Міністерство освіти і науки України

Національний університет «Києво-Могилянська академія»

Факультет інформатики

Кафедра математики

Курсова робота

освітній ступінь – магістр

на тему: «**МЕТОД ДЕТЕКЦІЇ ОБ'ЄКТІВ ВИКОРИСТОВУЮЧИ ІЄРАРХІЧНЕ ДЕРЕВО КЛАСІВ ОБ'ЄКТІВ**»

Виконав: студент 1-го року навчання,

Освітньої програми «Прикладна математика», 113

Панасюк Роман Олегович

Керівник

Кандидат фізико-математичних наук
Старший викладач

Швай Надія Олександрівна

Курсова робота захищена
з оцінкою _____

Секретар ЕК _____
«____» _____ 2023 р.

Київ – 2023

ЗМІСТ

ВСТУП

Задача детекції дуже актуальна в наш час задача, адже з розвитком технологій застосування методів розв’язання цієї задачі знаходить все більше практичних застосувань, таких як розпізнавання лиць, контроль якості на виробництва, підрахунок кількості людей в натовпі, та багато інших.

Мета дослідження полягає у порівнянні існуючих методів для задачі детекції, виявленні сильних та слабких сторін методу детекції об’єктів використовуючи ієрархічне дерево класів у порівнянні з іншими методами.

Об’єкт дослідження - задача детекції

Методи дослідження - метод детекції об’єктів використовуючи ієрархічне дерево класів, класичні моделі для задачі детекції: SSD, R-CNN, YOLO

Отримані результати дозволяють зрозуміти слабкі та сильні сторони існуючих методів для задачі детекції, особливу увагу надаючи методу детекції об’єктів використовуючи ієрархічне дерево класів, як тому методу, що дозволяє здійснювати детекцію для дуже велику кількість класів. В подальшому було б доцільним поівняти розглянутий метод з найсучаснішими методами для цієї задачі.

ПЕРЕЛІК ПРИЙНЯТИХ СКОРОЧЕНЬ

МН - машинне навчання

ШІ - штучний інтелект

КЗ - комп’ютерний зір

ГН - глибока мережа

НМ - нейронна мережа

ЗАДАЧА ДЕТЕКЦІЇ ОБ'ЄКТІВ

Загальні відомості

Комп'ютерний зір (КЗ) це галузь Штучного інтелекту (ШІ), що дозволяє комп'ютерним системам виконувати задачі з цифровими зображеннями, відео, або будь-якими іншими видами візуальної або звукової інформації. Виходячи з назви вже стає зрозуміло, що суть завдань, що стоять перед системами комп'ютерного зору так чи інакше пов'язані з людським зором. Організм людини сам по собі є дуже складною біологічною системою, яка складається з великої кількості підсистем, у вигляді людських органів. З точки зору розуміння - задача кожного органу досить проста і зрозуміла, проте за цією простотою стоять безліч складних біологічних та хімічних процесів. Однією з таких систем є людські очі - вони щодня займаються безперервним процесом обробки вхідних даних, разом з мозком, який безпосередньо керує цим процесом. На перший погляд прості задачі, які ми виконуємо навіть не задумуючись, з точки зору реалізації за допомогою комп'ютера є дуже складними і як доказ тому слугує той факт, що деякі з таких задач були розв'язані тільки нещодавно. Загалом ця галузь дуже стрімко розвивається, і протягом останніх десятиліть було здійснено чимало відкриттів, які вже тут і зараз впроваджуються в різні сфери життя.

Задачі комп'ютерного зору

Серед безлічі задач, що розв'язуються за допомогою КЗ існують деякі класичні задачі: класифікація зображень, детекція об'єктів, семантична сегментація, сутнісна сегментація. Для загального розуміння розглянемо в чому полягає кожна з цих задач.

Класифікація зображень полягає у визначенні класу до якого належить об'єкт, що знаходиться на зображенні. Як приклад таких класів можуть бути собаки, коти, дерева і так далі. По суті класом може бути будь-яке

слово, яке описує певну групу об’єктів, проте варто зазначити, що зазвичай використовуються загальноприйняті класи об’єктів, для уникнення можливих непорозумінь.

Детекція об’єктів полягає у визначенні об’єктів на зображенні, та їх розташування в просторі. Загально прийнято здійснювати детекцію об’єктів за допомогою прямокутників, що обмежують в собі відповідний об’єкт.

Семантична сегментація полягає у визначенні схожих об’єктів на зображенні, що належать спільному класу на рівні пікселів.

Сутнісна сегментація полягає у визначенні різних сутностей, що знаходяться на зображенні з їхніми межами на піксельному рівні.

В рамках цієї роботи мене найбільше цікавить задача детекції, що зрозуміло з обраної теми, проте у подальшому будуть згадуватися деякі моменти, які мають відношення до задачі класифікації.

Практичне застосування задачі детекції об’єктів

Отож, задача детекції об’єктів полягає в тому, щоб встановити розташування об’єкту та його належність до певного класу. Практичне застосування цієї задачі є досить широким, а саме: розпізнавання лиць, асистенти для автомобілів, відеоспостереження, фіксація кількості людей в натовпі, детекція аномалій на виробництвах та інші. Звісно цей список не є вичерпним, і в наш час варіантів застосування методів для розв’язання цієї задачі стає все більше.

Висновки

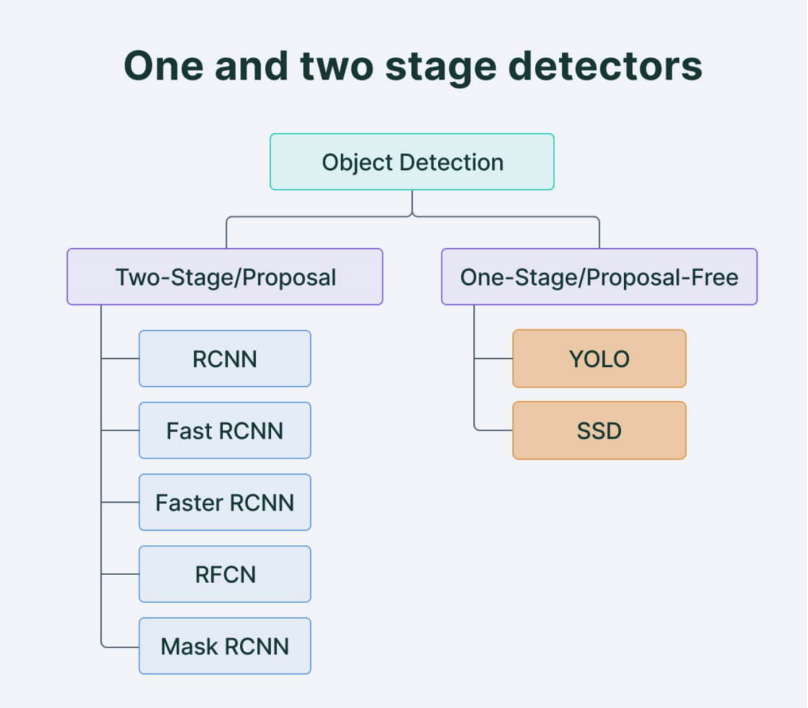
В рамках розділу було роглянуто сенс та практичне застосування задачі детекції.

КЛАСИЧНІ МЕТОДИ ДЛЯ ЗАДАЧІ ДЕТЕКЦІЇ

Загальні відомості

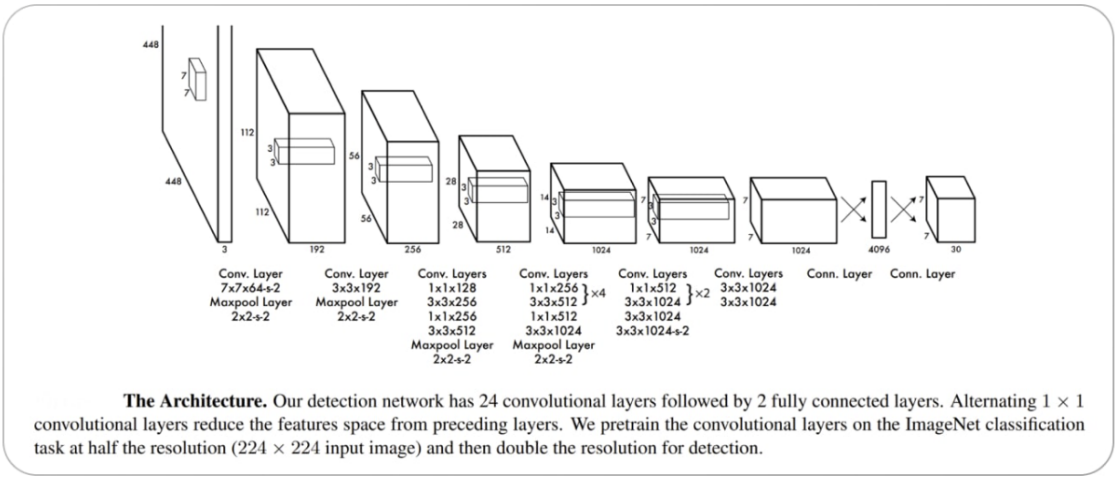
У цьому розділі розглянемо класичні методи для розв’язання задачі детекції. Умовно усі підходи можна поділити на ті що використовують машинне навчання (МН), та ті що використовують глибоке навчання (ГН). У більш традиційних підходах МН методи комп’ютерного зору використовують для того щоб отримати певні ознаки із зображень, по типу гістограма кольорів, або меж об’єктів, щоб визначити групи пікселів, що можуть належати об’єкту. Ці ознаки можуть бути використані у моделі регресії, щоб передбачити локацію об’єктів з та їх назву. З іншого боку, методи на основі ГН, використовують згорткові нейронні мережі (CNN) для виявлення об’єктів при чому ознаки не потрібно діставати окремо. Саме методи на основі глибинного навчання стали так званими “state-of-the-art" підходами, тобто передовими та найефективнішими підходами. Зазвичай моделі виявлення об’єктів на основі глибинного навчання складаються з двох частин. Енкодер приймає зображення на вхід і пропускає його через низку блоків і шарів, які навчаються відокремлювати статистичні ознаки, що використовуються для визначення місцезнаходження та позначення об’єктів. Вихідні дані енкодера передаються декодеру, який визначає рамки знаходження та мітки для об’єктів. Найпростіший декодер - це чистий регресор підключений до виходу енкодера і безпосередньо прогнозує розташування і розмір кожної обмежувальної рамки. Результатом роботи моделі є пара координат X, Y для об’єкта то його розміри на зображенні. Незважаючи на простоту цей тип моделі є доволі обмеженим, в тому контексті, що необхідно заздалегідь знати кількість об’єктів, які потрібно передбачити на відповідному зображенні, проте якщо заздалегідь відомо кількість об’єктів, то чисті регресійні моделі можуть добре працювати.

Загалом моделі детекції поділяються на кілька “сімей”, отож розглянемо загальні принципи роботи кожного з підходів.



YOLO

Алгоритм YOLO отримує зображення на вході, а потім використовує просту нейронну мережу глибокої згортки для виявлення об'єктів на зображенні. Архітектура моделі CNN, яка є основою YOLO, показана нижче.



The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224 input image) and then double the resolution for detection.

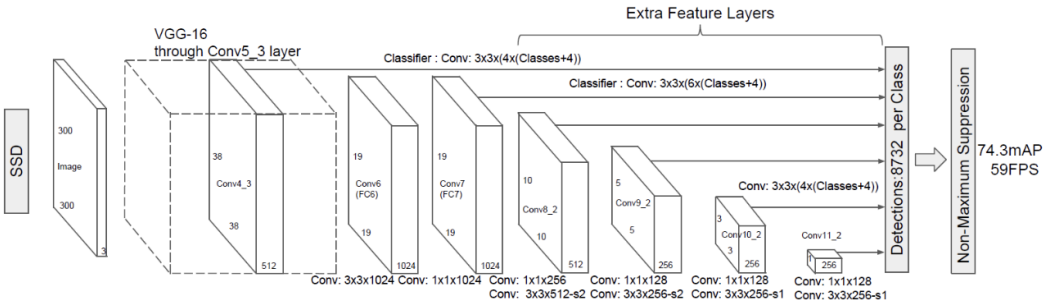
Перші 20 шарів згортки моделі попередньо навчаються за допомогою ImageNet шляхом додавання тимчасового середнього об'єднання і повністю зв'язаного шару. Потім ця попередньо навчена модель конвертується для виконання виявлення, оскільки попередні дослідження показали, що додавання згортки та зв'язаних шарів до попередньо навченої мережі покращує продуктивність. Останній повністю зв'язний шар YOLO прогнозує як ймовірності класів, так і координати обмежувальних рамок. YOLO ділить вхідне зображення на сітку S × S. Якщо центр об'єкта потрапляє в комірку сітки, ця комірка відповідає за виявлення цього об'єкта. Кожна комірка сітки прогнозує B обмежувальних рамок і довірчі оцінки для цих рамок. Ці оцінки впевненості відображають, наскільки модель впевнена в тому, що область містить об'єкт, і наскільки точною вона вважає передбачену область. YOLO прогнозує декілька обмежувальних рамок на кожну клітинку сітки. Під час навчання ми хочемо, щоб за кожен об'єкт відповідав лише один предиктор граничної області. YOLO призначає один предиктор "відповідальним" за прогнозування об'єкта, на основі якого прогноз має найвищу поточну IOU з базовою істиною. Це призводить до спеціалізації між предикторами граничної області. Кожен предиктор стає кращим у прогнозуванні певних розмірів, співвідношень сторін або класів об'єктів, покращуючи загальну оцінку прогнозу. Однією з ключових технік, що використовуються в моделях YOLO, є не-максимальне придушення (NMS). NMS - це етап пост обробки, який використовується для підвищення точності та ефективності виявлення об'єктів. При виявленні об'єктів для одного об'єкта на зображенні зазвичай генерується кілька обмежувальних рамок. Ці рамки можуть перекриватися або розташовуватися в різних місцях, але всі вони представляють один і той самий об'єкт. NMS використовується для виявлення та видалення надлишкових або неправильних рамок, а також для виведення однієї рамки для кожного об'єкта на зображенні.

YOLO (You Only Look Once) - популярний алгоритм виявлення об'єктів, який здійснив революцію в галузі комп'ютерного зору. Він швидкий і ефективний, що робить його чудовим вибором для задач детекції об'єктів у реальному часі. Він показав найкращі результати в різних тестах і отримав широке застосування в різних реальних додатках. Однією з головних переваг YOLO є висока швидкість виведення, що дозволяє обробляти зображення в реальному часі. Він добре підходить для таких застосувань, як відеоспостереження, безпілотні автомобілі та доповнена реальність. Крім того, YOLO має просту архітектуру і вимагає мінімальних навчальних даних, що полегшує його впровадження та адаптацію до нових завдань. Незважаючи на такі обмеження, як боротьба з дрібними об'єктами і нездатність виконувати тонкоструктурну класифікацію об'єктів, YOLO виявився цінним інструментом для виявлення об'єктів і відкрив багато нових можливостей для дослідників і практиків. Оскільки сфера комп'ютерного зору продовжує розвиватися, буде цікаво спостерігати за тим, як YOLO та інші алгоритми виявлення об'єктів розвиваються і вдосконалюються.

SSD

SSD базується на використанні згорткових мереж, які створюють декілька обмежувальних рамок різного фіксованого розміру та оцінюють наявність екземплярів об'єктного класу в цих рамках, після чого виконується не максимальний крок придушення для отримання остаточного результату виявлення. Модель SSD працює наступним чином: кожне вхідне зображення розбивається на сітки різного розміру, і на кожній сітці виконується детектування для різних класів і різних співвідношень сторін. Кожній з цих сіток присвоюється оцінка, яка показує, наскільки добре об'єкт відповідає конкретній сітці. Для отримання остаточного виявлення з набору виявлень, що перекриваються, застосовується не максимальне

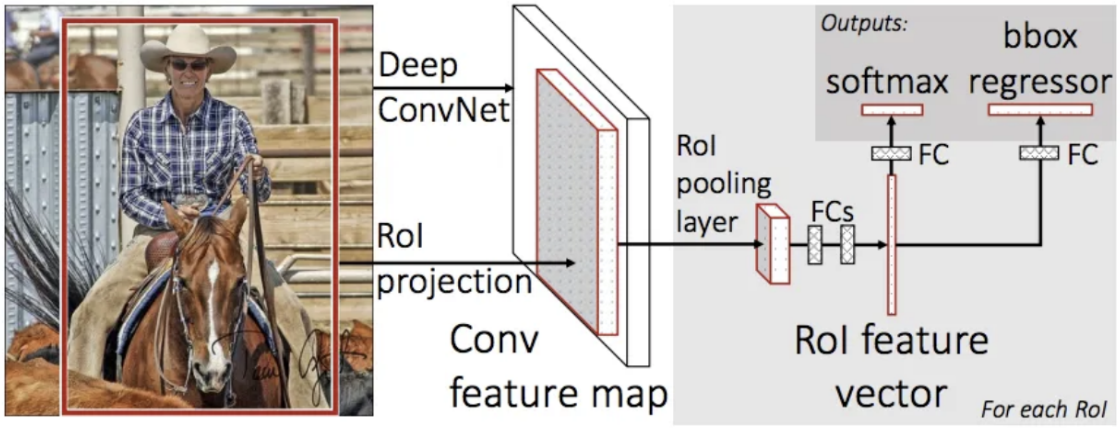
придушення. Це основна ідея моделі SSD. Використовуються різні розміри сітки для виявлення об'єктів різного розміру, наприклад, подивіться на зображення нижче, коли ми хочемо виявити кота, використовується менша сітка, але коли ми хочемо виявити собаку, розмір сітки збільшується, що робить SSD більш ефективною. Багатомасштабні карти особливостей додаються в кінець усіченої опорної моделі. Ці різномасштабні карти особливостей поступово зменшуються в розмірі, що дозволяє проводити детектування на різних масштабах зображення. Шари згортки, що використовуються тут, відрізняються для кожного шару ознак. Додавання кожного додаткового шару дає фіксовану кількість прогнозів з використанням згорточних фільтрів у них. Ці додаткові шари показані у верхній частині моделі на наведеній нижче діаграмі. Наприклад, для шару ознак розміром m x n з p каналами мінімальним параметром передбачення, який дає пристойне виявлення, є маленьке ядро розміром 3 x 3 x p. Таке ядро дає нам оцінку для категорії або зміщення фігури відносно координат поля за замовчуванням.



Fast R-CNN

Щоб обійти проблему вибору величезної кількості регіонів, Росс Гіршик (Ross Girshick) та ін. запропонували метод, в якому ми використовуємо вибірковий пошук для вилучення лише 2000 регіонів із зображення, і він назвав їх пропозиціями регіонів. Таким чином, тепер, замість того, щоб

намагатися класифікувати величезну кількість регіонів, ви можете просто працювати з 2000 регіонами. Ці 2000 пропозицій регіонів генеруються за допомогою алгоритму вибіркового пошуку, який описано нижче.



Підхід схожий на алгоритм R-CNN. Але замість того, щоб подавати в CNN пропозиції регіону, ми подаємо вхідне зображення в CNN для створення згорнутої карти ознак. На згортковій карті ознак ми визначаємо область пропозицій, розбиваємо їх на квадрати і, використовуючи шар об'єднання RoI, перетворюємо їх у фіксований розмір, щоб його можна було подати в повністю зв'язаний шар. На основі вектора ознак RoI ми використовуємо шар softmax для прогнозування класу запропонованої області, а також значень зсуву для обмежувальної рамки.

Висновки:

Отже, в рамках розділу було розглянуто основні моделі та їх принципи роботи, що використовуються для задачі детекції.

МЕТОД ДЕТЕКЦІЇ ОБ'ЄКТІВ ВИКОРИСТОВУЮЧИ ІЄРАРХІЧНЕ ДЕРЕВО КЛАСІВ ОБ'ЄКТІВ

Сильні сторони цього методу

Загальна мета даного методу полягає в тому, щоб бути швидким, точним та здатним розпізнати широку різноманітність об'єктів. Крім цього, датасети для деткції об'єктів є обмеженими у порівнянні з датасетами для класифікації зображень. В даному методі використовується підхід, що дозволяє використовувати велику кількість даних для задачі класифікації, використовувати для того, щоб розширити можливості систем детекції об'єктів. Отже, пропонується об'єднаний алгоритм навчання, що може тренувати детектор об'єктів, як на даних для детекції, так на даних для класифікації. Використовуються розмічені зображення для виявлення об'єктів, щоб точно навчитися локалізувати об'єкти, та класифікаційні зображення, щоб розширити словниковий запас та збільшити надійність. Використовуючи даний метод, вдалося отримати детектор, що може виявити об'єкти, які належать до понад 9000 різних категорій. В основі методу лежить модель YOLO, а точніше її покращена версія YOLO v2. Зараз розглянемо модифікації, які були здійснені.

Батчева нормалізація.

Батчева нормалізація призводить до значного покращення збіжності, усуваючи при цьому необхідність в інших формах регуляризації. Додавання такої нормалізації на всіх згорткових шарах YOLO дозволило покращити метрику mAP на понад 2%. Крім цього, батчева нормалізація допомагає регуляризувати модель. Крім цього варто зазначити, що з такою нормалізацією вдається уникати викидів, уникаючи перенавчання.

Класифікатор великої розмірності.

Всі state-of-art методи використовують заздалегідь навчені класифікатори на базі даних ImageNet. Починаючи з класифікатора AlexNet, більшість класифікаторів оперують зображеннями менше ніж 256 x 256. Оригінальна модель YOLO для класифікації тренується на розмірностях 254 x 254 і розмірність збільшується до 448 для детекції. Це означає, що нейронна мережа повинна одночасно переключитися на навчання детекції об’єктів і збільшити розмірність вхідних зображень. У даному методі для YOLOv2 використовувались зображення розмірності 448 x 448 для 10 епох на даних ImageNet. Це дає мережі час щоб налаштувати фільтри для кращої роботи на вході з високою роздільною здатністю. Саме така класифікація з високою розмірністю дає збільшення mAP майже на 4%.

Згортки з якірними коробками

YOLO передбачає координати обмежувальних рамок, використовуючи повністю з’єднані шари поверх екстрактора згорткових ознак. Замість прямого передбачення координат Faster R-CNN прогнозує обмежувальні рамки, використовуючи попередні значення. Використовуючи лише згорткові шари, мережа регіональних пропозицій (RPN) у Faster R-CNN прогнозує зміщення та достовірність для якірних коробок. Оскільки шар прогнозування є згортковим, RPN прогнозує ці зміщення в кожному місці об'єкта на зображенні. Прогнозування зміщень замість координат спрощує задачу і полегшує навчання мережі. Видаляється повністю з’єднані шари з YOLO і використовуються опорні блоки для передбачення обмежувальних блоків. Спочатку вилучається один об’єднувальний шар, щоб зробити вихідні згорнуті шари мережі з вищою роздільною здатністю. Об’єкти, особливо великого розміру мають займати єдине місце прямо в центрі зображення, щоб спрогнозувати положення цих об’єктів, замість можливих розташувань поруч. Згорткові шари YOLO зменшують дискретизацію

зображення у 32 рази, тому, використовуючи вхідне зображення розмірності 416, ми отримуємо вихідну карту ознак розміром 13 × 13. Коли ми переходимо до якірних коробок, ми також відокремлюємо механізм передбачення класів від просторового розташування і натомість прогнозуємо клас і об'єктність для кожного опорного блока. Використовуючи якірні коробки, маємо зменшення в точності. YOLO передбачує тільки 98 коробок на зображенні але з використанням якірних коробок отримана модель може передбачити більше тисячі. Без якірних коробок модель отримує 69.5 mAP з показниками метрики recall 81%. З ними модель отримує 69.2 mAP з показниками метрики recall 88%. Незважаючи на зниження mAP, збільшення метрики recall означає що модель має в собі простір для покращення.

Кластери розмірності.

Отож, існує дві проблеми з якірними коробками в контексті використання їх з моделлю YOLO. Перша полягає в тому, що розмірність для якірних коробок підбирається вручну. Мережа може навчитися правильно підбирати такі ящики, але якщо вдасться підібрати правильний пріоритет для мережі щоб почати, ми спростимо задачу для мережі, щоб вчитися гарно здійснювати детекції. Замість того, щоб визначати розмірність власноруч, ми можемо використати алгоритм кластеризації K-means на тренувальний сет рамок об’єктів, для автоматичного визначення гарних пріоритетів.

Пряме передбачення локації.

Коли ми використовуємо якірні коробки з YOLO ми стикаємося з другою проблемою: нестабільністю моделі, особливо на ранніх ітераціях. Більша частина нестабільності пов'язана з прогнозуванням розташування (x, y) коробки. Замість прогнозування зміщень ми використовуємо підхід YOLO

і прогнозуємо координати місцезнаходження відносно розташування комірки сітки. Це обмежує значення істини в межах між 0 та 1. Ми використовуємо логістичну активацію, щоб обмежити прогнози мережі в цьому діапазоні.

Дрібні ознаки

Цей модифікований YOLO прогнозує виявлення об’єктів на карті ознак розмірності 13 x 13. Хоча цього достатньо для великих об'єктів, для локалізації менших об'єктів можуть бути корисними більш дрібні ознаки. Швидші R-CNN та SSD використовують свої різноманітні мапи об'єктів у мережі, щоб отримати діапазон роздільної здатності. Використовуються різні підходи додаючи прохідний шар, який використовує ознаки більш раннього шару розмірності 26 x 26. Прохідний шар об’єднує об’єкти з високою роздільною здатністю з об’єктами низької роздільної здатності шляхом об’єднання сусідніх об’єктів у різні канали замість просторового розташування, подібного до відображення ідентичності у ResNet. Це перетворює карту ознак розмірності 26 x 26 x 512 на розмірність 13 x 13 x 2048, які можуть бути об’єднаними з оригінальними ознаками.

Різномасштабне навчання

Оригінальний YOLO використовує вхідні данні з роздільною здатністю 448 × 448. З додаванням якірних коробок було змінено розмірність на 416 × 416. Однак, оскільки модель використовує лише згорнуті та об'єднані шари, їй можна змінювати розмір на в процесі роботи. Ми хочемо, щоб YOLOv2 був стійким до роботи на зображеннях різного розміру, тож ми навчаємо модель цьому. Кожні 10 пакетів, наша мережа випадковим чином обирає новий розмір зображення. Оскільки модель зменшує розмірність на 32 то вибираються наступні числа кратні 32: {320, 352, ..., 608}. Таким чином найменший варіант - 320 × 320, а найбільший - 608 × 608. Розмір

мережі змінюється і процес обробки продовжується. Цей режим дозволяє мережі як навчатися так і здійснювати передбачення на різних розмірностях. Мережа працює швидше і на менших розмірах, отож YOLOv2 пропонує баланс між швидкістю та точністю.

Darknet-19

В рамках даного методу пропонується нова модель класифікації на базі YOLOv2. Модель ґрунтується на загальних знаннях у цій сфері. Подібно до моделей VGG, використовуються переважно 3 × 3 фільтрів і подвоюємо кількість каналів після кроку об'єднання. Для глобального середнього об'єднання використовується так звана Network in Network, крім цього використовуються фільтри 1 × 1 для стиснення представлення ознак між згортками 3 × 3. Крім цього використовується пакетна нормалізація для стабілізації навчання, прискорення збіжності та регуляризації моделі. Фінальна модель Darknet-19 має 19 згорткових шарів і 5 шарів з максимальним об'єднанням. Для повного опису можна поглянути в таблицю нижче.

Type	Filters	Size/Stride	Output
Convolutional	32	3×3	224×224
Maxpool		$2 \times 2/2$	112×112
Convolutional	64	3×3	112×112
Maxpool		$2 \times 2/2$	56×56
Convolutional	128	3×3	56×56
Convolutional	64	1×1	56×56
Convolutional	128	3×3	56×56
Maxpool		$2 \times 2/2$	28×28
Convolutional	256	3×3	28×28
Convolutional	128	1×1	28×28
Convolutional	256	3×3	28×28
Maxpool		$2 \times 2/2$	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Convolutional	256	1×1	14×14
Convolutional	512	3×3	14×14
Maxpool		$2 \times 2/2$	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	512	1×1	7×7
Convolutional	1024	3×3	7×7
Convolutional	1000	1×1	7×7
Avgpool		Global	1000
Softmax			

Darknet-19 потребує лише 5,58 мільярдів операцій для обробки зображення, але досягає 72.9% точності на рівні top-1 і 91.2% точності топ-5 на ImageNet.

Об’єдання даних для класифікації та детекції

Тренування для класифікації. Мережа навчається на стандартному датасеті для класифікації ImageNet1000 для 160 епох, використовуючи стохастичний градієнтний спуск зі стартовим learning rate 0.1, поліноміальний степінь 4, decay 0.0005 і momentum 0.9. В процесі навчання використовуються стандартні прийоми перетворення даних, такі як масштабування, обертання, а також зміну відтінку, насиченості та експозиції. Як було сказано вище, після початкового навчання на зображеннях розміром 224 × 224, ми доналаштовуємо нашу мережу на

більшому розмірі, 448. Для цього тонкого налаштування ми тренуємось з наведеними вище параметрами, але лише для 10 епох і зі швидкістю навчання 10^{-3}. При цій вищій роздільній здатності наша мережа досягає найкращої точності 76.5% і точності топ-5 93.3%.

Навчання для детекції. Ми модифікуємо цю мережу для виявлення, видаливши останній згортковий шар і натомість додавши три зготкові шари 3 × 3 з 1024 фільтрами кожен, за якими слідує останній згортковий шар 1 × 1 з кількістю вихідних об’єуктів, які необхідно виявити. Для VOC ми прогнозуємо 5 коробок з 5 координатами в кожній і 20 класів на коробку, тобто 125 фільтрів. Ми також додаємо наскрізний шар з останнього шару 3 × 3 × 512 до передостаннього згорткового шару, щоб модель могла використовувати дрібні ознаки.

В рамках цього методу пропонується механізм для з’єднання даних для задачі детекції та класифікації. Коли мережа бачить зображення з міткою для детекції, воно проходить зворотню прогонку на основі функції втрат YOLOv2. Коли вона бачить зображення класифікації, ми поширюємо втрати лише з частин архітектури специфічних для класифікації. Такий підхід пов'язаний з кількома проблемами. У наборах даних є лише спільні об’єкти та згальні мітки, проте на наприклад датасет ImageNet включає в себе більше сотні порід собак. Якщо потрібно тренуватися на цих даних, нам потрібен узгоджений спосіб об’єднати ці мітки. Більшість підходів використовує шар softmax для всіх можливих категорій, щоб обчислити остаточний розподіл ймовірностей. Використання софтмаксу передбачає, що класи є взаємовиключними. Це створює проблеми при об'єднанні наборів даних, наприклад, не вдасться об'єднувати ImageNet і COCO за допомогою цієї моделі, тому що класи “собака” та “помарнцевий шпіц” не є взаємовиключні. Отож, треба якийсь варіант об’єдання даних в таких

випадках. Це може бути використання, мульти-міток для об’єднання яка не припускає взаємо виключення.

Ієрархічна класифікація

Розмітка даних ImageNet використовує WordNet, мову баз даних що структурує принцип того як ці слова взаємопов’язані. WordNet має структуру орієнтованого графа, а не дерева, оскільки мова є складною. Наприклад, "собака" є одночасно тип "собака" і тип "домашня тварина", які у WordNet є синтаксичними множинами. Замість того, щоб використовувати повну структуру графа ми спростимо задачу, побудувавши ієрархічне дерево з концептів в ImageNet. Щоб побудувати це дерево, ми розглядаємо візуальні іменники в ImageNet і дивимося на їхні шляхи через граф WordNet до кореневого вузла, у цьому випадку "фізичний об'єкт". Потім ми ітеративно переглядаємо концепції, що залишилися, і додаємо шляхи, які збільшують дерево на якомога меншу відстань. Отже, якщо концепт має два шляхи до кореня корінь, і один з них додає три ребра до нашого дерева, а інший - лише одне, а інший - лише одне ребро, ми обираємо коротший шлях. Фінальним результатом буде WordTree, ієрархічна модель, що візуалізує концепти. Щоб використати класифікацію WordTree, ми передбачуємо умовну ймовірність на кожному вузлі для кожного гіпоніма цієї синонімічної множини з урахуванням цієї синонімічної множини. Наприклад, для слова “terrier”:

$$Pr(\text{Norfolk terrier}|\text{terrier})$$
$$Pr(\text{Yorkshire terrier}|\text{terrier})$$
$$Pr(\text{Bedlington terrier}|\text{terrier})$$
$$\ldots$$

Якщо ми хочемо обчислити абсолютну ймовірність для певного вузла, просто пройдімося по дереву до кореневого вузла і помножити на умовну

ймовірність. Таким чином якщо ми хочемо дізнатися, чи зображений на фото Norfolk terrier, ми обчислюємо

$$\begin{aligned} Pr(\text{Norfolk terrier}) &= Pr(\text{Norfolk terrier}|\text{terrier}) \\ &*Pr(\text{terrier}|\text{hunting dog}) \\ &*\ldots* \\ &*Pr(\text{mammal}|Pr(\text{animal}) \\ &*Pr(\text{animal}|\text{physical object}) \end{aligned}$$

Для класифікаційних цілей ми вважаємо, що зображення містить об’єкт: Pr(physical object) = 1. Для перевірки цього підходу ми тренуємо модель Darknet-19

на WordTree, побудованому з використанням 1000 класову ImageNet. Для побудови WordTree1k ми додаємо всі проміжні вузли, що розширює простір міток з 1000 до 1369. Під час навчання ми поширюємо істинні мітки вгору по дереву так, що якщо зображення позначено як "норфолк тер'єр", то воно також буде позначено як "собака", "ссавець" і т.д. Для обчислення умовних ймовірностей модель прогнозує вектор з 1369 значень і ми обчислюємо softmax для всіх синтаксичних наборів, які є гіпонімами одного поняття.

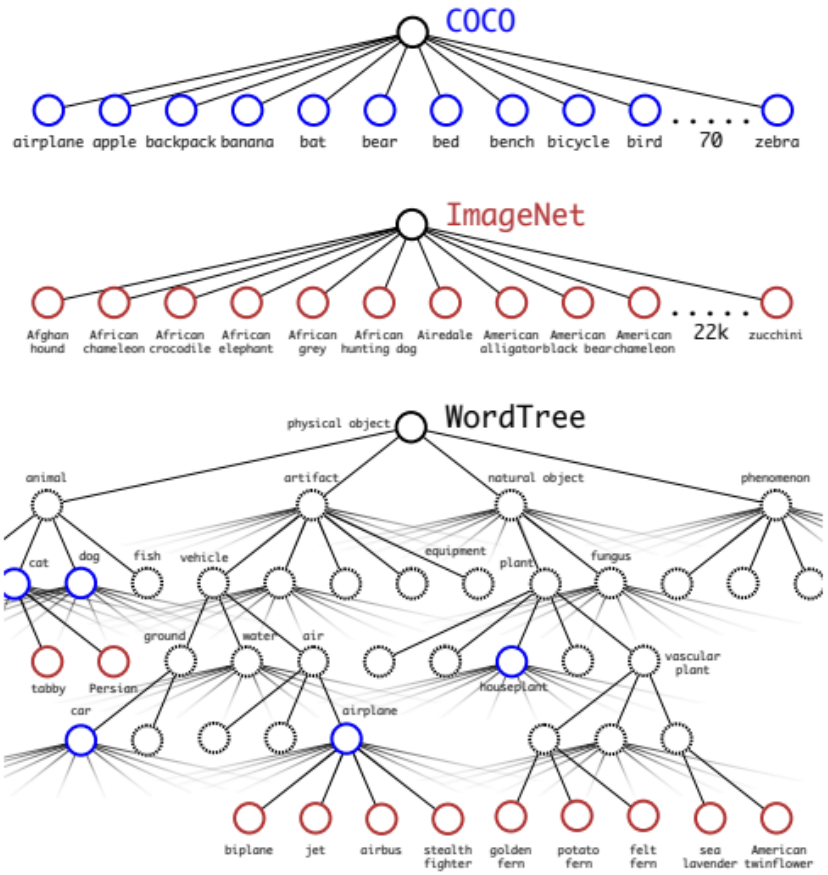
Комбінація датасетів за допомогою WordTree

Отож, використовуючи такий підхід, можна комбінувати кілька датасетів разом. По суті відбувається встановлення відповідності між категоріями в датасеті та елементами дерева. На зображенні нижче зображено приклад того, як поєднуються данні ImageNet та Coco.

Комбінація класифікації та детекції

Тепер, коли можна об'єднати набори даних за допомогою WordTree, ми

можемо тренувати нашу спільну модель на класифікацію та виявлення. Для того, щоб навчити надзвичайно масштабний детектор, тому ми створюємо наш комбінований набір даних, використовуючи набір даних для виявлення COCO і 9000 найкращих класів з повної версії ImageNet. Також потрібно оцінити метод, тому додаються будь-які класи з ImageNet, яких ще не було включено. Відповідне дерево WordTree для цього набору даних містить 9418 класів. ImageNet є набагато більшим набором даних, тому ми збалансуємо набір даних шляхом надмірної вибірки COCO, так що ImageNet був більшим лише у співвідношенні 4:1.



Коли стоїть зображення класифікації, ми лише поширюємо втрату класифікації. Для цього ми просто знаходимо обмежувальну область, яка прогнозує найвищу ймовірність для цього класу і обчислюємо втрату лише на його передбаченому дереві. Ми також припускаємо, що прогнозована

область перекриває те, що могло б бути міткою ground truth щонайменше на 0.3 IOU, і ми зворотньо розповсюджуємо втрату об'єктності на основі цього припущення. Використовуючи це спільне навчання, YOLO9000 вчиться знаходити об'єкти на зображеннях, використовуючи дані виявлення в COCO, і він вчиться класифікувати широкий спектр цих об'єктів, використовуючи дані з ImageNet. Оцінюється YOLO9000 на задачі виявлення ImageNet. Завдання виявлення для ImageNet розділяє 44 категорії об'єктів з COCO, що означає, що YOLO9000 обробляв дані класифікації лише для більшості тестових зображень, а не дані виявлення. YOLO9000 отримує 19,7 mAP в цілому з 16,0 mAP на розрізнених 156 класах об'єктів, для яких він ніколи не бачив жодних даних про виявлення міток. Цей показник mAP вищий ніж результати, досягнуті DPM, але YOLO9000 навчався на різних наборах даних з лише частковим наглядом [4]. Він також одночасно виявляє 9000 інших категорій об'єктів, і все це в режимі реального часу. Коли ми аналізуємо роботу YOLO9000 на ImageNet, то бачимо, що він добре вчиться розпізнавати нові види тварин, але має проблеми з такими категоріями, як одяг та спорядження.

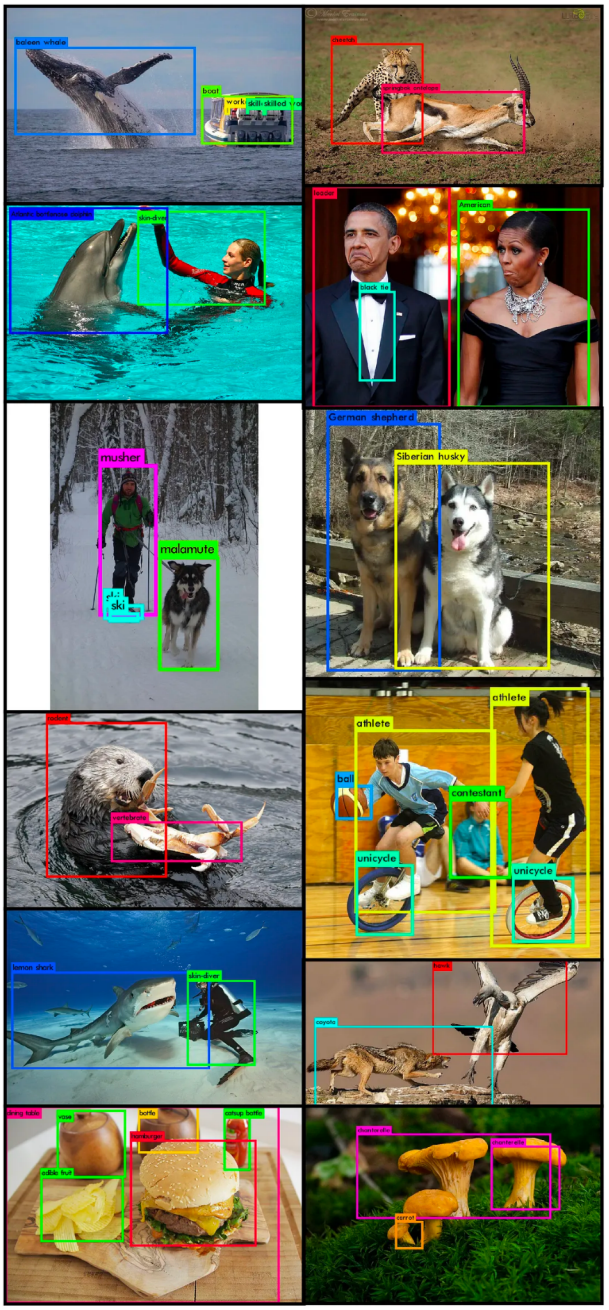
Підсумки

Отож, отримана в рамках даного методу модель, YOLO9000 - це фреймворк для виявлення в реальному часі більше 9000 категорій об'єктів шляхом спільної оптимізації виявлення та класифікації. Ми використовуємо WordTree для об'єднання даних з різних джерел різних джерел і нашу спільну техніку оптимізації для навчання одночасно на ImageNet і COCO. YOLO9000 - це значний крок на шляху до подолання розриву в розмірах наборів даних між виявленням і класифікацією. WordTree представлення ImageNet пропонує багатший, детальніший вихідний простір для класифікації зображень. Об'єднання наборів даних за допомогою ієрархічної класифікації може бути було б корисним у сферах

класифікації та сегментації. Методи навчання, такі як багато масштабне навчання, можуть бути корисними користь у вирішенні різноманітних візуальних завдань.

ПОРІВНЯННЯ МЕТОДІВ

На зображенні нижче показано приклади роботи даного метода



В таблиці нижче зображено основні порівнянні результатів роботи методу для різних класів VOC 2012 detection dataset

Method	data	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Fast R-CNN [5]	07++12	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	87.5	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
Faster R-CNN [15]	07++12	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
YOLO [14]	07++12	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
SSD300 [11]	07++12	72.4	85.6	80.1	70.5	57.6	46.2	79.4	76.1	89.2	53.0	77.0	60.8	87.0	83.1	82.3	79.4	45.9	75.9	69.5	81.9	67.5
SSD512 [11]	07++12	74.9	87.4	82.3	75.8	59.0	52.6	81.7	81.5	90.0	55.4	79.0	59.8	88.4	84.3	84.7	83.3	50.2	78.0	66.3	86.3	72.0
ResNet [6]	07++12	73.8	86.5	81.6	77.2	58.0	51.0	78.6	76.6	93.2	48.6	80.4	59.0	92.1	85.3	84.8	80.7	48.1	77.3	66.5	84.7	65.6
YOLOv2 544	07++12	73.4	86.3	82.0	74.8	59.2	51.8	79.8	76.5	90.6	52.1	78.2	58.5	89.3	82.5	83.4	81.3	49.1	77.2	62.4	83.8	68.7

Результати на датасеті COCO 2015 test-dev dataset

		0.5:0.95	0.5	0.75	S	M	L	1	10	100	S	M	L
Fast R-CNN [5]	train	19.7	35.9	-	-	-	-	-	-	-	-	-	-
Fast R-CNN[1]	train	20.5	39.9	19.4	4.1	20.0	35.8	21.3	29.5	30.1	7.3	32.1	52.0
Faster R-CNN[15]	trainval	21.9	42.7	-	-	-	-	-	-	-	-	-	-
ION [1]	train	23.6	43.2	23.6	6.4	24.1	38.3	23.2	32.7	33.5	10.1	37.7	53.6
Faster R-CNN[10]	trainval	24.2	45.3	23.5	7.7	26.4	37.1	23.8	34.0	34.6	12.0	38.5	54.4
SSD300 [11]	trainval35k	23.2	41.2	23.4	5.3	23.2	39.6	22.5	33.2	35.3	9.6	37.6	56.5
SSD512 [11]	trainval35k	**26.8**	**46.5**	**27.8**	**9.0**	**28.9**	**41.9**	**24.8**	**37.5**	**39.8**	**14.0**	**43.5**	**59.0**
YOLOv2 [11]	trainval35k	21.6	44.0	19.2	5.0	22.4	35.5	20.7	31.6	33.3	9.8	36.5	54.4

Порівняння модифікації Darknet-19

Darknet19: A good balance of speed and accuracy

	Top 1	**Top 5**	**FLOPs**	**GPU Speed**
VGG-16	70.5	90.0	30.95 Bn	100 FPS
Extraction (YOLOv1)	72.5	90.8	8.52 Bn	180 FPS
Resnet50	**75.3**	**92.2**	7.66 Bn	90 FPS
Darknet19	74.0	91.8	**5.58 Bn**	**200 FPS**

ВИСНОВКИ

Отже, YOLOv2 став наступним логічним етапом у розвитку першої версії моделі. Її модифікація Darknet-19 це так чи інакше гарний баланс між швидкістю та тончістю. Ключовий метод даного дослідження YOLO9000 має ключову перевагу, у вигляді здатності розпізнати велику кількість класів, проте варто зазначити, що можливо на сьогоднішній день існують методи, які можуть похизуватися більш високую кількістю класів, які є здатність розпізнати, тому для подальших досліджень варто розглнути більш сучасні методи та порівняти з тими, що були роглянуті в рамках роботи.

ПОСИЛАННЯ

1. Redmon, J., Farhadi, A. (2017). YOLO9000: Better, Faster, Stronger. University of Washington, Allen Institute for AI.
2. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector.
3. Redmon, J., Divvala, S., Girshick, R., Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. University of Washington, Allen Institute for AI, Facebook AI Research
4. Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.
5. Analytics Vidhya. (n.d.). Deep Dive on YOLOv2 and YOLO9000 [Blog post]. Retrieved from https://medium.com/analytics-vidhya/deep-dive-on-yolov2-and-yolo9000-2eba212dcf8a
6. V7 Labs. (n.d.). YOLO Object Detection [Blog post]. Retrieved from https://www.v7labs.com/blog/yolo-object-detection
7. Brownlee, J. (n.d.). Object Recognition with Deep Learning [Blog post]. Retrieved from https://machinelearningmastery.com/object-recognition-with-deep-learning/