

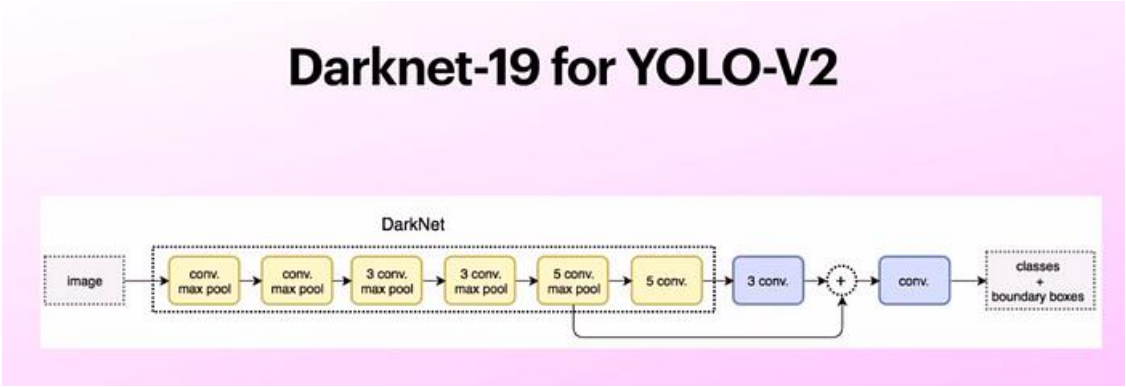

Рисунок 3.2 – Darknet-19 для другої версії YOLO

Пакетна нормалізація - всі конволюційні шари використовують пакетну нормалізацію (batch normalization), що сприяє стабільнішому та швидшому навчанню.

Анкори (Anchor Boxes) - у YOLOv2 введені анкори для прогнозування bounding box'ів. Це дозволяє моделі краще справлятися з різноманіттям розмірів і форм об'єктів. Анкори визначаються заздалегідь за допомогою кластеризації K-means на тренувальному наборі даних.

Висока Роздільна Здатність Вхідних Зображень - вхідні зображення збільшені до розміру 416x416 пікселів, що дозволяє моделі бачити більше деталей і покращувати точність передбачень [6].

Введено метод передбачення ймовірності класу окремо від довірчого значення. Це дозволяє моделі краще розрізняти об'єкти, навіть якщо їх довірче значення низьке.

YOLO v1 спочатку тренує мережу класифікаторів на зображеннях розміром 224 × 224 пікселів, а потім збільшує роздільну здатність до 448 пікселів для виявлення об'єктів (рис 3.3). Це означає, що мережа повинна

одночасно адаптуватися до задачі виявлення об'єктів і до нової роздільної здатності вхідного сигналу [5]. Таке одночасне перемикання на виявлення об'єктів і налаштування на іншу роздільну здатність може призвести до зниження продуктивності YOLO v1.

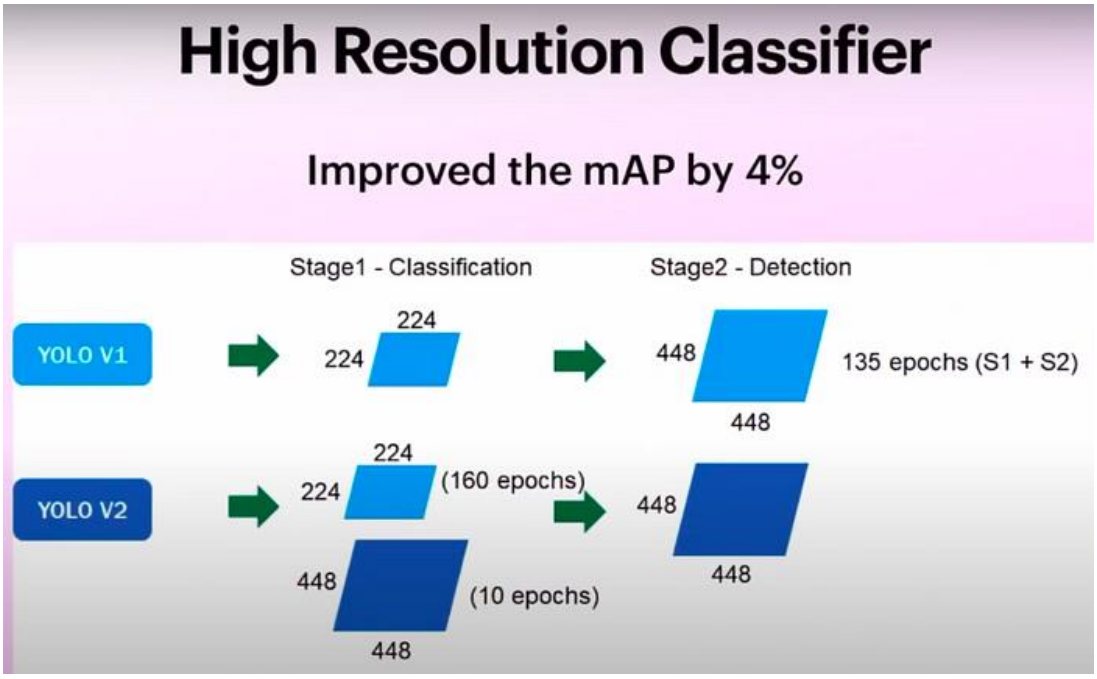

Рисунок 3.3 – Порівняння класифікаторів високої роздільної здатності

У YOLOv2 модель спочатку тренується на зображеннях розміром 224x224, а потім налаштовується на зображеннях розміром 448x448 протягом 10 епох. Це робиться з двох причин:

– адаптація;

– навчання.

Це дозволяє мережі адаптуватися до нової роздільної здатності входу перед тим, як почати навчання для виявлення об'єктів.

Це допомагає мережі навчитися виявляти важливі особливості для детекції об'єктів, оскільки вона вже пройшла навчання на великому наборі даних із зображеннями.

Для точної настройки моделі ваги з попередньо навченої моделі завантажуються в нову модель, яка потім тренується на наборі даних із

зображеннями розміром 448x448. Нова модель зазвичай навчається протягом меншої кількості епох, ніж попередньо навчена, оскільки мережа вже має певні знання про важливі особливості для детекції об'єктів. Це явище відоме як перенесене навчання (transfer learning), яке дозволяє використовувати раніше набуті знання для пришвидшення і покращення процесу навчання на нових даних [17].

Навчання на декількох датасетах, YOLO9000 здатна розпізнавати більше 9000 класів об'єктів, що є значним покращенням у порівнянні з попередніми версіями. Це досягається завдяки об'єднанню навчання на декількох датасетах. Для навчання використовується поєднання двох типів даних: анотовані зображення з точно визначеними bounding box'ами і класифіковані зображення з етикетками класів. Це дозволяє моделі навчатися на великій кількості даних навіть при відсутності точних анотацій.

Введена ієрархічна класифікація, де класи об'єктів організовані у вигляді дерева. Це дозволяє моделі ефективніше навчатися, використовуючи знання про схожість між класами. Функція втрат у YOLOv2 включає три основні компоненти: втрати локалізації, втрати класифікації і втрати довірчих значень. Це допомагає моделі точно передбачати координати bounding box'ів, класи об'єктів і довірчі значення. Використовується стохастичний градієнтний спуск (SGD) з адаптивною швидкістю навчання, що дозволяє моделі швидше сходитися.

YOLOv2 значно покращила точність детекції об'єктів у порівнянні з YOLOv1. Це досягається за рахунок використання анкорів, більшої роздільної здатності вхідних зображень і покращеної архітектури.

Незважаючи на підвищену точність, YOLOv2 зберігає високу швидкість обробки, що дозволяє використовувати її в реальному часі.

Завдяки можливості навчання на декількох датасетах, YOLO9000 може розпізнавати більше 9000 класів об'єктів, що робить її дуже гнучкою і універсальною моделлю. Хоча точність значно покращена, YOLOv2 все ще

має труднощі з детекцією дуже дрібних об'єктів. Використання анкорів може збільшити складність моделі і вимоги до обчислювальних ресурсів.

YOLOv2 (YOLO9000) стала значним кроком вперед у детекції об'єктів, поєднавши високу точність і швидкість. Інновації, такі як використання анкорів, пакетної нормалізації і навчання на декількох датасетах.

YOLOv2 видаляє повнозв'язний шар і останній шар max-pooling, збільшуючи роздільну здатність карти ознак з 7x7 до 14x14 (рис 3.4). Це покращує локалізацію об'єктів, оскільки мережа отримує більше просторової інформації з зображення.

Рисунок 3.4 - Карти функцій високої роздільної здатності

3.3 YOLOv3

YOLOv3 (You Only Look Once version 3) була представлена Джозефом Редмоном і Алі Фархаді в 2018 році, і вона продовжує удосконалювати концепції попередніх версій YOLO. YOLOv3 відзначається надзвичайною швидкістю та точністю. За показником mAP, виміряним при .5 IOU, YOLOv3

досягає результатів, порівнянних з моделлю з фокусною втратою, але працює приблизно в 4 рази швидше. Додатково, можна легко збалансувати швидкість і точність, просто змінюючи розмір моделі без необхідності повторного навчання.

Основні поліпшення в YOLOv3:

- нова архітектура;
- детекція на різних масштабах;
- розширене використання анкорів;
- поліпшена функція активації;
- логітні значення для передбачень.

Нова Архітектура, така як Darknet-53 в YOLOv3 використовує нову базову архітектуру, яка складається з 53 конволюційних шарів. Ця мережа є більш глибокою та ефективною в порівнянні з Darknet-19, що використовується в YOLOv2.

ResNet-подібні залишкові блоки: Darknet-53 включає залишкові блоки (residual blocks), що дозволяє мережі тренуватися глибше, зменшуючи проблему зникнення градієнта.

Детекція на різних масштабах YOLOv3 виконує детекцію об'єктів на трьох різних масштабах. Це дозволяє моделі краще розпізнавати об'єкти різного розміру. Кожен масштаб відповідає різному рівню особливостей мережі.

Розширене використання анкорів (Anchor Boxes), як і в YOLOv2, YOLOv3 використовує анкори для передбачення bounding box'ів. Однак в цій версії кількість анкорів збільшено, що дозволяє моделі краще справлятися з об'єктами різних розмірів і форм.

У YOLOv3 використовується нова функція активації під назвою Leaky ReLU [29], яка допомагає мережі краще тренуватися, зменшуючи проблему зникнення градієнта.

Логітні значення для передбачень використовує замість передбачення ймовірностей класів напряму, YOLOv3 використовує логітні значення, що робить процес навчання більш стабільним і точним.

Всю систему можна поділити на два основних компоненти: екстрактор ознак і детектор (рис 3.5), причому обидва працюють на багатьох масштабах [13]. Коли з'являється нове зображення, воно спочатку проходить через екстрактор ознак, який отримує вбудовування об'єктів на трьох (або більше) різних масштабах.

Потім ці ознаки подаються в три (або більше) гілки детектора, щоб отримати обмежувальні рамки та інформацію про класи.

Рисунок 3.5 – Архітектурна мережа

Екстрактор ознак, який використовується в YOLOv3, називається Darknet-53. Можливо, ви знайомі з попередньою версією Darknet від YOLOv1, яка мала лише 19 шарів. Відтоді мережі класифікації зображень значно розвинулися від просто глибоких стеків шарів. ResNet запропонував ідею пропускних з'єднань для допомоги в поширенні активацій через глибші шари без зменшення градієнта. Darknet-53 запозичує цю ідею та успішно розширює мережу з 19 до 53 шарів (рис 2.10).

Це легко зрозуміти: кожен прямокутник на діаграмі представляє залишковий блок. Вся мережа складається з кількох таких блоків з деякими проміжними шарами Conv для зменшення розмірності. Всередині кожного блоку є структура "вузького місця" (1 на 1, потім 3 на 3) плюс пропускне з'єднання. Якщо мета полягає в класифікації кількох класів, як у випадку з

ImageNet, додається середнє об'єднання, 1000 повнозв'язних шарів і активація softmax.

Однак, для виявлення об'єктів ми не включаємо класифікаційний блок. Натомість, до екстрактора ознак додається блок "виявлення". Оскільки YOLOv3 розроблений як багатомасштабний детектор, нам також потрібні ознаки з різних масштабів. Таким чином, ознаки з останніх трьох залишкових блоків використовуються для подальшого виявлення. На діаграмі (рис 3.6) припускається, що вхідні дані мають розмір 416x416, тому три вектори масштабу будуть 52x52, 26x26 і 13x13. Важливо зазначити, що якщо розмір вхідного зображення зміниться, розміри вихідних векторів також зміняться.

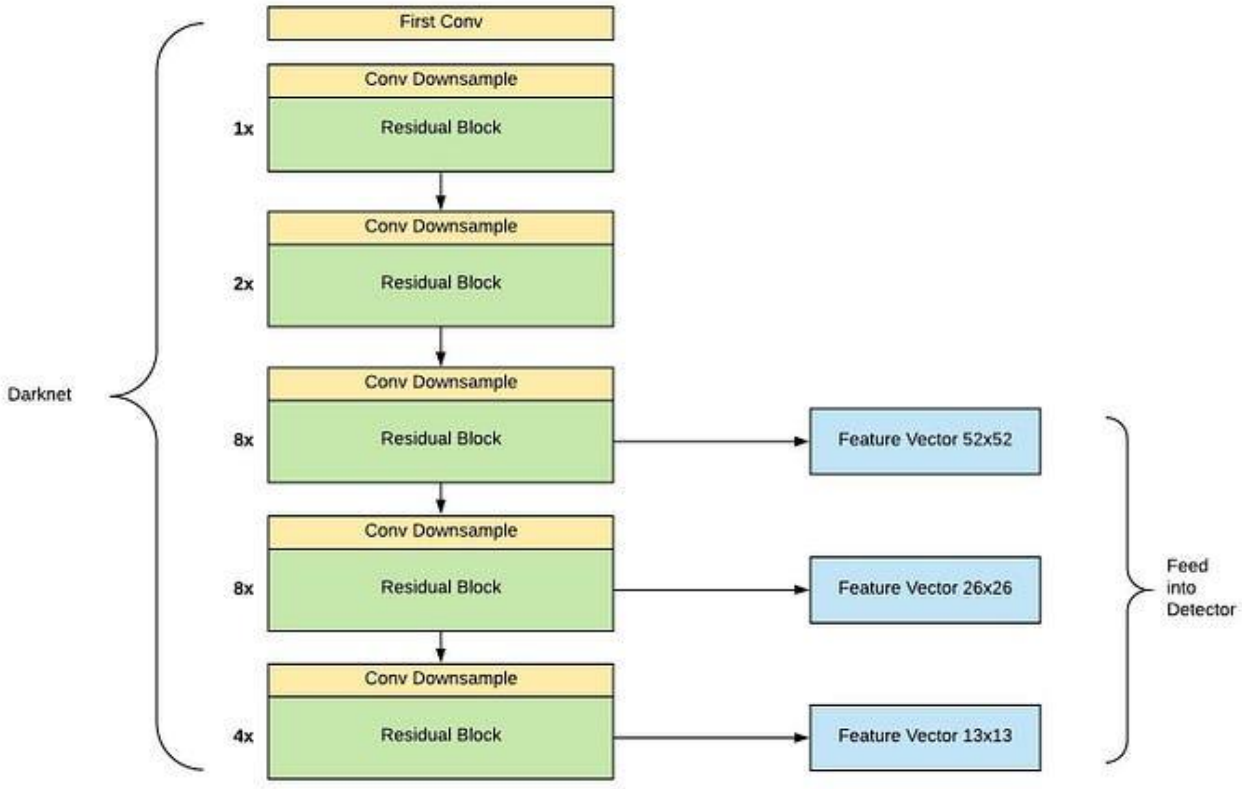

Рисунок 3.6 – Діаграма з трьома векторами масштабу

YOLOv3 є значним покращенням у порівнянні з попередніми версіями YOLO, зокрема за рахунок нової архітектури Darknet-53, багатомасштабної детекції, розширеного використання анкорів і покращеної функції активації.

Ці інновації дозволяють досягти вищої точності і зберегти високу швидкість обробки, що робить YOLOv3 однією з провідних моделей для детекції об'єктів у реальному часі.

Одним з ключових аспектів YOLOv3 є використання багатомасштабної детекції, яка дозволяє моделі ефективніше розпізнавати об'єкти різних розмірів. Це досягається завдяки обробці зображення на трьох різних масштабах, що значно підвищує точність виявлення дрібних та великих об'єктів. Такий підхід дозволяє моделі бути більш універсальною та адаптивною до різних типів даних і сцен. Розширене використання анкорів (anchor boxes) у YOLOv3 також сприяє підвищенню точності [21].

Останнім компонентом у системі виявлення об'єктів є постпроцесор. Зазвичай постобробка включає тривіальні завдання, такі як перетворення машинного ідентифікатора класу на читабельний текст. Однак, у виявленні об'єктів є ще один важливий крок для отримання остаточних результатів, зрозумілих для людини, – немаксимальне придушення (Non-Maximum Suppression, NMS).

Втрата об'єктивності - коли хибна пропозиція має великий збіг із обґрунтованою істиною, ми не караємо її за допомогою noobj_loss. Це сприяє тому, що мережа прогнозує близькі результати, полегшуючи її навчання. Крім того, хоча це не використовується в YOLO, при використанні методу ковзного вікна, кілька вікон можуть виявити один і той самий об'єкт (рис 3.7). Щоб усунути ці повторювані результати, було розроблено алгоритм під назвою немаксимальне придушення (NMS).

Рисунок 3.7 – Приклад роботи алгоритму NMS

Ідея немаксимального придушення (NMS) досить проста. Спочатку вибирається поле з найвищою впевненістю, додається до кінцевого результату, а потім усуваються всі інші поля, які мають значення IOU (Intersection Over Union) вище певного порогу з цим найкращим полем. Далі вибирається наступне поле з найбільшою впевненістю серед решти та повторюється той самий процес, поки не залишиться жодного поля. У коді, особливо в TensorFlow [18], де часто потрібно явно задавати форми, зазвичай визначається максимальна кількість виявлень і процес зупиняється, якщо це число досягнуто. У YOLOv3 класифікація більше не є взаємовиключною, і один виявлений об'єкт може мати більше одного правильного класу. Проте деякі існуючі реалізації NMS можуть не враховувати це, тому слід бути обережним при їх використанні. Невзаємовиключна класифікація в YOLOv3 дозволяє моделі одночасно передбачати декілька класів для одного об'єкта. Це особливо корисно у випадках, коли об'єкти можуть належати до кількох категорій одночасно, наприклад, автомобіль може бути класифікований як транспортний засіб і як об'єкт, що рухається.

Layer	Filters size	Repeat	Output size
Image			416 × 416
Conv	32 3 × 3/1	1	416 × 416
Conv	64 3 × 3/2	1	208 × 208
Conv Conv Residual	32 1 × 1/1 64 3 × 3/1	Conv Conv Residual × 1	208 × 208 208 × 208 208 × 208
Conv	128 3 × 3/2	1	104 × 104
Conv Conv Residual	64 1 × 1/1 128 3 × 3/1	Conv Conv Residual × 2	104 × 104 104 × 104 104 × 104
Conv	256 3 × 3/2	1	52 × 52
Conv Conv Residual	128 1 × 1/1 256 3 × 3/1	Conv Conv Residual × 8	52 × 52 52 × 52 52 × 52
Conv	512 3 × 3/2	1	26 × 26
Conv Conv Residual	256 1 × 1/1 512 3 × 3/1	Conv Conv Residual × 8	26 × 26 26 × 26 26 × 26
Conv	1024 3 × 3/2	1	13 × 13
Conv Conv Residual	512 1 × 1/1 1024 3 × 3/1	Conv Conv Residual × 4	13 × 13 13 × 13 13 × 13

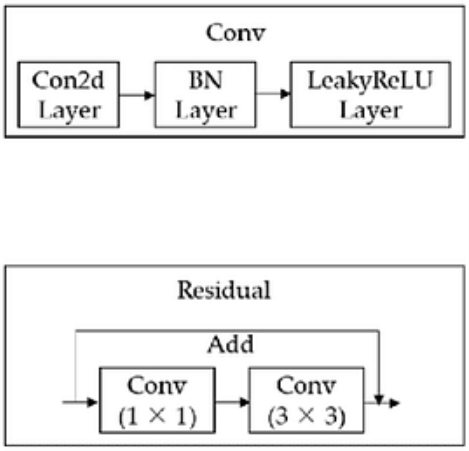

Рисунок 3.8 – Архітектурна схема YOLOv3

3.4 YOLOv4

YOLOv4, розроблений Алексеєм Бочарським (Alexey Bochkovskiy) та співавторами, є значним оновленням до попередніх версій YOLO. Основні вдосконалення включають підвищену точність і продуктивність завдяки інтеграції різних передових технік з області глибокого навчання.

В YOLOv4 використовується кілька інноваційних функцій, які спільно оптимізують його продуктивність. Серед них: зважені резидуальні зв'язки (Weighted-Residual-Connections, WRC), перехресні часткові зв'язки (Cross-Stage-Partial-connections, CSP), перехресна міні-пакетна нормалізація (CmBN), самоадверсаріальне навчання (SAT), активація Mish, збільшення даних Mosaic, регуляризація DropBlock і втрата CIoU. Поєднання цих функцій дозволяє досягти найсучасніших результатів [22].

Типовий детектор об'єктів складається з кількох частин: вхід, основна мережа (backbone), "шия" (neck) та "голова" (head). Основна мережа YOLOv4 попередньо навчена на ImageNet і використовується для передбачення класів і обмежувальних рамок об'єктів. Основна мережа може включати кілька моделей, таких як VGG, ResNet, ResNeXt або DenseNet. Частина "шиї" детектора використовується для збору карт ознак з різних етапів і зазвичай містить кілька висхідних та низхідних шляхів. "Голова" використовується для остаточного виявлення та класифікації об'єктів.

Схема архітектури YOLOv4 (рис 3.9) показує складну мережеву конструкцію YOLOv4, включаючи компоненти основної мережі (backbone), "шиї" (neck) та "голови" (head), а також їх взаємозв'язані шари для оптимального виявлення об'єктів у режимі реального часу. Ефективність YOLOv4 підтверджується її здатністю досягати високої точності при обробці зображень у реальному часі. Це робить її надзвичайно корисною для широкого спектра застосувань. Це робить YOLOv4 однією з найкращих моделей для задач детекції об'єктів у реальному часі. Її застосування охоплює широкий спектр галузей, таких як системи відеоспостереження, автономне водіння,

медична діагностика та інші області, де необхідна швидка та точна детекція об'єктів.

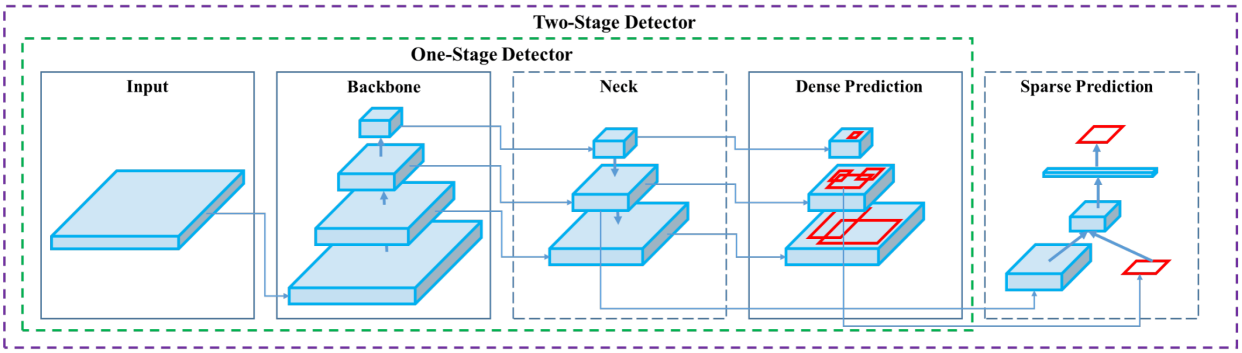

Рисунок 3.9 – Схема архітектури YOLOv4

CSPDarknet53 є основною мережею (backbone) для YOLOv4, створеною для екстракції ознак зображень. Вона базується на Darknet-53, але включає в себе технологію CSP (Cross Stage Partial), яка підвищує продуктивність і ефективність моделі. CSPDarknet53 розділяє базову мережу на дві частини і інтегрує їх у кінець, що дозволяє зменшити кількість обчислень і покращити передачу градієнтів через мережу [23]. Це робить CSPDarknet53 швидшим та більш точним порівняно з попередніми версіями.

Архітектурні особливості:

– Cross Stage Partial connections (CSP);
– залишкові блоки (Residual Blocks);
– batch normalization;
– mish activation.

Cross Stage Partial Connections (CSP) використовує CSP-з'єднання, що розділяють потік даних на дві частини та з'єднують їх на більш пізніх етапах. Це дозволяє зменшити обчислювальні витрати, зберігаючи при цьому високу точність. CSP-з'єднання допомагають уникнути проблеми з перевантаженням обчислень і зменшують залежність між шарами.

Залишкові блоки (Residual Blocks) як і Darknet53, CSPDarknet53 використовує залишкові блоки, які дозволяють зберігати інформацію через

глибокі мережі, допомагаючи уникнути проблеми зникнення градієнта. Кожен залишковий блок складається з кількох згорток, об'єднаних пропускними з'єднаннями (skip connections).

Batch Normalization CSPDarknet53 застосовує нормалізацію міні-пакетів (batch normalization) після кожного згорткового шару, що сприяє стабілізації та прискоренню процесу навчання.

Mish Activation CSPDarknet53 використовує функцію активації Mish, яка показала кращу продуктивність порівняно з ReLU та Leaky ReLU, забезпечуючи плавнішу оптимізацію та кращу передану інформацію.

Переваги CSPDarknet53:

- точність;
- зменшені обчислювальні витрати;
- стабільність навчання;
- глибоке вилучення ознак.

Покращена точність завдяки використанню CSP-з'єднань, CSPDarknet53 забезпечує кращу точність за рахунок більш ефективного вилучення ознак.

Зменшені обчислювальні витрати - CSP-з'єднання дозволяють зменшити обчислювальні витрати, зберігаючи високу продуктивність моделі.

Стабільність навчання - Batch normalization та Mish activation сприяють стабільності та швидкості навчання.

Глибока архітектура CSPDarknet53 (рис 3.8) включає 53 згорткових шари, які дозволяють моделі витягувати складні ознаки з вхідних зображень. Це глибока архітектура, що забезпечує високу здатність до узагальнення та дозволяє обробляти складні і різноманітні зображення. дозволяє моделі ефективно витягувати складні ознаки з зображень, що є критичним для високоякісного виявлення об’єктів. Таким чином, глибока архітектура CSPDarknet53 є важливим компонентом YOLOv4, який забезпечує високу ефективність та точність у задачах детекції об'єктів, роблячи модель однією з найкращих для застосувань у реальному часі.

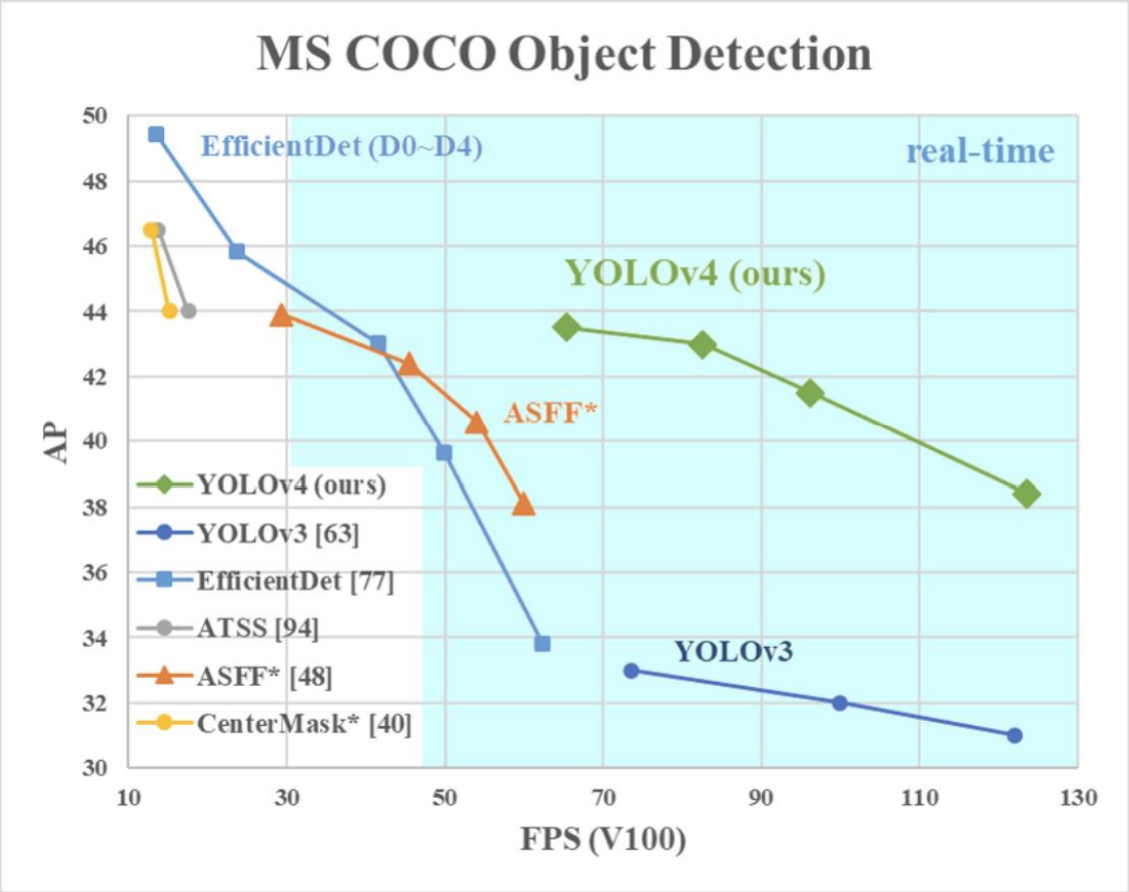

Рисунок 3.10 – Діаграма CSPDarknet53

Bag of Freebies (BoF) включає методи та техніки, які покращують точність моделі без збільшення часу інферування. Деякі з цих методів:

– data augmentation;
– mosaic augmentation;
– DropBlock.

Data Augmentation (Розширення даних) - використання різних технік для створення різноманітних варіацій зображень тренувального набору даних, що допомагає моделі краще узагальнювати на нові дані.

Mosaic Augmentation - техніка, що дозволяє поєднувати декілька зображень в одне, надаючи моделі можливість бачити об'єкти з різних перспектив та в різних умовах.

DropBlock - регуляризаційна техніка, яка випадковим чином "вимикає" блоки активацій під час тренування, що допомагає запобігти перенавчанню.

Bag of Specials (BoS) включає техніки та компоненти, які додаються до моделі для підвищення її продуктивності. Деякі з цих технік:

– mish activation function;
– Spatial Pyramid Pooling (SPP);
– Path Aggregation Network (PANet).

Mish Activation Function - нелінійна функція активації, яка забезпечує плавніші градієнти, покращуючи процес навчання та узагальнення моделі.

Spatial Pyramid Pooling (SPP) - техніка, що дозволяє витягати ознаки на різних масштабах, об'єднуючи інформацію з декількох рівнів деталізації.

Path Aggregation Network (PANet) - архітектурне вдосконалення, яке поліпшує агрегацію ознак на різних рівнях мережі, підвищуючи точність виявлення об'єктів.

YOLO Head — це останній компонент архітектури YOLO, який безпосередньо відповідає за передбачення обмежувальних рамок (bounding boxes), класів об'єктів та їхні ймовірності. У YOLOv4, як і в попередніх версіях, YOLO Head використовує вихідні ознаки, витягнуті з основної мережі (backbone), для генерування передбачень. Цей компонент працює на декількох масштабах, що дозволяє моделі ефективно розпізнавати об'єкти різних розмірів.

Основні функції YOLO Head:

– bounding box regression;
– class prediction;
– confidence score.

Bounding Box Regression - прогнозує координати обмежувальних рамок для об'єктів у зображенні.

Class Prediction - визначає клас об'єкта для кожної обмежувальної рамки.

Confidence Score - оцінює ймовірність того, що об'єкт дійсно знаходиться в передбаченій рамці.

Post-processing: Non-Maximum Suppression (NMS) - постпроцесінг є важливою складовою системи виявлення об'єктів, яка допомагає очистити та

відфільтрувати результати, щоб зменшити кількість помилкових передбачень та дублювання об'єктів. Основною технікою постпроцесінгу в YOLO є Non-Maximum Suppression (NMS).

NMS виконується після того, як YOLO Head [28] видає початкові передбачення обмежувальних рамок та класів об'єктів. Основна ідея NMS полягає в тому, щоб залишити лише найбільш впевнені передбачення для кожного об'єкта та усунути надлишкові рамки, які перекриваються з найвпевненішими передбаченнями.

Основні кроки NMS:

– вибір найбільш впевненої рамки;
– видалення перекриттів;
– повторення процесу.

Вибір найбільш впевненої рамки - вибирається рамка з найвищою впевненістю (confidence score).

Видалення перекриттів усіх інших рамок, які мають високий коефіцієнт перетину з цією рамкою (Intersection over Union, IoU) вище певного порогу, видаляються з розгляду.

Повторення процесу повторюється з рештою рамок, поки всі рамки з високою впевненістю не будуть розглянуті.

NMS дозволяє системі виявлення об'єктів видавати чисті та точні результати, уникати надмірних передбачень одного й того ж об'єкта та забезпечувати краще візуальне представлення для подальшого використання. Завдяки цим етапам і технікам, таким як YOLO Head і NMS, модель YOLO забезпечує високу точність і ефективність виявлення об'єктів у зображеннях, зберігаючи при цьому високу швидкість роботи. Крім того, модель YOLO (You Only Look Once) використовує єдину нейронну мережу, яка поділяє зображення на сітку і одночасно прогнозує межі об'єктів та їх класифікацію. Це дозволяє уникнути багатократного сканування зображення, як це робиться в традиційних підходах до виявлення об'єктів, що значно підвищує продуктивність і зменшує затрати часу на обробку.

3.5 YOLOv10

YOLOv10, створений на основі Python пакету Ultralytics дослідниками з університету Цінхуа, представляє новий підхід до виявлення об'єктів в режимі реального часу, усуваючи недоліки постобробки та архітектури моделі, виявлені у попередніх версіях YOLO. YOLOv10 досягає найсучасніших показників продуктивності при значному зменшенні обчислювальних навантажень.

Архітектура YOLOv10 базується на сильних сторонах попередніх моделей YOLO та впроваджує кілька ключових інновацій. Модель складається з наступних компонентів:

- backbone;
- шия;
- one-to-many head;
- one-to-one head.

Backbone використовується для вилучення ознак використовується покращена версія CSPNet (Cross Stage Partial Network), яка покращує градієнтний потік і зменшує обчислювальну надмірність.

Шия об'єднує ознаки з різних масштабів і передає їх у голову. Вона включає шари PAN (Path Aggregation Network) для ефективного злиття ознак різного масштабу.

One-to-Many Head генерує кілька прогнозів для кожного об'єкта під час навчання, що забезпечує багаті супервізорні сигнали та підвищує точність навчання.

One-to-One Head генерує найкращий прогноз для кожного об'єкта під час інференсу, що усуває необхідність у NMS (рис 3.9), зменшуючи затримку і підвищуючи ефективність.

Основні характеристики:

- навчання без NMS;
- холістичний дизайн моделі;

– розширені можливості моделі.

Навчання без NMS використання послідовних подвійних призначень усуває необхідність у NMS, зменшуючи затримку при інференсі.

Холістичний дизайн моделі всеосяжна оптимізація різних компонентів з точки зору ефективності та точності, включаючи полегшені класифікаційні головки, просторово-канальну декорельовану вибірку вниз і ранжований дизайн блоків.

Розширені можливості моделі включає згортки з великими ядрами та модулі часткового самонавчання, що підвищують продуктивність без значних обчислювальних витрат (рис 3.11).

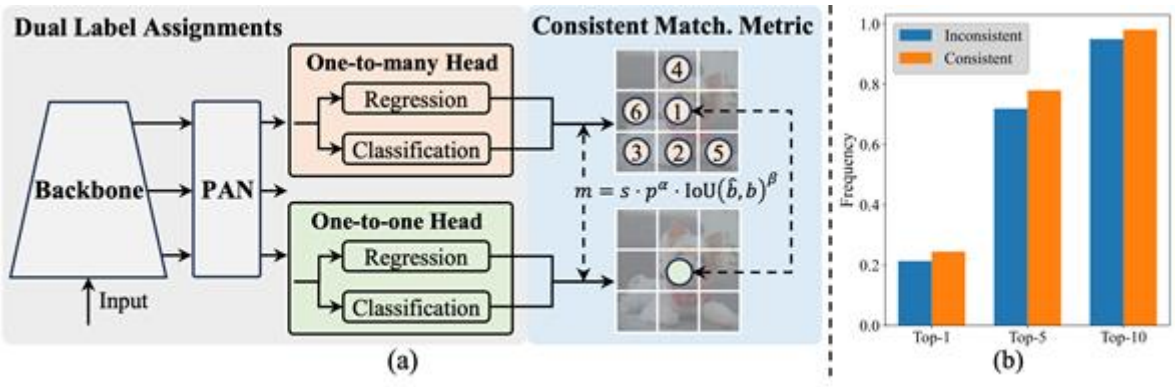

Рисунок 3.11 – Подвійні завдання для навчання без NMS

YOLOv10 перевершує попередні версії YOLO та інші сучасні моделі за точністю та ефективністю. YOLOv10-S у 1,8 рази швидший за RT-DETR-R18 з аналогічним AP на наборі даних COCO. YOLOv10-B має на 46% меншу затримку та на 25% менше параметрів, ніж YOLOv9-C з тією ж продуктивністю. Крім того, YOLOv10-L демонструє значно покращену точність порівняно з YOLOv9-L за рахунок ефективнішого використання ресурсів. Наприклад, на наборі даних ImageNet YOLOv10-L показує збільшення точності на 2% у порівнянні з YOLOv9-L при збереженні продуктивності. Це вдосконалення досягнуто завдяки оптимізації архітектури

моделі та впровадженню нових методів навчання, таких як поліпшена обробка вхідних зображень та вдосконалені алгоритми тренування.

Таблиця 3.1 – Характеристики та параметри детекторів

Модель	**Розмір входу**	**APval**	**FLOPs (G)**	**Latency (ms)**
YOLOv10-N	640	38,5	6,7	1,89
YOLOv10-S	640	46,3	21,6	2,49
YOLOv10-M	640	51,1	59,1	4,74
YOLOv10-B	640	52,5	92,0	5,74
YOLOv10-L	640	53,2	120,3	7,28
YOLOv10-X	640	54,4	160,4	10,70

Порівняння з іншими сучасними детекторами:

– YOLOv10-S / X у 1,8× / 1,3× швидший за RT-DETR-R18 / R101 при схожій точності;

– у YOLOv10-B на 25% менше параметрів і на 46% менша затримка, ніж у YOLOv9-C при тій самій точності;

– YOLOv10-L / X перевершує YOLOv8-L / X на 0,3 AP / 0,5 AP з 1,8× / 2,3× меншим числом параметрів.

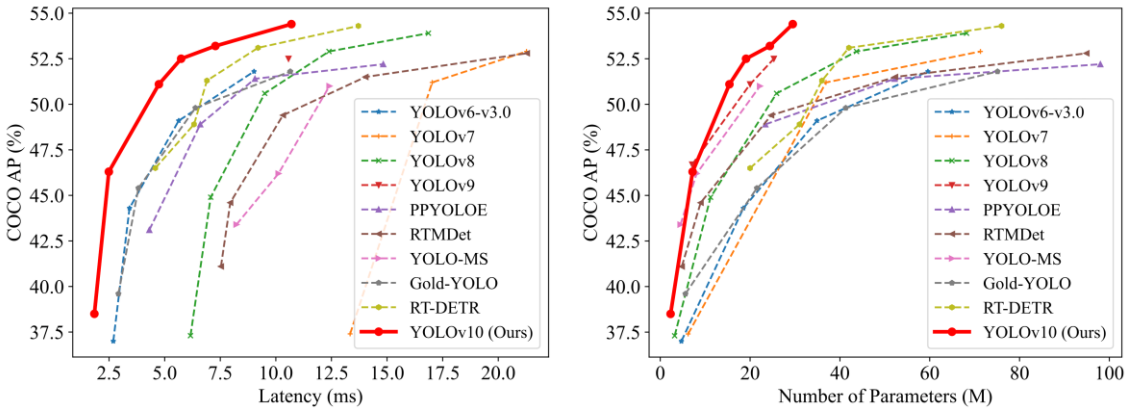

Рисунок 3.12 – Порівняння детекторів

Детальне порівняння варіантів YOLOv10 з іншими сучасними моделями. YOLOv10-S з RT-DETR-R18 та YOLOv10-B з YOLOv9-C підтверджують значний технічний прогрес у швидкості обробки [24].

Таблиця 3.2 – Порівняння YOLOv10 з сучасними моделями

Модель	**Parasm (M)**	**FLOPs (G)**	**APval (%)**	**Latency (ms)**	**Latency (forward) (ms)**
YOLOv6-3.0-N	4,7	11,4	37,0	2,69	1,76
Gold-YOLO-N	5,6	12,1	39,6	2,92	1,82
YOLOv8-N	3,2	8,7	37,3	6,16	1,77
YOLOv10-N	2,3	6,7	39,5	1,84	1,79
YOLOv6-3.0-S	18,5	45,3	44,3	3,42	2,35
Золото -YOLO-S	21,5	46,0	45,4	3,82	2,73
YOLOv8-S	11,2	28,6	44,9	7,07	2,33
YOLOv10-S	7,2	21,6	46,8	2,49	2,39
RT-DETR-R18	20,0	60,0	46,5	4,58	4,49
YOLOv6-3.0-M	34,9	85,8	49,1	5,63	4,56
Золото -YOLOM	41,3	87,5	49,8	6,38	5,45
YOLOv8-M	25,9	78,9	50,6	9,50	5,09
YOLOv10-M	15,4	59,1	51,3	4,74	4,63
YOLOv6-3.0-L	59,6	150,7	51,8	9,02	7,90
Золото -YOLO-L	75,1	151,7	51,8	10,65	9,78
YOLOv8-L	43,7	165,2	52,9	12,39	8,06
RT-DETR-R50	42,0	136,0	53,1	9,20	9,07
YOLOv10-L	24,4	120,3	53,4	7,28	7,21
YOLOv8-X	68,2	257,8	53,9	16,86	12,83
RT-DETR-R101	76,0	259,0	54,3	13,71	13,58
YOLOv10-X	29,5	160,4	54,4	10,70	10,60

YOLOv10 використовує подвійне призначення міток, поєднуючи стратегії "один до багатьох" і "один до одного" під час навчання, щоб забезпечити багатий контроль і ефективне наскрізне розгортання. Метрика узгодженого відповідності вирівнює контроль між обома стратегіями, покращуючи якість прогнозів під час виведення.

Підвищення ефективності:

– полегшена класифікаційна голівка;
– просторово-канальне розділення;
– Rank-Guided block design.

Полегшена класифікаційна голівка - зменшення обчислювальних витрат на класифікаційну голівку за рахунок використання роздільних згорток по глибині.

Просторово-канальне розділене зниження вибірки - розділення просторового зменшення та канальної модуляції для мінімізації втрати інформації та обчислювальних витрат.

Rank-Guided Block Design - адаптація дизайну блоку на основі притаманної йому надмірності етапів, забезпечуючи оптимальне використання параметрів.

Підвищення точності:

– згортка з великим ядром;
– Partial Self-Attention (PSA).

Згортка з великим ядром збільшує рецептивне поле для покращення здатності до вилучення ознак, що сприяє покращенню здатності моделі до вилучення ознак. Наприклад, в порівнянні з попередніми версіями, застосування такої згортки дозволяє моделі краще адаптуватися до різних контекстів об'єктів на зображенні та ефективніше виділяти важливі ознаки.

Partial Self-Attention (PSA) - включає модулі самовнушення для покращення навчання глобальних уявлень з мінімальними накладними витратами. YOLOv10 пройшов всебічне тестування на стандартних бенчмарках, таких як COCO, демонструючи чудову продуктивність та

ефективність. Модель досягає найсучасніших результатів у різних варіантах, показуючи значне покращення затримки та точності порівняно з попередніми версіями та іншими сучасними детекторами.

Моделі YOLOv10 доступні у різних варіантах для задоволення різноманітних потреб у застосуванні:

- YOLOv10-N;
- YOLOv10-S;
- YOLOv10-M;
- YOLOv10-B;
- YOLOv10-L.

YOLOv10-N - нано-версія для обмежених ресурсів. YOLOv10-S - компактна версія, що забалансована між швидкістю та точністю. YOLOv10-M середня версія для універсального використання. YOLOv10-B збалансована версія з розширеною шириною для більшої точності. YOLOv10-L велика версія для підвищення точності за рахунок збільшення обчислювальних ресурсів. YOLOv10-X: Екстра-велика версія для максимальної точності та продуктивності.

ВИСНОВКИ

У цій роботі було досліджено детекцію множин об'єктів на зображеннях з використанням алгоритму YOLO (You Only Look Once). Вивчення особливостей різних версій YOLO, таких як YOLOv1, YOLOv2, YOLOv3, YOLOv4, та найновіших моделей, продемонструвало еволюцію архітектури та покращення в точності та швидкості обробки.

Основні переваги YOLO включають його здатність до реального часу обробки, високу точність виявлення та можливість детекції об'єктів на різних масштабах. Завдяки інноваційним підходам, таким як використання CSPDarknet53 у YOLOv4 та вдосконалення головки класифікації в YOLOv10, алгоритм забезпечує ефективне виявлення об'єктів з мінімальними обчислювальними витратами.

Розглянуті методи навчання без використання NMS (немаксимального придушення) у YOLOv10, подвійні призначення міток, а також холістичний дизайн моделей показали значний прогрес у покращенні якості прогнозів та зниженні затримок. Ці підходи дозволили досягти найсучасніших результатів на стандартних бенчмарках, таких як COCO, підтвердивши ефективність та точність YOLO у виявленні множин об'єктів на зображеннях.

Таким чином, YOLO залишається одним з найпотужніших та найефективніших інструментів для детекції об'єктів, що робить його незамінним у багатьох практичних застосуваннях, включаючи автономні транспортні засоби, системи безпеки та аналіз відео. Подальші дослідження та вдосконалення цієї архітектури можуть призвести до ще більш вражаючих результатів у сфері комп'ютерного зору.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Gorokhovatskyi V., Tvoroshenko I., and Yakovleva O. (2024) Transforming image descriptions as a set of descriptors to construct classification features, Indonesian Journal of Electrical Engineering and Computer Science, 33(1), pp. 113-125.

2. Lyashenko, V., Kobylin, O., & Baranchykov, Y. (2018, October). Ideology of Image Processing in Infocommunication Systems. In 2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T) (pp. 47-50). IEEE.

3. Kobylin O., Vyskrebentseva S., & Petrova R. (2019). Обробка даних, що містять пропуски в задачах кластеризації. Системи управління, навігації та зв'язку. Збірник наукових праць, 5(57).

4. Mustafa, S. K., Kopot, M., Ahmad, M. A., Lyubchenko, V., & Lyashenko, V. (2020). Interesting Applications of Mobile Robotic Motion by using Control Algorithms. International Journal, 9(3).

5. Kuzomin, O., & Lyashenko, V. (2022). Key Elements of a Specialized Complex for Solving Modeling Problems.

6. Yousef Ibrahim Daradkeh & Irina Tvoroshenko (2020), Technologies for Making Reliable Decisions on a Variety of Effective Factors using Fuzzy Logic. International Journal of Advanced Computer Science and Applications(IJACSA), pp. 11(5),

7. Tvoroshenko, I., & Koriakin, I. (2021). Analysis of methods for detecting and classifying the likeness of human features.

8. Dubnitskiy, V., Kobylin, A., Kobylin, O., Kushneruk, Y., & Khodyrev, A. (2023). Обчислення значень функції Харрінгтона (функції бажаності) при інтервальному визначенні її аргументів. Advanced Information Systems, 7(1),71-81.

9. Titova O., Vysotskyi D. (2021) The development of a subsystem for palliative patients information support. Trends in science and practice of today. Abstracts of V International Scientific and Practical Conference. Ankara, Turkey. Pp. 401–404.

10. Kinoshenko D., Kobylin O., Mashtalir S., & Stolbovyi M. (2019, March). Metric video retrieval speedup by irrelevant data elimination. In Eleventh International Conference on Machine Vision (ICMV 2018) (Vol. 11041, pp. 176-183). SPIE.

11. Помазан В. (2022) Аналіз технологій ідентифікації, розпізнавання та оброблення емоцій людини на зображеннях, Abstracts of XVIII International Scientific and Practical Conference «Advancing in research, practice and education» (May 10 – 13, 2022). Florence, Italy, pp. 645-648.

12. Lyashenko, V., Kobylin, O., Ryazantsev, O., Ryazantsev, I., Barbaruk, V., & Zhychenko, Y. (2020). General Ideology of Analysis Digital Medical Images in RGB Format.

13. Олешко, Т. І., & Бахорчук, В. (2020). Моделювання бізнес-процесів за допомогою діаграми випадків використання.

14. Розпізнавання об'єктів в режимі реального часу на iOS з допомогою YOLOv3. URL: https://devzone.org.ua/post/rozpiznavannia-obyektiv-v-rezymi-realnoho-chasu-na-ios-z-dopomohoiu-yolov3 (дата звернення 21.05.2024).

15. Concept of YOLOv1:The Evolution of Real-Time Object Detection URL: https://medium.com/@sachinsoni600517/concept-of-yolov1-the-evolution-of-real-time-object-detection-d773770ef773 (дата звернення 23.05.2024).

16. YOLOv1 URL: https://paperswithcode.com/method/yolov1 (дата звернення 22.05.2024).

17. YOLO v2 Comprehensive Tutorial: Building on YOLO v1 Mistakes URL: https://medium.com/@sachinsoni600517/yolo-v2-comprehensive-tutorial-building-on-yolo-v1-mistakes-aa7912292c1a (дата звернення 25.05.2024).

18. YOLOv3 - Ultralytics YOLO Docs URL: https://docs.ultralytics.com/ru/models/yolov3/#overview (дата звернення 24.05.2024).

19. YOLO: Real-Time Object Detection URL: https://pjreddie.com/darknet/yolo/ (дата звернення 25.05.2024).

20. YOLOv3 URL: https://pjreddie.com/media/files/papers/YOLOv3.pdf (дата звернення 25.05.2024).

21. Dive Really Deep into YOLO v3: A Beginner’s Guide URL: https://towardsdatascience.com/dive-really-deep-into-yolo-v3-a-beginners-guide-9e3d2666280e (дата звернення 26.05.2024).

22. YOLOv4 - Ultralytics YOLO Docs URL: https://docs.ultralytics.com/models/yolov4/ (дата звернення 26.05.2024).

23. CSPDarknet53 Explained | Papers With Code URL: https://paperswithcode.com/method/cspdarknet53 (дата звернення 28.05.2024).

24. YOLOv10 - Ultralytics YOLO Docs URL: https://docs.ultralytics.com/ru/models/yolov10/#comparisons (дата звернення 30.05.2024).

25. YOLOv8 - Ultralytics YOLO Docs URL: https://docs.ultralytics.com/models/yolov8/ (дата звернення 26.05.2024).

26. Meituan YOLOv6 - Ultralytics YOLO Docs URL: https://docs.ultralytics.com/models/yolov6/ (дата звернення 26.05.2024).

27. YOLOv7: Trainable Bag-of-Freebies - Ultralytics YOLO Docs URL: https://docs.ultralytics.com/models/yolov7/ (дата звернення 26.05.2024).

28. YOLOv9: A Leap Forward in Object Detection Technology - Ultralytics YOLO Docs URL: https://docs.ultralytics.com/models/yolov9/ (дата звернення 30.05.2024).

29. Review — Mish: A Self Regularized Non-Monotonic Activation Function URL: https://medium.com/@sh-tsang/review-mish-a-self-regularized-non-monotonic-activation-function-a7afe19b4af7 (дата звернення 30.05.2024).

30. YOLO — Intuitively and Exhaustively Explained URL: https://towardsdatascience.com/yolo-intuitively-and-exhaustively-explained-83143925c7a9 (дата звернення 30.05.2024).