КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСАШЕВЧЕНКА
Факультет комп'ютерних наук та кібернетики

Кафедра обчислювальної математики

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА
за спеціальністю 113 Прикладна математика
на тему:

Розпізнавання об’єктів на супутникових знімках

Студента 4 курсу

кафедри обчислювальної математики

Сальника Дмитра Віталійовича

Науковий керівник

кандидат фізико-математичних наук,

Оноцький В’ячеслав Валерійович

_____

"___"_____2021р.

Робота заслухана на засіданні кафедри обчислювальної математики та рекомендована до захисту в ДЕК, протокол №__

Завідувач кафедри обчислювальної математики проф. Клюшин Д.А.

Київ 2021

Зміст

1 Вступ

У наш час ми опинилися в оточенні безлічі технологій, здатних взаємодіяти з навколишнім середовищем та збирати всі типи даних, починаючи від руху певних об’єктів до звуку для візуальних даних, отриманих при виявленні різних частин електромагнітного спектра. Одними з найбільш багатими на інформацію джерелами, і, можливо, найважчими для інтерпретування є візуальні дані.

Сьогодні ми можемо скористатися зростаючою кількістю супутників на навколоземній орбіті, обладнаними високоякісними камерами для виведення поточної інформації про земну кулю у великих масштабах, починаючи від розвідки і закінчуючи навколишнім середовищем та соціально-економічними тенденціями. Вміння аналізувати та інтерпретувати супутникові зображення автоматизовано може дати нам уявлення про корисну інформацію і дозволити нам приймати кращі оптимальні рішення, чи стосується це бізнесу, навколишнього середовища чи науки загалом.

Метою даної дипломної роботи є розробка та експериментальна оцінка нейронної мережі для виявлення та класифікації об’єктів на супутникових зображеннях високої роздільної здатності. Робота включає алгоритми та прийоми для попередньої обробки зображень, а також міркування та алгоритми для вилучення відповідних навчальних та тестових наборів, які використовуються для оптимізації запропонованої далі архітектури нейронних мереж для даного завдання.

Отже очікується, що архітектура виявлення та класифікації зможе обробляти зображення з різних джерел та з різною роздільною здатністю. Нейронна мережа також повинна класифікувати дані при неідеальних

ситуаціях, таких як надмірному покритті хмар або надмірному/ недоекспонованому зображенні.

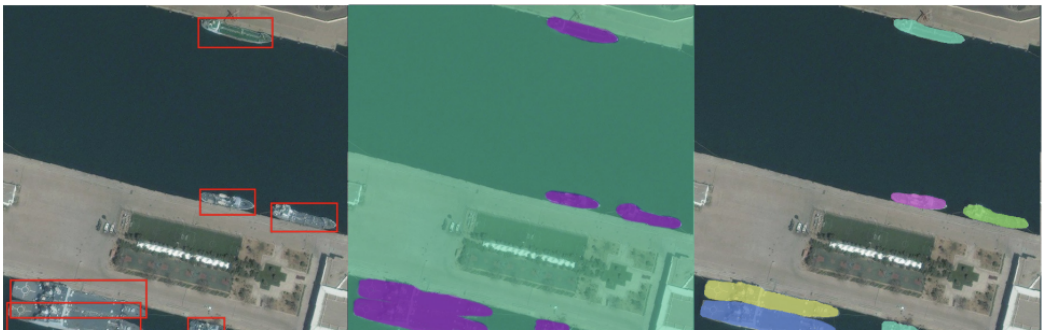
2 Постановка задачі

З появою комп'ютерного зору дослідники розробляють різні методики, які можуть дати змогу комп'ютерній програмі зрозуміти цифрові зображення. Це поняття поділяється на різні категорії.

По-перше, це класифікація зображень, яка є завданням класифікації зображень. Зазвичай це передбачає використання інформації з цілого зображення та передбачення однієї або кількох міток для цього зображення.

По-друге, локалізація об’єкта, яка є завданням передбачення обмежувального поля навколо об’єкта або декількох об’єктів на певному зображенні.

Більш недавній і більш складним завданням у розпізнаванні зображень є семантична сегментація, яка є процесом поділу цифрового зображення на декілька сегментів. Мета сегментації полягає в спрощені і/або зміні представлення зображення для більш простого подальшого аналізу. Сегментація зображень зазвичай використовується для того, щоб виділити об'єкти і кордони на зображеннях. Більш точно, сегментація зображень - це процес присвоєння таких міток кожному пікселю зображення, що пікселі з однаковими мітками мають загальні візуальні характеристики. Результатом сегментації зображення є безліч сегментів, які разом покривають все зображення, або безліч контурів, виділених з зображення. Всі пікселі в сегменті схожі за деякою характеристикою або обчисленому властивості, наприклад, за кольором, яскравості або текстурі. Сусідні сегменти значно відрізняються за цією характеристикою.



Локалізація об’єкта (обмежувальні рамки), семантична сегментація (фіолетові маски), сегментація екземпляру (кожен корабель має свій власний колір для розрізнення екземплярів).

Протягом багатьох років дослідники розробляли різні методики, що дозволити їм вирішити вищезазначені завдання. Деякі з них, такі як алгоритм виявлення обличчя Віоли-Джонса [12] та гістограма методу градієнтів для виявлення пішоходів були дуже успішними. Основна мета - перетворити необроблений піксель інформацію в більш абстрактне подання, що називається простором ознак, до якого ми можемо використовувати класифікатор для знаходження корисної інформації на зображенні. Більшість з методів, що досягають цього, складаються або включають певну форму згортки на зображенні для отримання карти об’єктів та статистичного регіонального об'єднання інформації. У методі Віоли-Джонса вони використовували характеристики Хаара [12], які в основному є зваженими сусідніми прямокутними областями, які були зсунуті по всьому зображенню, підсумовуючи інтенсивність пікселів у кожній області, щоб отримати корисне подання. В іншому методі автори використовували гістограми орієнтованого градієнта як основний екстрактор ознак, який згодом подавався лінійний класифікатор SVM [6]. Спільним у вищезазначених методів те, що вони досить чутливі до умов поставленої задачі, так зміна умов задачі може привести до зміну алгоритму. Цей недолік залишив дослідників у пошуках більш надійного та ефективного рішення.

У 1998 р. Ян ЛеКун продемонстрував дуже ефективну модель розпізнавання рукописних цифр на базі використання нейронних мереж, який використовував згортку як свої трансформаційні шари, що дозволило досягти понад 99% точності в завданні розпізнавання цифр, що відновило інтерес до досліджень використання нейронних мереж для розпізнавання зображень. На жаль, на той момент ми були суворо обмежені відсутністю обчислювальної потужності та техніки для ефективної підготовки нейронних мереж для виконання більш складних завдань, тому від них відмовлялися протягом багатьох років.

Справжній прихід і відродження нейронних мереж відбувся в 2012 році, коли Алекс Крижевський з великим відривом виграв ILSVRC, з своєю 7-шаровою згортковою нейронною мережею [21], кодова назва AlexNet. Цей успіх був досягнутий значною мірою завдяки вдосконаленням архітектури та технік, для навчання нейронних мереж та розвиток набагато потужніших обчислювальних машин, особливо велику роль зіграло значне прискорення GPU. Звідти дослідники розробили нові методи та архітектури для подальшого вдосконалення адаптація глибоких нейронних мереж для розпізнавання зображень.

3 Теоретична частина

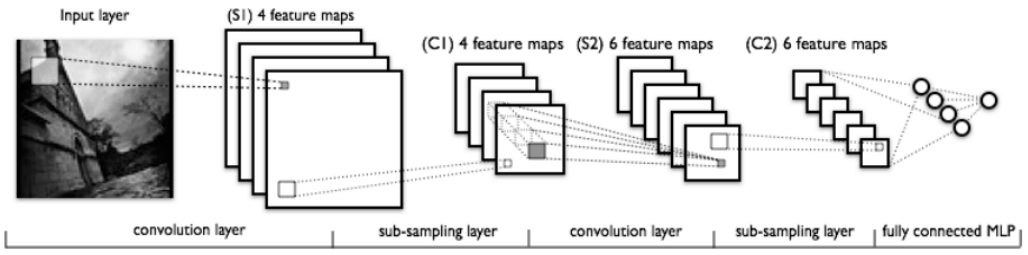
3.1 Концепція нейронних мереж для візуальних даних

Концепція нейронних мереж існує досить давно, але як було згадано в попередніх розділах, були деякі серйозні обмеження, що перешкоджали їх використанню і це навіть повністю виключало їх із завдань класифікації.

Під поняттям глибокої нейронної мережі найчастіше розуміють багатошаровий персептрон (MLP). Під "глибокою" ми розуміємо те, що мережа містить більше одного прихованого шару. Одне з обмежень раніше повністю підключених MLP при застосуванні до зображень було те, що нам потрібна була величезна кількість зв'язків між кожним пікселем та наступним шаром, що суттєво обмежило їх використання через обчислювальні вузькі місця. Як швидкий приклад, MLP с введення зображення розміром 256 × 256 та 256 × 256 прихованих одиниць у першому шарі потребує понад 4 × 109 параметрів. Інше питання до цієї архітектури полягає в тому, що ми не використовуємо просторову інформацію, яка є критичною у візуальних даних.

Для адаптації поняття MLP до візуальних даних Ян ЛеКан популяризував згортковий рівень перетворення нейронних мереж, концепція якого була вже введена в 1980р. Фукусімою (CNN) [7]. Ця техніка використовувала ідею ковзання квадратних згорткових ядр або фільтрів по всьому зображенню, щоб створити так звані карти ознак об’єктів, які подавались як вхідні параметри для наступного шару. Природно, в такій нейронної мережі фільтр не один, а ціла гама, що кодує елементи зображення (наприклад лінії і дуги під різними кутами). При цьому такі ядра згортки не закладаються дослідником заздалегідь, а формуються самостійно шляхом навчання мережі класичним методом

зворотного поширення помилки. Прохід кожним набором ваг формує свій власний примірник карти ознак, роблячи нейронну мережу багатоканальною (багато незалежних карт ознак на одному шарі). Також слід зазначити, що при переборі шару фільтром його пересувають зазвичай не на повний крок (розмір цієї матриці), а на невелику відстань. Так, наприклад, при розмірності матриці ваг 5 × 5 її зрушують на один або два пікселя замість п'яти, щоб не “переступити” через шукану ознаку. Використовуючи одне й те саме ядро для всього зображення означало, що ми значно зменшили кількість параметрів та виконуємо меншу кількість операцій на шар, а також отримуємо просторову інформацію зображення. У проміжку між згортковими шари використовують шари активації та max-pool шари або шари субдискретизації для подальшого зменшення роздільної здатності зображення. Кінцеві шари згорткової нейронної мережі підключають до MLP з одним або декількома прихованими шарами. Після чого серед поданих ознак робиться передбачення класів представлених на фото об’єктів. Далі розглянемо структуру згорткової мережі більш детально.



приклад побудови CNN

3.2 Будова CNN

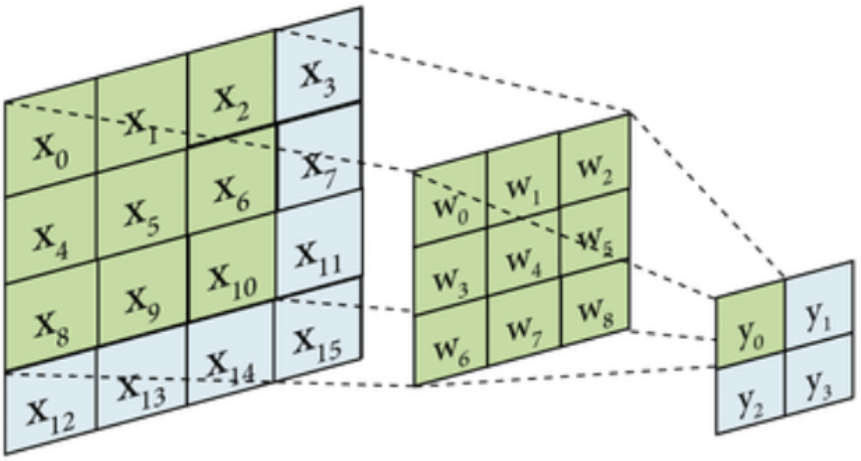
У попередньому пункті було описано роботу згорткової нейронної мережі. Так були зазначенні основні види внутрішніх шарів даного типу мереж, а також були оголошені головні плюси у порівнянні с MLP моделлю.

Розглянемо типову структуру CNN більш детально. Мережа складається з великої кількості шарів. Після початкового шару, на який подається вхідне зображення, сигнал проходить серію згорткових шарів, в яких чергується власне згортка і pooling. Чергування шарів дозволяє складати карти ознак, на кожному наступному шарі карта зменшується в розмірі, але збільшується кількість каналів. На практиці це означає здатність розпізнавання складних ієрархій ознак. На виході CNN додатково встановлюють кілька шарів MLP, на вхід якого подаються кінцеві карти ознак. Тепер опишемо структуру та функцію кожного шару згорткової нейронної мережі більш детально.

Шар згортки – це основний блок згорткової нейронної мережі. Шар згортки включає в себе декілька фільтрів, ядро згортки якоих обробляє попередній шар за фрагментами операцією двовимірної згортки, яка є досить простою операцією. Так ядро, що представляє із себе матрицю ваг, “ковзає” над двовимірним зображенням, поелементно виконуючи операцію множення з тією частиною вхідних даних, над якою воно зараз знаходиться, і потім підсумовує всі отримані значення в один вихідний піксель, де вагові коефіцієнти ядра згортки невідомі і встановлюються в процесі навчання. Ядро повторює цю процедуру з кожної локацією, над якою воно “ковзає”, перетворюючи двовимірну матрицю в іншу все ще двовимірну матрицю ознак. Ознаки на виході є зваженими сумами ознак на вході, розташованих приблизно в тому ж місці, що і вихідний піксель на вхідному шарі. Незалежно від того, чи потрапляє вхідна ознака в

“приблизно те ж місце”, визначається в залежності від того, знаходиться вона в зоні ядра, що створює вихідні дані, чи ні. Це означає, що розмір ядра CNN визначає кількість ознак, які будуть об'єднані для отримання нової ознаки на виході.

Особливістю цього шару є порівняно невелика кількість параметрів, яка встановлюється при навчанні. Так наприклад, якщо вихідне зображення має розмірність 100 × 100 пікселів по трьом каналам, тобто містить у собі 30000 вхідних нейронів, а згортковий шар використовує фільтри с ядром 3 × 3 пікселя з виходом на 6 каналів, тоді в процесі навчання визначається тільки 9 ваг ядра, однак по всім сполученням каналів, тобто 9 × 3 × 6 = 162, в такому випадку даний шар вимагає знаходження тільки 162 параметрів, що істотно менше кількості шуканих параметрів повної нейронної мережі.



Слід зазначити, що згортка зменшує початкову матрицю, а для того щоб цього уникнути використовують padding, який додає до країв підроблені пікселі, зазвичай нульового значення, внаслідок цього до них застосовується термін – “zero padding”. Таким чином, ядро при

прослизанні дозволяє непідробним пікселям надаватися в своєму центрі, а потім поширюється на підроблені пікселі за межами краю, створюючи вихідну матрицю того ж розміру, що і вхідні.

Буває ж навпаки, що при роботі зі згортковим шаром, потрібно отримати вихідні дані меншого розміру, ніж вхідні. Це зазвичай необхідно в CNN, де розмір просторових розмірів зменшується при збільшенні кількості каналів. Один із способів досягнення цього – використання субдискретізаційних шарів або pooling шар, наприклад, приймати середнє/максимальне значення кожної квадрату розміром 2 * 2 щоб зменшити всі просторові розміри в два рази. Ще один спосіб домогтися цього – використовувати stride.

Ідея stride полягає в тому, щоб пропустити деякі області, над якими ковзає ядро. Stride 1 означає, що беруться прольоти через піксель, тобто за фактом кожен проліт є стандартною згорткою. Stride 2 означає, що прольоти відбуваються через кожні два пікселя, пропускаючи всі інші прольоти в процесі і зменшуючи їх кількість приблизно в 2 рази, stride 3 означає пропуск 3х пікселів, скорочуючи кількість в 3 рази тощо.

Операція субдискретизації або операція pooling виконує зменшення розмірності сформованих карт ознак. У даній архітектурі вважається, що інформація про факт наявності шуканої ознаки важливіше точного знання її координат, тому з кількох сусідніх нейронів карти ознак вибирається максимальний і приймається за один нейрон ущільненої карти ознак меншої розмірності. За рахунок цієї операції, крім прискорення подальших обчислень, мережа стає більш інваріантною до масштабу вхідного зображення.

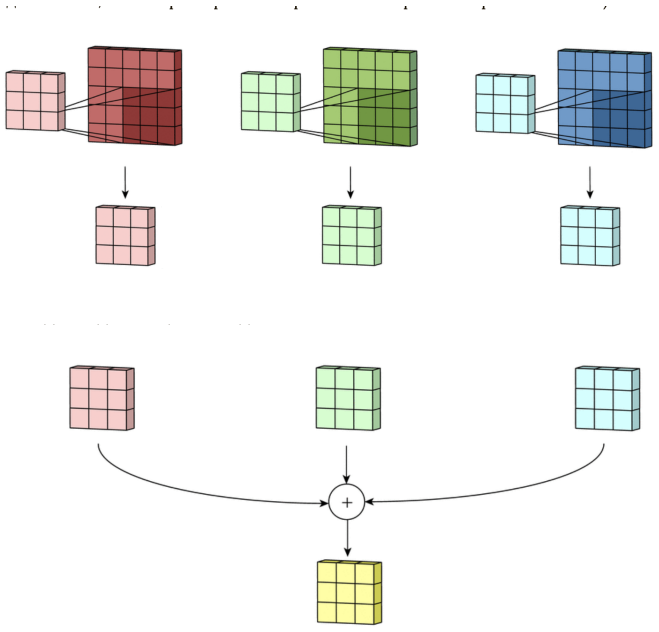
Більш сучасні мережі, такі як архітектура ResNet, повністю відмовляються від pooling шарів у внутрішніх шарах на користь чергування згорток, коли необхідно зменшити розмір на виході.

Також існує так званий шар активації, на якому скалярний результат кожної згортки потрапляє на функцію активації, яка представляє собою певну нелінійну функцію. Шар активації зазвичай логічно пов'язують з шаром згортки, тобто вважають, що функція активації вбудована в шар згортки. Функцію нелінійності можна обрати будь-яку гладку функцію з монотонною похідною, але традиційно для цього використовували функції типу гіперболічного тангенса $(f(x) = \tanh(x), f(x) = |\tanh(x)|)$ або сигмоїди $(f(x) = (1 + e^{-x})^{-1}).$. Однак в 2000х роках була запропонована і досліджена нова функція активації – ReLU (Rectified linear unit), яка дозволила суттєво прискорити процес навчання і одночасно спростити обчислення за рахунок простоти самої функції $f(x) = \max(0, x)$. Тобто це операція відсікання негативній частини скалярної величини. Станом на 2021 рік ця функція і її модифікації Noisy ReLU, Leaky ReLU та інші є найбільш часто використовуваними функціями активації в глибоких нейросетях, зокрема, в згорткових.

3.3 CNN для RGB зображень

На практиці більшість вхідних зображень мають 3 канали, і чим глибше ви в мережі, тим більше це число. Ось де ключові відмінності між термінами стають потрібними: тоді як у випадку з 1 каналом терміни “фільтр” та “ядро” взаємозамінні, в загальному випадку вони різні. Кожен фільтр насправді являє собою колекцію ядр, причому для кожного окремого вхідного каналу цього шару є одне ядро, і кожне ядро унікальне. Кожен фільтр в згортковому шарі створює тільки один вихідний канал і

робить він це так: кожне з ядр фільтра “ковзає” по їх відповідним вхідним каналам, створюючи оброблену версію кожного з них. Деякі ядра можуть мати більшу вагу, ніж інші, для того щоб приділяти більше уваги певним вхідних каналах (наприклад, фільтр може задати червоному каналу ядра більшої ваги, ніж іншим каналам, і, отже, більше реагувати на відмінності в образах з червоного каналу). Потім кожна з оброблених в каналі версій підсумовується разом для формування одного каналу. Ядра кожного фільтра генерують одну версію кожного каналу, а фільтр в цілому створює один загальний вихідний канал. Нарешті, кожен вихідний файл має своє звільнення. До вихідного каналу додається зсув для створення кінцевого вихідного каналу. Результат для будь-якої кількості фільтрів ідентичний: кожен фільтр обробляє вхід зі своїм відмінним від інших набором ядр і скалярним зміщенням, створюючи один вихідний канал. Потім вони об'єднуються разом для отримання загального виходу, причому кількість вихідних каналів дорівнює числу фільтрів. При цьому зазвичай застосовується нелінійність перед передачею входу іншого шару згортки, який потім повторює цей процес.



згортка по каналам

3.4 Алгоритми тренування мережі

Існує лише кілька алгоритмів, які можуть бути використані для тренування нейронних мереж, найефективнішим з яких є контрольований метод, який називається зворотнім розповсюдженням помилки. Ідея полягає в тому, що ми виконуємо пряму передачу по мережі і зберігаємо значення кожного нейрона. Після чого обчислюємо градієнт, який використовуємо при оновленні ваг нейронної мережі. Ми порівнюємо результат мережі з точним результатом та обчислюємо помилку за попередньо визначеною функцією помилки. Потім ми повертаємо помилку до кожного нейрона, після чого коригуємо ваги цього нейрона в процесі. По суті, ми розглядаємо, наскільки кожен нейрон відповідає за помилку цілого мережі та виправляємо цей нейрон, щоб мінімізувати його.

Основна ідея цього методу полягає в поширенні сигналів помилки від виходів мережі до її входів, в напрямку зворотному прямому поширенню сигналів у звичайному режимі роботи. Для можливості застосування методу зворотного поширення помилки передавальна функція нейронів повинна бути диференційована. Метод є зміною класичного методу градієнтного спуску.

3.5 Проблема навчання глибокої нейронної мережі

Основною перешкодою, яку потрібно було подолати під час навчання глибоких мереж, було питання збереження цілісності градієнта при зворотному розповсюдженні його через шари мережі. Це зазвичай називають проблемою “затухаючого градієнту”. Так градієнт зменшується з кожним проходячим шаром до точки, де він майже дорівнює нулю при досягненні нижніх шарів, даючи незначні оновлення параметрів на кожному новому кроці. Це призводить до того, що нейронна мережа або взагалі не вчиться, або затрачений на це час є занадто довгом. Деякі

спроби виправити це зробив Хінтон, навчаючи мережевий шар за шаром [10], поєднуючи результати для того, щоб заздалегідь навчити всю мережу спільно. Проблема затухання градієнта була значною мірою покращена із введенням блоків активації Relu Олексієм Крижевським [21]. Також з'явились різні статті, що описують належні параметри ініціалізації мережі, одна з найпопулярніших – ініціалізація Хав'єра. Також використовується така техніка як нормалізація партії [21], так вводиться шар після кожного згорткового шару, який шар, що вводиться після кожного згорткового шару, який змушує активації мати одиничний гауссовский розподіл. Практично всі сучасні глибокі нейронні мережі навчаються за допомогою алгоритмів оптимізації SGD з імпульсом. Деякі з них, в тому числі RMSprop і ADAM є найбільш використовувані.

Великий крок в адаптації нейронної мережі до різних завдань і наборів даних – це концепція трансферного навчання. Виявляється, чим нижче шари в глибокій нейронної мережі вивчають більш загальні особливості до такої міри, що перший шар майже завжди складається з кольорових світлофільтрів. Це означає, що глибоку мережу можна навчати на великому анотованому наборі даних, а потім нижні згорткові шари можуть бути повторно використані на іншій задачі. Це величезна перевага, так як існує багато завдань, таких як ця робота, де існує маленький анотований набір доступних даних і навчання на ньому з нуля дає погані результати.

3.6 Основні методи оптимізації мереж

SGD [27] оновлює кожен параметр, віднімаючи градієнт функції, яку ми оптимізуємо, по відповідному параметру й масштабуючи його на крок навчання η, який є гіперпараметром. Якщо η занадто великий, то метод буде розходитися, якщо занадто маленький – буде сходитися повільно.

$$i \sim \mathcal{U}\{1, 2, \ldots, n\}$$

$$\theta_{t+1} = \theta_t - \eta \nabla f_i(\theta_t),$$

Правило перерахунку:
де fi - функція, підрахована на і-му міні-Батчі (частини) даних, індекс і вибирається випадковим чином.

Для подальшого розуміння методу RMSprop розглянемо метод адаптивного градієнта (Adagrad) [22] ефективно перемасштабує крок навчання для кожного параметра окремо, з огляду на історію всіх минулих градієнтів для цього параметра. Це робиться шляхом ділення кожного елемента в градиенті ∇fi на квадратний корінь суми квадратів минулих відповідних елементів градієнта. Перемасштабування таким способом ефективно зменшує крок навчання для параметрів, які мають велику величину градієнта. Також метод зменшує сам крок навчання з часом, так як сума квадратів збільшується з кожною итерацией. При ініціалізації масштабирующего параметра g = 0 формула для перерахунку має вигляд:

$$g_{t+1} = g_t + \nabla f_i(\theta_t)^2$$

$$\theta_{t+1} = \theta_t - \frac{\eta \nabla f_i(\theta_t)}{\sqrt{g_{t+1} + \epsilon}},$$

де розподіл виконується поелементно, а ϵ – це невелика константа, введена для чисельної стабільності. Метод має хороші теоретичні гарантії та практичні результати [22].

Метод адаптивного змінного середнього градієнтів (RMSprop) [23] дуже схожий по принципом роботи на метод Adagrad. Єдина його відмінність в тому, що шкалирующие член g*t* обчислюється, як експоненціальне ковзне середнє замість кумулятивної суми. Це робить g*t* оцінкою другого моменту градієнта ∇f та усуває той факт, що крок навчання з часом зменшується. Правило перерахунку:

$$g_{t+1} = \gamma g_t + (1-\gamma)\nabla f_i(\theta_t)^2$$
$$\theta_{t+1} = \theta_t - \frac{\eta \nabla f_i(\theta_t)}{\sqrt{g_{t+1} + \epsilon}}$$

Метод адаптивної інерції (Adam) [24] схожий на попередні методи Adagrad та RMSprop. Відрізняється він від них двома ідеями. По-перше, оцінка перших кроків обчислюється як ковзне середнє. По-друге, через те, що оцінки першого і другого моментів ініціалізуються нулями, використовується невелика корекція, щоб результуючі оцінки не були зміщені до нуля. Метод також інваріантний до масштабування градієнтів. При заданих гіперпараметрах γ1, γ2, λ, η і m0 = 0, g0 = 0 правило перерахунку наступне: [24]

$$m_{t+1} = \gamma_1 m_t + (1-\gamma_1)\nabla f_i(\theta_t)$$
$$g_{t+1} = \gamma_2 g_t + (1-\gamma_2)\nabla f_i(\theta_t)^2$$
$$\hat{m}_{t+1} = \frac{m_{t+1}}{1-\gamma_1^{t+1}}$$
$$\hat{g}_{t+1} = \frac{m_{t+1}}{1-\gamma_2^{t+1}}$$
$$\theta_{t+1} = \theta_t - \frac{\eta \hat{m}_{t+1}}{\sqrt{\hat{g}_{t+1} + \epsilon}}$$

3.7 Сучасне виявлення об'єктів

У цьому розділі ми розглянемо деякі сучасні методи, які використовують глибокі нейронні мережі як класифікатори для виявлення об'єктів на зображеннях, а також методи, які повністю використовують архітектуру та переваги наявності потужної екстрактор функції.

Класична проста техніка локалізації об'єкта в комп’ютерному зорі полягає в простому порівнянні вікон різного розміру у всіх можливих місцях на зображенні і класифікуванні цього поля на кожному кроці. Це дає результат кратного вікна, звані обмежувальними рамками, які потім можуть бути оброблені після алгоритму, такий як NMS для усунення перекриттів. Використання глибокої нейронної мережі як потужного класифікатора для кожного вікна, приводить до надійних результатів, які показав Overfeat у 2013 р. [18], який отримав оптимальні на сьогодні результати щодо проблеми локалізації та виявлення об’єктів ILSVRC. Очевидним недоліком цього методу є дуже висока вартість виконання глибокої нейронної мережі для сотень або тисяч вікон на одному зображенні. Overfeat [18] покращує це шляхом перетворення повністю зв’язаного шару нейронної мережі в згортковий рівень, що дозволяє мережі працювати на вікнах різних розмірів. Таким чином, для даної позиції ми можемо провести класифікацію на обмежувальних полях різних розмірів ефективно, повторно використовуючи обчислювані карти об’єктів, таким чином роблячи спільні обчислення. На жаль, даний підхід не використовує повний потенціал глибокої нейронної мережі і досить швидко був затьмарений іншими більш потужними методами, такими як R-CNN [17].

Для одиничної або фіксованої кількості об’єктів на зображенні можна сформулювати проблему локалізації для задачі регресії [14]. Ідея полягає в тому, що ми прикріплюємо регресію до останнього згорткового

шару в глибокій нейронній мережі та використовуємо її для регресії координат і розмірів фіксованої кількості обмежують прямокутників, а іншу частину навченої мережі ми використовуємо для класифікування об'єкта, присутнього в кожному з цих вікон. Недоліком такого підходу є складність адаптації до зміни кількості наявних об’єктів на зображеннях. Тоді почали використовувати підхід YOLO (англ. you only look once) [16], який використовує подібний прийом, знаходячи за допомогою регресії фіксовану кількість прямокутників та класифікує об’єкти всередині них не для всього зображення, а для кожного окремого квадрата у фіксованій сітці на зображенні. Потім результати фільтруються і агрегується для отримання кінцевих обмежувальних вікон.

Ймовірно, найбільш ефективним і точним методом виявлення об'єктів є метод, заснований на регіональних згортальних мережах R-CNN [9]. Ідея полягає в тому, що ми використовуємо метод пропозиції області, такий як EdgeBoxes або Selective Search [13], щоб знайти області інтересу (ROI) на зображенні, які ми б перетворили за фіксованим розміром, після чого використовуючи глибоку нейронну мережу для прогнозування класу запропонованого об’єкта. Потім цей метод був поліпшений за рахунок використання мережі для згортки всього зображення разом [8], створення карти функцій більш високого рівня, створюючи карту класів більш високого рівня, після чого використовуємо спроектовану область на карті для класифікації об’єкта. Незважаючи на те, що такий алгоритм спільно обчислює лише карту об'єктів всього зображення один раз, ми виявляємо, що вузьким місцем методу тоді є алгоритми пропозицій регіону. Остання зміна алгоритму для покращення швидкості його роботи була названа faster R-CNN [17], використовує мережу пропозицій регіонів [17] поверх згорткової карти ознак для прогнозування оцінок для якірних ящиків, які представляють собою фіксовану кількість n вікон різних форм і розмірів.

Ці блоки потім використовуються як регіони, які класифікуються класифікатором верхнього рівня. Цей метод повністю виключає необхідність пропозиції зовнішніх областей [17] і вміло використовує гнучкість архітектури нейронної мережі, забезпечуючи продуктивність практично в реальному часі.

Інший метод виявлення об’єктів – це сегментарний підхід. Ідея полягає в тому, що вводиться все зображення в нейронну мережу і прогнозуємо карту міток для кожного пікселя. Є кілька різних підходів для цього. Один з них – згортка всього зображення для одночасного одержання прогнозів пікселів в різних масштабах, а потім підвищення дискретизації і об'єднання результатів для створення остаточної карти пікселів. При використанні нейронної мережі для прогнозування просторової інформації отримуємо карту значно меншого розміру, ніж вхідне зображення, тому що до мережі вводяться шари субдискретизації. Автори однієї роботи [15] використовують дані шари для реконструкції зображення за ознаками високого рівня. Вони також використовують переваги так званих skip-connections [15], які є методами використання просторової інформації з нижчих шарів, які мають більш високу роздільну здатність, а також семантичної інформації з мешною роздільною здатністю з шарів вищого рівня для побудови прогнозу, вперше такий підхід було використано у нейронній мережі VGG. Тобто мережу VGG використовують для вилучення високоякісних ознак із зображення, а перевернуту VGG для реконструкції семантичної карти за допомогою роз’єднання, тобто повторного використання просторової інформації з карт активації об’єднання, та згортання-транспонування.

Розширенням семантичної сегментації є сегментація екземплярів, де ми зацікавлені в сегментуванні та маркуванні окремих екземплярів певних об’єктів на сцені. Так робота, яка виграла MS COCO 2015, працює шляхом

виявлення екземплярів об’єктів на сцені за допомогою нейронної мережею регіональних пропозицій, як faster R-CNN, а потім генерує маску та сегментує окремі екземпляри. Методи семантичної та екземплярної сегментації є значно складнішими та вимагають набагато більшої роботи, також вони є досить чутливими до навчальних наборів.

4 Практична частина

4.1 Дистанційне зондування

Дистанційне зондування – це отримання інформації про об’єкт без безпосереднього контакту, зазвичай за допомогою сонарного або електромагнітного випромінювання. Говорячи про супутникові знімки, ми маємо на увазі категорію дистанційного зондування з назвою “пасивне зондування” за допомогою електромагнітного спектра. Це означає, що ми не розсилаємо жодних сигналів чи інформації, а просто спостерігаємо відбиття сонячного випромінювання за допомогою спеціалізованих датчиків, встановлених на супутнику. Супутникові знімки знайшли безліч застосувань як у вивченні Землі астрономами, так і в геології, в економічних, соціальних та військових розвідних галузях.

Супутники дистанційного зондування, такі як GeoEye та WorldView, виведені на низькоземну орбіту, яка знаходиться на відстані приблизно 600-800 км від поверхні. Супутники розгорнуті на кругових сонячно-синхронних приполярних орбітах, що означає, що вони перетинають екватор кілька разів на день і залишаються приблизно під одним і тим же кутом по відношенню до Сонця, що робить їх ідеальними для знімків Землі. На цій відстані орбітальні періоди становлять приблизно 90 хвилин, і супутники можуть повертатися до тих самих точок Землі з інтервалом у 1,1-3,7 доби.

Дані супутники обладнані найсучаснішими технологіями сенсорів та стабілізації, які дозволяють отримувати точні високоточні зображення потрібного місця. Бортові датчики складаються з мультиспектральних RGB-систем, панхроматичних датчиків, як правило, з більш високою роздільною здатністю, ІЧ-датчиків, які дають додаткову температурну

інформацію, і навіть супер- і гіперспектральних систем з понад 10 та 100 каналами відповідно для фіксації точних геологічних змін. Роздільна здатність датчиків зазвичай варіюється від 0,5 до 1,6 м на піксель, при цьому 0,5 м є законним обмеженням у більшості країн, за винятком уряду США, який дозволяє отримувати зображення з безмежною точністю та виставляти на комерційний продаж зображень з роздільною здатністю на 0,25 м панхроматично. Додаткова інформація з каналів може потенційно полегшити завдання розпізнавання об’єктів, але, на жаль, такі дані важче адаптувати через обмеження трансферів навчання. Анотовані набори даних отримати важко і дорого, також більшість з них навіть недоступні для широкої більшості людей. Тому просто не існує глибинних нейронних мереж, які навчаються на багатоканальних входах, крім RGB та градацій сірого.

4.2 Постановка задачі

У даній роботі виконаємо задачу класифікації об’єктів а потім й їх детекції на супутникових знімках. Будемо намагатися виявити транспортні засоби на зображеннях, після чого будемо їх класифікувати та знаходити їх точне розташування на фотографії.

Основний спосіб вирішення такої задачі комп’ютерного зору – використання техніки зсування вікна. Ідея полягає в тому, що ми просуваємо вікно по зображенню з фіксованим кроком. На кожній ітерації ми витягуємо патч, що відповідає розташуванню вікна на зображенні, і використовуємо класифікатор для класифікації цього патчу. Якщо він класифікується як об’єкт інтересу, тоді ми просто заявляємо, що об'єкт інтересу існує в цій групі. Зазвичай для виявлення об'єктів, що цікавлять у різних масштабах і позах, застосовуються вікна кількох розмірів та пропорцій. Можна сказати, цей метод в основному є грубим підходом,

тому що ми пробуємо вікна різного розміру і намагаємося скрізь їх використати. Проблемою даного методу є те, що занадто маленьке вікно не вміщує всередині великі об’єкти і не має достатньої контекстної інформації, тоді як занадто велике вікно дає занадто багато інформації, що вводить модель у неоднозначність й з'являються помилки вкласифікації шуканих об'єктів.

Задачу будемо робити у два етапи. Спочатку розробимо модель класифікації, а потім й детекції об’єктів. По ходу роботи будемо використовувати такі моделі як VGG16 та FRCNN. Також у кінці роботи розглянемо й реалізацію YOLO моделі, як загальноприйнятого стандарту відносно якості детекції об’єктів та швидкості роботи самої моделі.

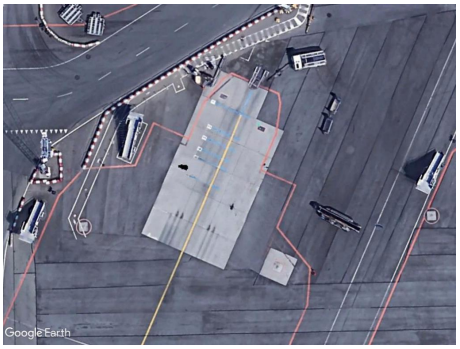
Для даної задачі використаємо датасет Satellite Imagery Multi-vehicles Dataset (SIMD), яких розглянемо більш детально у наступному пункті роботи.

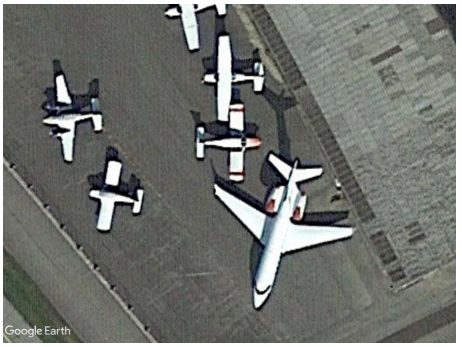
4.3 Опис датасету

Satellite Imagery Multi-vehicles Dataset складається з 5000 зображень роздільної здатності 1024 x 768 та колективно містить 45,303 об’єктів розподілених на 15 різних класах транспортних засобів, включаючи легкові автомобілі, вантажні автомобілі, автобуси, різні види літаків, гелікоптерів та човнів. Вихідні зображення взяті з загальнодоступних супутникових зображень.

З кожним зображенням доступна анотація, яка міститься як окремий текстовий файл. Формат анотації можна описати як (C, Xi, Yi, W, H), де C – це клас об'єкта, індекси класів починаються з 0, (Xi, Yi) – координати, які вказують на центр об’єкта, а (W, H) – це ширина та висота відповідно. Всі значення даних анотацій записані у форматі відсотків до даного

зображення, таким чином навчена модель зможе шукати об’єкти на знімках відмінного розміру від представлених для навчання.





приклад зображень обраного датасету SIMD

4.4 Використані технології

Для написання нейронної мережі була використана мова програмування python, фреймворк TensorFlow, бібліотека Keras, яка є надбудовою над TensorFlow. Представимо опис роботи даного фреймворку у парі з Keras.

TensorFlow – це фреймворк машинного навчання з відкритим кодом від Google, який дозволяє користувачеві швидко та ефективно впроваджувати різні алгоритми. Існують й інші відомі й широко використовувані фреймворки, наприклад Torch. Але Tensorflow надає найширший вибір функцій та операцій, починаючи від простих згорток і закінчуючи цілими готовими оптимізаторами, які можна застосувати та адаптувати в будь-якому місці за допомогою простого їх підключенню в модель в один рядок коду. Також фреймворк від Google надає можливість легкої паралелізації обчислень, що стає можливим завдяки архітектурі фреймворку.

Також TensorFlow використовує інший підхід до стандартної парадигми імперативного програмування, де користувач пише код, який

виконується послідовно. Натомість у даному фреймворку користувач оголошує обчислювальну архітектуру як символічні операції та рівняння, які потім компілюються в обчислювальний графік, а потім запускаються стільки разів, скільки потрібно. Цей тип обчислень вже реалізований у Theano [5] та вдосконалений у TensorFlow.

Такий підхід дає можливість отримувати автоматичні градієнти на будь-якому краю графіка щодо будь-якого іншого, оскільки градієнтний потік між основними математичними операціями може бути визначений за простими правилами. Це величезна перевага для числових методів, які вимагають градієнтів, які використовуються при навчанні нейронних мереж.

Keras – це високорівневий API TensorFlow: доступний, високопродуктивний інтерфейс для вирішення проблем машинного навчання з акцентом на сучасне глибоке навчання. Він забезпечує основні абстракції та будівельні блоки для розробки та транспортування рішень для машинного навчання з високою швидкістю виконання ітерацій.

Так Keras надає можливість повністю скористатися масштабованістю та крос-платформними можливостями TensorFlow.

Отже, TensorFlow обширний арсенал функцій для написання нейронних мереж, а Keras полегшує використання фреймворку й робить розробку моделі швидшою.

4.5 Трансфертне навчання

Поняття трансфертного навчання існує вже понад 2 десятиліття у сфері машинного навчання і в основному описує здатність використовувати інформацію, отриману в результаті одного завдання, та застосовувати її до іншого. При глибокому навчанні це використовується

дуже часто. Було виявлено, що функції, які вивчає ConvNet у першому шарі, завжди однакові і нагадують кольорові крапки та фільтри Габора. Це означає, що особливості низького рівня є загальними, і зрозуміло, що чим вище ми піднімаємося по мережі, тим більш специфічними є завдання знаходження ознак для класифікації об’єктів.

Ідея використання трансфертного навчання в процесі глибокого навчання полягає в наступному: ми спочатку навчаємо нейронну мережу на високоякісному та великому наборі даних, такому як ImageNet. Це змушує мережу вивчати широкий спектр функцій, які допомагають їй виконувати поставленні перед нею задачі класифікації. Далі ми припускаємо, що функції нижчих рівнів мережі є загальними і можуть бути “перенесені” на наше завдання. Після чого використовуємо нижню частину навченої мережі як екстрактор функцій, поверх якого ми можемо навчити іншу нейронну мережу або лінійний класифікатор, такий як SVM, для інших завдань регресії або класифікації.

Ефективність даного способу є не раз перевіреною теоретично та практично при навчанні різних моделей глибоких нейронних мереж. Так, наприклад, робота Разавіана показує сучасні результати використання цієї техніки, які значно прискорюють навчання мережі, що в свою чергу дає змогу нам використовувати більш складні архітектурні рішення щодо їх побудови або дає змогу навчати модель на менш сучасному апаратному забезпеченні.

Для швидкого й точного створення практичної моделі будемо використовувати дану техніку при розробці нейронної мережі.

4.6 Реалізація моделі

У цій дипломній роботі було вирішено використовувати 16-шарову мережу VGG [19] як екстрактор функцій представленної моделі. VGG16 має помилку близько 8,5% на датасеті ILSVRC. Це приблизно на 1% вище, ніж 19-шарової версії даної глибинної мережі, але для полегшення обробки та швидкості обчислення була використана саме 16-шарова для подальшого застосування в FRCNN.

Історично VGG був обраний серед AlexNet та інших архітектур за свою простоту, обчислювальну гнучкість та глибину, що дає великий шанс отримати більш загальні ознаки для подальшого використання у класифікації та детекції об’єктів, а саме транспортних засобів.

Мережа VGG пройшла навчання оригінальними авторами на вищезазначеному наборі даних ILSVRC від ImageNet, який складається з 1,3 мільйона зображень, розділених на 1000 класів, що робить цей датасет хорошим екстрактором функцій. Автори також використовували різні типи збільшення зображень під час тренувань.

Наша система виявлення об'єктів, що називається Faster RCNN, складається з двох модулів. Перший модуль – це вищеописана глибока повністю згорнута мережа VGG16, яка пропонує регіони, а другий модуль – це детектор Fast RCNN, який використовує запропоновані регіони. Вся система являє собою єдину уніфіковану мережу для виявлення об'єктів.

Так у змаганнях ILSVRC та COCO 2015 FRCNN посіла 1-е місце у класифікації, локалізації та сегментації об’єктів на датасеті ImageNet, також забрала 1-е місце й на наборі даних COCO. Результатти представлених змагань свідчать про те, що FRCNN є не лише економічно ефективним рішенням для практичного використання, але й ефективним способом підвищення точності виявлення об’єктів.

Представлена архітектура навчається за допомогою оптимізатора градієнтного спуску з імпульсом. Традиційно використовується алгоритм SGD для оптимізації нейронних мереж, але для більш глибоких мереж стає важливим додавати імпульс до моделі оптимізації. TensorFlow пропонує нам ефективні готові оптимізатори, які ми можемо використовувати для мінімізації функції витрат. Для практичного завдання був використаний алгоритм ADAM [20]. ADAM – це оптимізатор, який використовує SGD з адаптивним імпульсом й був описаний у першому розділі.

Для обраної нейронної мережі використовуємо зворотне розповсюдження. Так під час кожної ітерації алгоритму ми виконуємо прямий прохід на навчальному прикладі та спостерігаємо вплив кожного нейрона на вихідну помилку. Далі ми обчислюємо градієнт відносно виходу, а потім спільно оновлюємо всі нейрони. Правило оновлення для кожного нейрона показано у рівнянні нижче, де *wij* – *ij*-та вага між двома нейронами *i*, *j* у двох сусідніх шарах, α – швидкість навчання та *E* – функція витрат.

$$w_{ij} = w_{ij} - \alpha \cdot \frac{\partial E}{\partial w_{ij}}$$

Помилка обчислюється за формулою середньоквадратичної помилки або MSE. Вона дає загальне уявлення про те, наскільки точно нейронна мережа працює.

$$E = \frac{1}{k} \sum_{N} (\sum_{m} \sum_{n} (P^{j}_{s_{m,n}} - G^{j}_{s_{m,n}})^2)$$

Слід також зазначити, що обчислення градієнту для кожного нейрона окремо є неефективним, тому перетворення між шарами виконується за

допомогою множення щільної матриці, які можна ефективно обчислити та розпаралелювати як на процесорі, так і на графічному процесорі.

4.7 Опис моделі YOLO

Обрана нейронна мережа FRCNN переробляє класифікатор VGG16 для здійснення детекції об’єктів. Так модель застосовується до зображення в різних місцях та масштабах. Області з високим балом зображення вважаються виявленнями. Для оцінки ефективності порівняємо FRCNN з YOLO версії 3, тому що ця модель є наступним кроком у розвитку методів детекції об’єктів.

Так YOLO використовує зовсім інший підхід. А саме, застосовується єдина нейронна мережа до всього зображення одночасно. Ця мережа ділить зображення на регіони і передбачає обмежувальні рамки та ймовірності для кожного регіону, де обрані рамки зважуються за прогнозованими ймовірностями.

YOLO має кілька переваг перед системами на основі класифікаторів. Оскільки розглядається ціле зображення під час тестування, тому його прогнози визначаються загальним контекстом зображення. Також, очевидно, вони робляться за допомогою єдиної оцінки мережі, на відміну, наприклад, від представленної мережі FRCNN, яка вимагає тисячі оцінок для одного великого зображення. Це робить YOLO надзвичайно швидкою моделью детекції об’єктів. Так, автори моделі заявляють швидкість у 100 більшу ніж у FRCNN.

Модифікація YOLOv3 використовує мережу Darknet-53 для класифікації та детекції об’єктів. Вона використовує більше послідовних згорткових шарів 3 × 3 та 1 × 1, ніж Darknet-19 у YOLOv2, та організовує їх як залишкові блоки [29]. Отже, Darknet-53 набагато потужніший, ніж

Darknet-19. Також YOLOv3 передбачає обмежувальні рамки в трьох різних масштабах, слідуючи ідеї функціональної пірамідної мережі для виявлення об'єктів [34]. Три заголовки виявлення, окремо побудовані поверх трьох функціональних карт з різними масштабами, відповідають за виявлення об'єктів різного розміру.

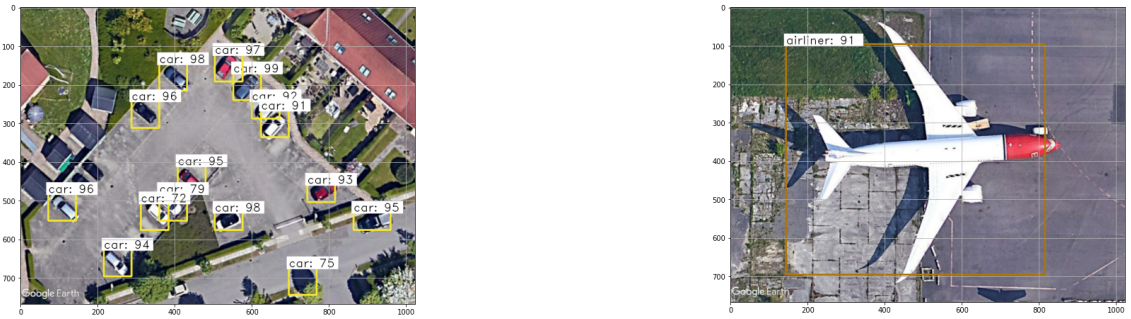
Для порівняння з FRCNN будемо використовувати вже готову модель YOLO, яку потрібно лише довчити на обраному датасеті.

4.8 Результати тренованої моделі

Після навчання описаної вище моделі FRCNN отримали нижче наведені результати.

Модель	**Validation mAP**	**Test mAP**
FRCNN	0.515	0.508
YOLOv3	0.608	0.634

де AP надає показник якості на всіх рівнях класифікації для одного класу. Тоді mAP є середнім значенням AP у багатокласовій класифікації.



Приклади роботи нейронної мережі FRCNN на обраному датасеті.

З наведених значень можна побачити, що YOLOv3 є набагато точнішою моделью. Але вона має 106 згорткових шарів, звідси слідує й велика кількість параметрів, які потрібно знайти. Отже, й для тренування потрібна велика потужність системи, на якій проводяться розрахунки. Звісно представлена FRCNN буде знаходити об’єкти значно довше й з меншою точністью, але натренувати її можна, маючи значно менші ресурси й складність та комплексність самої моделі набагато легша, а тому з нею легше працювати.

Отже, для не професійного використання або для задач, де швидкість роботи моделі та її точність не є досить критичними, можна використовувати описану у роботі FRCNN.

5 Висновок

У цій роботі було продемонстровано успішну спробу спроектувати та навчити за допомогою TensorFlow та Keras нейронну мережу FRCNN для виявлення об'єктів на супутникових знімках високої роздільної здатності та порівняний з однією з кращих на сьогодні моделей YOLO версії 3.

Так було зазначено, що FRCNN є досить тоною моделью, а також, на відміну від YOLO для якої потрібна велика потужність, представлена мережа може бути навчена у “домашніх” умовах, особливо, якщо для частини класифікатора VGG16 використовувати трансфертне навчання. Також слід зазначити, що за таким принципом можна використовувати вже готові моделі YOLO, які потрібно лише довчити на обраному датасеті, а не навчати з нуля.

При описі та навчанні FRCNN було показано нерозривний зв’язок між дослідженнями у виявленні об’єктів з іншими техніками комп’ютерного зору, такими як розпізнавання зображень та сегментація зображень, оскільки вони допомагають розуміти та аналізувати сцени на зображеннях чи відео. Але є важливі відмінності між ними. Так розпізнавання зображень виводить лише мітку класу для ідентифікованого об’єкта, а сегментація зображення створює піксельне розуміння елементів сцени. Детекція ж займається саме знаходженням певних об’єктів серед інших на зображенні чи відео, що потім дозволяє відстежувати їх. Беручи до уваги ці ключові відмінності та унікальні можливості виявлення об’єктів, можна побачити різні види застосування детекції об’єктів: підрахунок натовпу, самокеровані машини, відеоспостереження, виявлення обличчя, виявлення аномалій. Звичайно, це не вичерпний перелік, але він

включає деякі основні способи, за допомогою яких виявлення об’єктів формує наше майбутнє.

Дійсно, виявлення об’єктів є ключовим завданням для більшості комп’ютерних та роботологічних систем зору. Незважаючи на те, що за останні кілька років було досягнуто значного прогресу, у майбутньому відбудуться ще більші вдосконалення з появою штучного інтелекту в поєднанні з існуючими техніками, які зараз є частиною багатьох побутових електроніків або інтегровані в асистентські технології водіння.

Крім того, виявлення об'єктів не застосовувалось у багатьох областях, де це могло б бути дуже корисно. Наприклад, можливість застосування систем виявлення об'єктів для роботизованих розкопок при заході на раніше не досліджену територію, таку як глибоке море або інші планети, в яких системи виявлення повинні вивчати нові класи об’єктів. У таких випадках надзвичайно важлива здатність до навчання в режимі реального часу.

Майбутнє технології виявлення об’єктів доводить себе, і, подібно до оригінальної Промислової революції, вона має потенціал звільнити людей від важкої роботи, яку можна більш ефективно та ефективно виконувати за допомогою машин. Це також відкриє нові шляхи досліджень та операцій, які принесуть додаткові переваги в майбутньому.

Таким чином, технології детекції об’єктів перебувають у постійному розвитку, адже вони є дуже корисними у повсякденному житті людини. Можна навіть зазначити, що нові рішення даної задачі комп’ютерного зору можуть вплинути на людство не менше ніж промислова революція, але для цього потрібен розвиток не лиже самих алгоритмів для побудови моделей, а й швидкий розвиток технологій для покращення швидкості навчання нейронних мереж.

6 Список джерел

[1] Convolutional neural networks. http://d2l.ai/chapter_convolutional-neural-networks/index.html.

[2] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[3] Distributed tensorflow. https://www.tensorflow.org/guide/distributed_training.

[4] Tensorflow api. https://www.tensorflow.org/api_docs.

[5] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, and Y. Bengio. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

[6] C. Cortes and V. Vapnik. Support-vector networks. Machine Learning, 20(3):273–297, 1995.

[7] K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics, 36:193–202, 1980.

[8] R. B. Girshick. Fast r-cnn. CoRR, abs/1504.08083, 2015.

[9] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. CoRR, abs/1311.2524, 2013.

[10] G. E. Hinton, S. Osindero, and Y. W. Teh. A fast learning algorithm for deep belief nets. Neural computation, 18(7):1527–1554, 2006.

[11] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093, 2014.

[12] P. Viola and M. J. Jones. Robust real-time face detection. International Journal of Computer Vision, 57(2):137–154.

[13] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders. Selective search for object recognition. International Journal of Computer Vision, 2013.

[14] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. CoRR, abs/1312.4659, 2013.

[15] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. CoRR, abs/1411.4038, 2014.

[16] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. CoRR, abs/1506.02640, 2015.

[17] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In Advances in Neural Information Processing Systems (NIPS), 2015.

[18] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. CoRR, abs/1312.6229, 2013.

[19] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. CoRR, abs/1409.1556, 2014.

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. CoRR, abs/1412.6980, 2014.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou,

and K. Q. Weinberger, editors, Advances in Neural Information Processing Systems 25, pages 1097–1105. Curran Associates, Inc., 2012.

[22] Duchi, John, Elad Hazan, and Yoram Singer. "Adaptive subgradient methods for online learning and stochastic optimization."The Journal of Machine Learning Research 12 (2011): 2121-2159.

[23] Tieleman, Tijmen, and Geoffrey Hinton. "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude."COURSERA: Neural Networks for Machine Learning 4 (2012): 2.

[24] Kingma, Diederik, and Jimmy Ba. "Adam: A method for stochastic optimization."arXiv preprint arXiv:1412.6980 (2014).

[25] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks."Advances in neural information processing systems. 2012.

[26] Hecht-Nielsen, Robert. "Theory of the backpropagation neural network."Neural Networks, 1989. IJCNN., International Joint Conference on. IEEE, 1989.

[27] Amari, Shunichi. "A theory of adaptive pattern classifiers."Electronic Computers, IEEE Transactions on 3 (1967): 299-307.

[28] Mnih, Volodymyr, Nicolas Heess, and Alex Graves. "Recurrent models of visual attention."Advances in Neural Information Processing Systems. 2014.

[29] LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. Nature 521, 436–444 (2015).

[30] He, K., Zhang, X., Ren, S. & Sun, J. Deep Residual Learning for Image Recognition. in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 770–778 (IEEE, 2016). doi:10.1109/CVPR.2016.90.

[31] Huang, G., Liu, Z., Maaten, L. van der & Weinberger, K. Q. Densely Connected Convolutional Networks. in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2261– 2269 (IEEE, 2017). doi:10.1109/CVPR.2017.243.

[32] Ren, S., He, K., Girshick, R. & Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Trans. Pattern Anal. Mach. Intell. 39, 1137–1149 (2017).

[33] Liu, W. et al. SSD: Single Shot MultiBox Detector.

[34] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2117–2125, 2017. 2, 3.