Міністерство освіти і науки України
Національний університет «Києво-Могилянська академія»
Факультет інформатики
Кафедра мультимедійних систем

Кваліфікаційна робота

освітній ступінь – бакалавр

на тему: «**Features extraction with one-stage detectors**»

Виконав: студент 4-го року навчання,

Освітньої програми «Комп'ютерні науки», 122

Комонов Кирило Максимович

Керівник Швай Н. О.,

Старший викладач, кандидат наук

Рецензент _____

(прізвище та ініціали)

Кваліфікаційна робота захищена

з оцінкою

_____

Секретар ЕК _____

«____» _____ 20____ р.

Київ – 2022

ЗМІСТ

Умовні позначення

- Виділення ознак (feature extraction)
- Виявлення об’єктів (objects detection)
- GPU (General Processing Unit) - графічний процесор
- CPU (Central Processing Unit) - центральний процесор
- CNN (Convolutional Neural Network) - згорткова нейронна мережа
- Рівень (layer) - структурна частина нейронної мережі
- Одноетапний детектор (one-stage detector) - один з видів нейронної мережі для виявлення об’єктів
- Двоетапний детектор (two-stage detector) - один з видів нейронної мережі для виявлення об’єктів
- Зразковий датасет (benchmark dataset) - датасет з розміченими даними, що використовується для оцінки ефективності різних моделей
- Обмежуючий прямокутник (bounding box) – прямокутник, що виділяє якусь частину зображення, зазвичай вказується 4-чисельними значеннями
- FC рівень (Fully connected layer)
- Нода – нейрон
- Багатошарова нейронна мережа прямого поширення (Feedforward Multilayer Neural Network)
- Backpropagation – зворотне поширення
- MSN (Mean Square Error) – середньо-квадратична похибка
- depth slide (слайд по глибині) – частина нейронів згорткової нейронної мережі, що мають однакові значення координати по осі глибини (в середині рівня мережі)
- ReLU (Rectified linear unit) – активаційна функція, що відкидає всі від’ємні значення і замінює їх нулем, залишаючи позитивні незмінними
- Max Pooling (Максимізаційне агрегування)
- Обмежуючий прямокутник (bounding box)

- Ground bounding box – обмежуючий прямокутник для об’єкта, що є правильним.
- Зразковий прямокутник (default bounding box) – термін, що використовується у статті [3]

ВСТУП

Процес виділення ознак є одним з етапів виявлення об’єктів на зображенні чи відео. Виявлення об’єктів є одною з основних задач в сфері комп’ютерного зору. Виявлення об’єктів є дуже перспективною зоною досліджень, бо поєднує велику цінність кінцевого результату (тобто алгоритмів для виявлення різних об’єктів на зображенні) і порівняну молодість цієї галузі знань.

Виявленням об’єктів називають процес аналізу зображення таким чином, щоб алгоритм «зрозумів», які предмети зображені на фотографії і де вони там знаходяться. Метою і завданням алгоритмів комп’ютерного зору є надання можливості комп’ютерним системам отримування інформації про навколишнє середовище шляхом подібним до людського бачення. Завдання аналізу навколишнього середовища через світлові сенсори є рутиною для людини (зір), у той же час є непростою задачею для комп’ютерних систем.

Крім цього існує низка задач, які раніше могли виконуватись лише кваліфікованими спеціалістами, а зі створенням ефективних алгоритмів комп’ютерного зору виконуються комп’ютером. До такого типу завдань відноситься аналіз знімків у медичній сфері. Так, під час медичного обстеження можуть бути згенеровані рентгенографічні, томографічні і інші типи знімків, що необхідні для встановлення правильного діагнозу. Сучасні методи комп’ютерного зору дозволяють автоматично аналізувати такі знімки і спрощувати роботу медичних працівників.

Розвиток GPU у останні роки привів до того, що стало можливим тренування моделей для глибокого навчання на порядок швидше ніж це було можливим на звичайному процесорі (CPU). Саме це стало причиною для різкого зросту зацікавленості у алгоритмах штучного інтелекту.

Структура більшості моделей глибокого навчання передбачає на вхід ноди з даними, що репрезентують основу для роботи алгоритму. Тобто для завдань комп’ютерного зору це дані зображення. Одне кольорове зображення розміру

N*K пікселів репрезентується 3*N*K нодами. Враховуючи те, що складність навчання моделі залежить від кількості нод на різних рівнях моделі, зв'язків між ними, виникає необхідність зменшити кількість нод на вході нейронної мережі.

Крім цього зображення містить чимало надлишкової для задачі виявлення об'єктів інформації. Немало пікселів – це просто фон. Саме тому, перші рівні нейронної мережі мають на меті виділити корисні ознаки із зображення. Таким чином кількість нод зменшується, а корисна інформація майже не втрачається.

Більшість моделей для виявлення об'єктів використовує CNN рівні для виділення ознак зображення. Моделі, що мають на меті виявлення об'єктів, поділяють на одноетапні і двоетапні. Вони мають відмінності у структурі і відмінності у характеристиках. Однією з них є те, що одноетапні моделі часто мають швидшу роботу, яка компенсується деякою втратою точності.

Як було вказано вище, ця галузь є доволі молодою і вона активно розвивається. Кожен рік моделі вдосконалюються, їм на противагу з'являються конкурентні підходи. Різні моделі «змагаються» у тому, наскільки точно і швидко вони працюються на зразкових датасетах, таких як PASCAL VOC, ImageNet, Microsoft COCO. Ці датасети містять в собі тисячі зображень з відміченими людьми об'єктами.

Результатом виявлення об'єктів зазвичай є набір обмежуючих прямокутників з числовою характеристикою для кожного з них, що виражає ймовірність того, що у зоні зображення, обмеженій прямокутником, зображено об'єкт (будь який) чи об'єкт певного класу.

Характеристики за якими «змагаються» моделі розраховуються з урахуванням помилок у виділенні обмежуючих прямокутників і помилок класифікації об'єктів всередині них.

Мета: оглянути існуючі моделі одноетапних детекторів в розрізі їх підходів до виділення ознак і не тільки.
Мета роботи зумовила наступні завдання:

1. Оглянути найвідоміші моделі одноетапних детекторів і використані ними способи виділення ознак.
2. Дізнатись, які метрики використовують для оцінки якості моделей.
3. Створити порівняльну характеристику основних одноетапних детекторів, використовуючи релевантні метрики.

Об’єкт дослідження: порівняльна характеристика основних моделей одноетапних детекторів

Робота складається з 3 розділів.

Перший складається з огляду ключових компонентів, що використовуються у проектуванні структур основних моделей, що розглядатимуться у цій роботі.

Другий розділ описує будову основних архітектур одноетапних детекторів об’єктів

Третій розділ описує існуючі метрики для порівняння моделей

Четвертий розділ містить порівняльну характеристику згаданих вище архітектур

1. Основні компоненти одноетапних детекторів

1.1 Нейронні мережі

1.1.1 Структура нейронної мережі з усіма FC рівнями

Нейронні мережі – це основа алгоритмів глибокого навчання. Основна ідея полягає у тому, що ми маємо нейрони і зв'язки між ними. Мережа складається з вхідного рівня нейронів, вихідного рівня нейронів і певної кількості рівнів між ними. Нейронні мережі бувають різних типів і мають різне наповнення. Розглянемо для прикладу «просту» нейронну мережу для класифікації зображення у N класів. Це багатошарова нейронна мережа прямого поширення. Припустимо зображення має розмір X*X і є кольоровим (RGB).

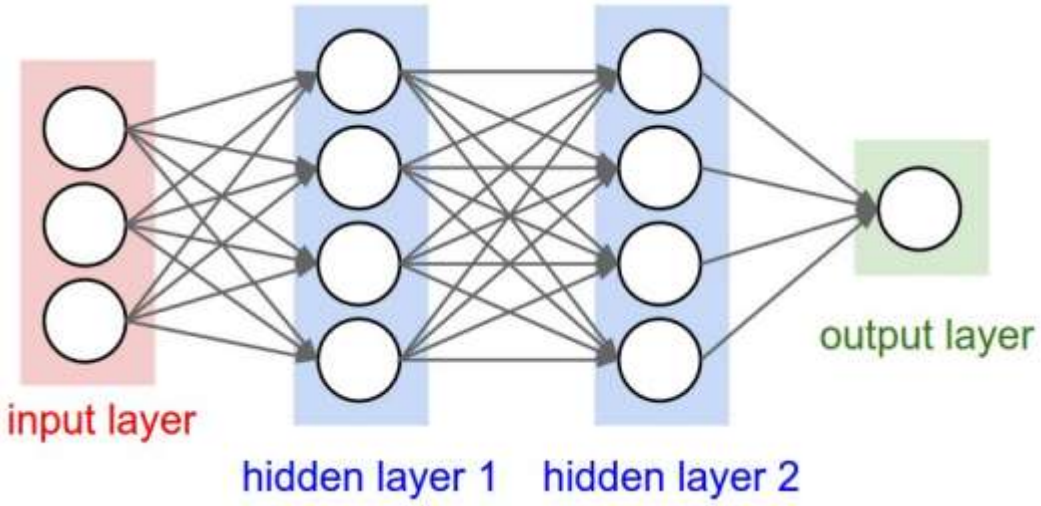

Рисунок 1 Структура нейронної мережі

Така мережа буде мати X*X*3 вхідних нейронів. Тобто один нейрон на одну чисельну характеристику одного пікселя. Всього у одного пікселя 3 канали кольору, тому і 3 характеристики. Також ця мережа матиме N вихідних нейронів. Тобто кожен вихідний нейрон відноситься до одного можливого класу.

Між нейронами сусідніх рівнів існують зважені зв'язки. Розглядаємо випадок, коли кожен з рівнів мережі є повністю зв'язаним (FC layer), тобто між всіма нейронами I-того рівня і кожним нейроном (I+1) – го рівня є зв'язок.

Кожен зв'язок має коефіцієнт ваги, про роль якого трохи нижче.

Також всі нейрони крім тих, що на вхідному рівні, мають коефіцієнт прихильності (bias).

1.1.2 Опис роботи нейронної мережі

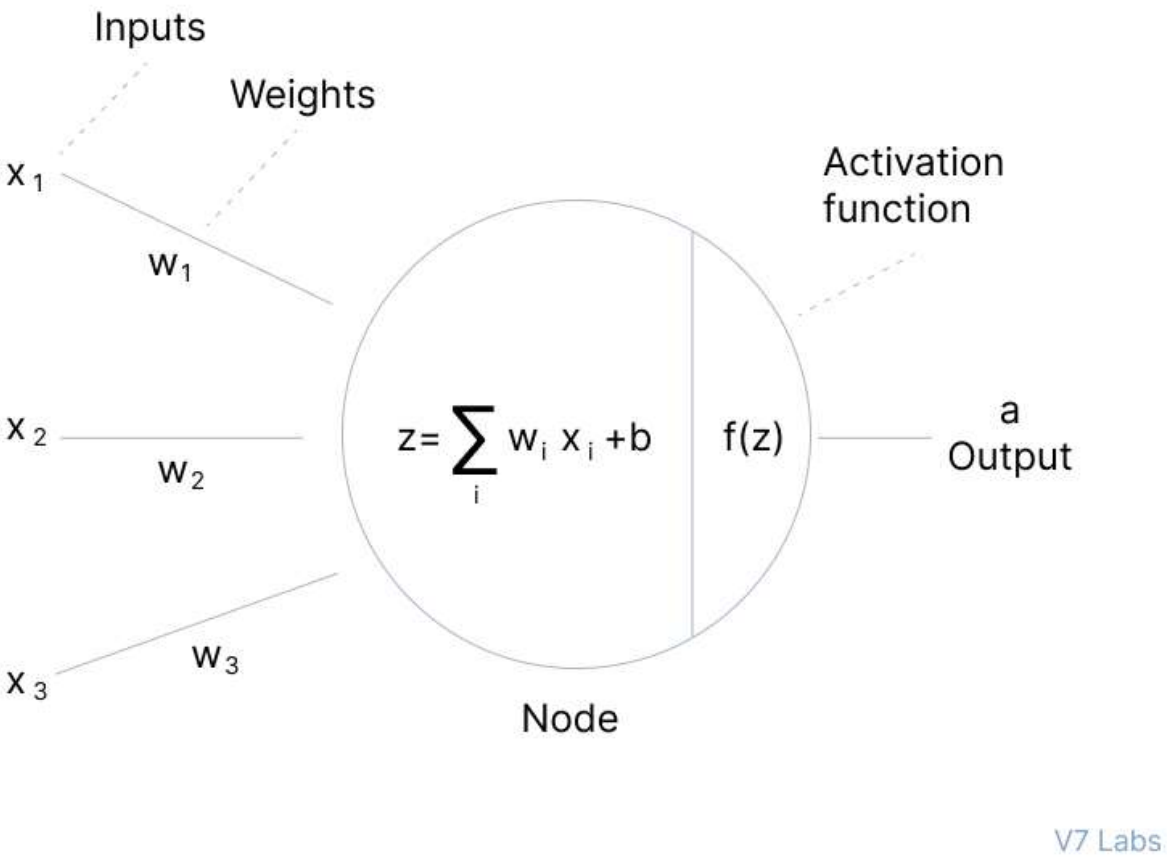

Рисунок 2 Активаційна функція

Процес класифікації відбувається наступним чином:

- Передаємо на вхідні нейрони значення кольору з зображення. Таким чином, всі нейрони на першому рівні містять числа, що описують кожен піксель зображення.
- Розраховуємо значення кожного нейрона другого рівня нейронів як результат активаційної функції, що приймає на вхід суму коефіцієнту упередженості (b) і зважену суму значень з нейронів попереднього рівня. Використовуються лише ті нейрони попереднього рівня, з якими є зв’язок, тобто в нашому варіанті – всі.
- Повторюємо процес для всіх наступних рівнів.

Як результат ми маємо якісь значення на вихідному рівні мережі. В ході тренування ми будемо намагатися змінити коефіцієнти зв’язків і коефіцієнти упередженості таким чином, щоб значення на вихідному рівні корелювали з ймовірністю того, що дане зображення належить до певного класу. Тобто клас, пов’язаний з нейроном вихідного рівня з найбільшим значенням, є найкращим кандидатом за версією моделі.

1.1.3 Активаційна функція

Вище згадана активаційна функція має важливу роль в дизайні моделі. Вона додає нелінійності системі. Тобто, якби її не було, модель з кількістю внутрішніх шарів більше ніж один, могла б бути перетворена у модель з одним внутрішнім шаром (цю модель називають single layer perceptron). Активаційна функція не повинна бути лінійною.

Існують різні функції активації, що використовуються у моделях глибокого навчання. Наприклад, сигмоїда і ReLU.

1.1.4 Навчання моделі

Навчання моделі необхідне для того, щоб вона могла правильно працювати. Тобто щоб модель «запам’ятала», зображення якого типу відносяться до певного класу, моделі спочатку необхідно показати певну кількість зображень, для яких відомі їх класи, і кожен раз трохи змінювати коефіцієнти моделі. Цей процес називається навчанням з вчителем.

1.1.4.1 Функція втрат

Для того, щоб оцінити наскільки добре модель справилась зі своїм завданням, використовується функція втрат. Вона має набувати високого значення, якщо розбіжність між реальними і передбаченими даними велика.

Класичним варіантом функції втрат є MSN. [1]
Наданий тут варіант формули, є модифікацією формули доступної за джерелом, оскільки в прикладі наведеному вище декілька вихідних нейронів а не один (як у джерелі).

$$E(X,\theta) = \frac{1}{2NM}\sum_{j=1}^{M}\sum_{i=1}^{N}\left(\widehat{y_{ij}} - y_{ij}\right)^2$$

Формула 1.1.4.1.1

Де,

X – набір пар (x_i; y_i);

x_i – вхідний вектор ознак;

y_i – правильний вектор, що має бути на виході нейронної мережі при подачі на вхід вектора x_i;

y_{ij} – j-те значення вектора y_i;

M – кількість вихідних нейронів;

N – кількість пар значень у X;

θ – множина коефіцієнтів моделі;

$\widehat{y_{ij}}$ – значення j-того нейрона на вихідному рівні розраховане моделлю при подачі на вхід вектора x_i.

У формулі 1.1.4.1.1 передбачено отримати на вхід N начальних прикладів, що необхідно при розрахунку ефективності моделі. Під час навчання модель оновлює коефіцієнти після кожного навчального прикладу, тобто під час навчання N дорівнює 1.

У цьому прикладі вектор y_i має складатися з нулів і мати лише одне ненульове (k-те) значення, що дорівнює 1. Таким чином ми репрезентуємо інформацію про те, що на i-тому зображенні знаходиться j-тий об’єкт класу k.

У моделі, описаній вище, значення на вихідних нейронах мають сенс лише у діапазоні [0; 1]. Для того щоб спростити процес навчання, можна змінити активаційну функцію останнього рівня на таку, що має потрібний нам діапазон значень. Такою, наприклад, є сигмоїда.

Замість цього можна використати функцію softmax.

1.1.4.2 Градієнтний спуск

Процес навчання відбувається наступним чином:

На вхід в модель подається навчальне зображення.

Модель розраховує ймовірності того, що це зображення відноситься до кожного з можливих класів

Ми порівнюємо правильний вектор класифікації з тим, що розрахувала модель. Ми розраховуємо значення функції втрат.

Тепер потрібно розрахувати нові значення коефіцієнтів моделі, для цього використовується наступна формула.

$$\theta^{t+1} = \theta^t - \alpha \frac{\partial E(X, \theta^t)}{\partial \theta}$$

Формула 1.1.4.2.1

Де,

θ^t – значення коефіцієнтів моделі на t-тій ітерації процесу навчання.

α – коефіцієнт швидкості навчання.

Принцип backpropagation означає, що першими будуть змінені значення коефіцієнтів у зв'язках між нодою на останньому рівні і нодами попереднього рівня. Після цього для кожної ноди, пов'язаної з попередньо обробленою нодою, буде розраховано значення помилки. Процес повторюватиметься для всіх задіяних зв'язків. Для розрахунку помилки на рівні k використовуються значення помилок на рівні k+1, ваги коефіцієнтів рівня k+1, значення похідної до функції активації. Тобто помилка розраховується, починаючи з останніх рівнів. Цей процес називають зворотним поширенням помилки.

1.2 CNN

1.2.1 Причини створення CNN

1.2.1.1 Дослідження роботи зорової кори мозку тварин

У 1959 році Девід Г'юбел та Торстен Візел провели експериментальне дослідження. У рамках цього дослідження піддослідному коту демонстрували патерни світла на екрані, у той же час записуючи дані про те, які нейрони з зорової кори мозку реагували на це.

У результаті дослідження було виявлено, що певні ділянки нейронів емітують сигнал в результаті подразників специфічного типу. А саме: лінії світла, під різними кутами зображені на екрані, і лінії світла, що рухаються у певних напрямках. Нейрони, які реагували на прості патерни світла, назвали простими клітинами (simple cell). Крім них існують і інші нейрони, які за припущенням дослідників спрацьовували, коли була необхідність розпізнати щось складніше за примітивні форми.

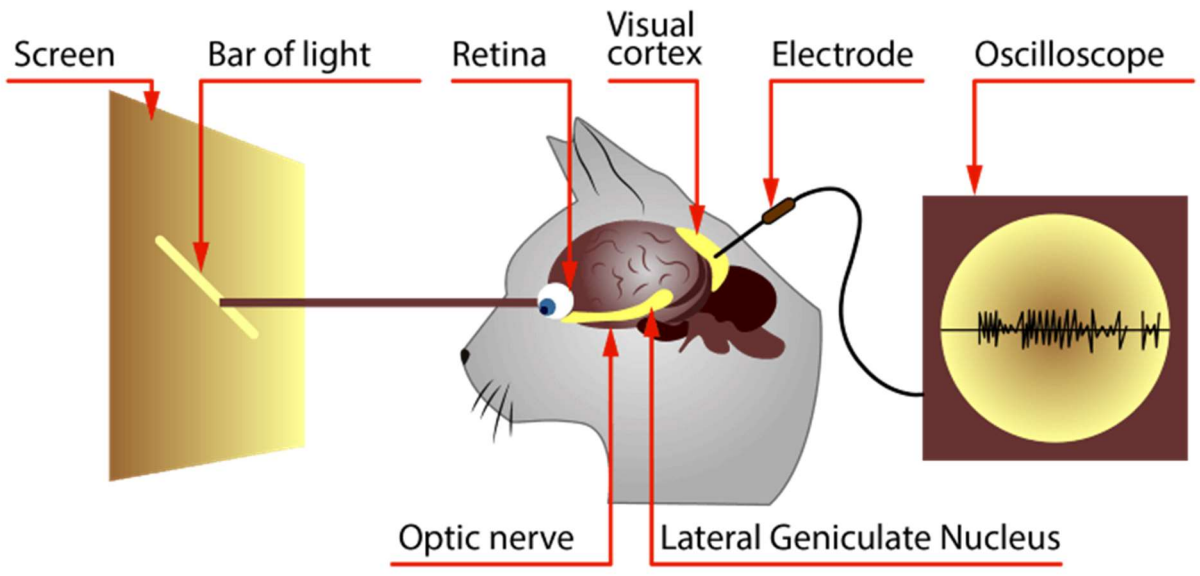

Рисунок 3 Візуалізація експерименту Д. Г'юбела і Т. Візела

Ці нейрони отримали назву складних клітин (complex cells). Було виявлено, що ці складні клітини були поєднані нейронними зв’язками з простими клітинами, що розміщуються близько один до одного.

Ці дослідження надихнули науковців розробляти алгоритми, які використовують дані, що відповідають пікселям, що знаходяться близько. Цей підхід отримав назву sparse interactions.

1.2.1.2 Вимоги еквіваріантності і інваріантності до трансляції зображення.

Еквіваріантність – це властивість нейронної мережі, яку можна описати наступним чином. Операції трансляції (зміна місцеположення об’єкта на фото) до вхідного зображення призведуть до аналогічної трансляції на рівні ознак (feature map).

Уявімо собі задачу: створити класифікатор, що зможе розпізнавати мавп на фото, незалежно від того в якій частині зображення вони знаходяться. При використанні MLP було б необхідно навчити модель на фото, де мавпи знаходяться на різних частинах фото, що сильно ускладнює процес і здається зайвим.

Причина цьому – це недосконалість MLP мереж-класифікаторів розпізнавати об’єкти, що знаходяться в різних областях зображення. Це відбувається, бо кожен нейрон MLP пов’язаний з майже усіма нейронами іншого рівня, незалежно від територіальній близькості на оригінальному зображенні.

Інваріантність означає, таку модель, яка буде не реагувати на операції трансляції, здійснені до вхідних зображень, і буде на виході вказувати клас об’єкта. Це і є опис нашого побажання мати можливість знаходити об’єкт на зображенні незалежно від його розташування.

1.2.1.3 Проблема з розмірністю даних

При розробці нейронних мереж для роботи з зображеннями слід звертати чимало уваги на кількість нейронів, необхідних для роботи моделі.

Припустимо, що ми маємо модель типу MLP, яка приймає на вхід зображення розміру 500 на 500 пікселів. Враховуючи те, що зображення кольорове і кожен піксель має бути закодованим трьома нейронами, маємо 500*500*3 = 750000 тисяч вхідних нейронів.

Припустимо, що ми вирішили використовувати один проміжний рівень з 500 нейронами. Тобто кількість коефіцієнтів зв’язків між 1 і 2 рівнем сягає 375 мільйонів. Це надзвичайно багато. Складно уявити, скільки ресурсів необхідно витратити для тренування такої моделі.
Очевидно, що для зображень хоча б розмірністю більше 100×100 пікселів, використання FC рівнів є непрактичним.

1.2.2 Структурні елементи CNN

Згорткова нейронна мережа – це новий підхід до створення моделей глибокого навчання. Він базується на припущенні, що для виявлення корисних ознак зображення достатньо створювати зв’язки лише з тими нейронами, що знаходяться близько один до одного. Через те, що інформація про те, які пікселі знаходяться близько один до одного, є ключовою для роботи моделі, вважається, що на вхід моделі подається тривимірний масив (тензор), глибина якого - це кількість паралельної інформації про зображення. При роботі з кольоровими зображеннями глибина вхідного зображення дорівнює трьом.

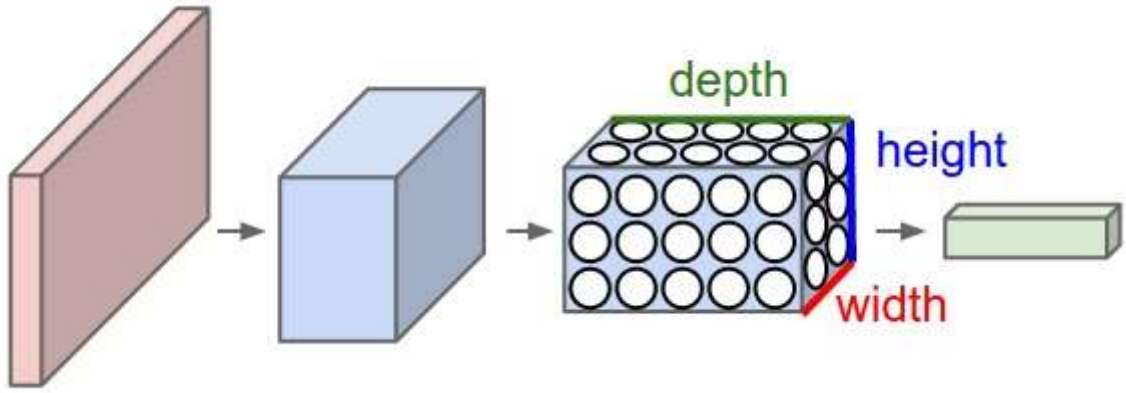

Рисунок 4 Глибина, ширина і висота нейронів у CNN

1.2.2.1 Згортання

Як було сказано вище, рівні згорткової нейронної мережі мають глибину, висоту і ширину (Рисунок 4). Висота і ширина відповідають висоті і ширині вхідного зображення. Операції між рівнями перетворюють тривимірні групи нейронів у тривимірні групи нейронів (часто змінюючи розмір).

Однією з таких операцій є згортання. В операції згортання використовується фільтр з параметрами, що навчаються.

Фільтр (також називають ядром) має невеликі розміри за осями ширини і висоти але має глибину таку ж як і тензор, що обробляється. Тобто, якщо на першому рівні мережі ми маємо тензор розміру [32x32x3], то вимогою до фільтра буде глибина 3.

За осями ширини і висоти фільтр зазвичай має однакові розміри. Розмір фільтру називається розміром чутливого поля (receptive field size).

Інтуїтивно розмір фільтр по осях ширини і висоти – це параметр, який визначає, як близько нейрони мають знаходитись один до одного, щоб бути з'єднані з нейроном операцією згортання на наступному рівні.

Фільтр почергово застосовують до різних ділянок зображення рухаючись у площині осей ширини і висоти з певним кроком (stride).
Перша позиція фільтра знаходиться так, щоб всі коефіцієнти фільтра співвідносились до верхнього лівого кутка зображення.

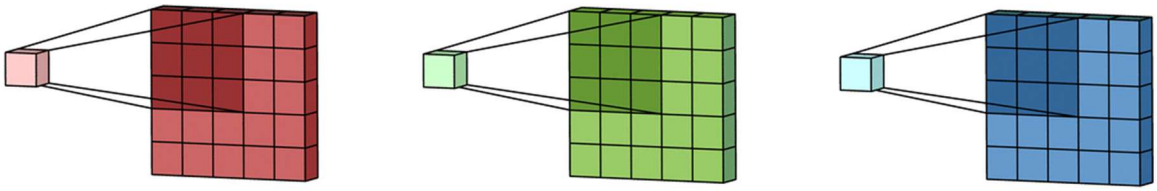

Рисунок 5 Візуалізація роботи згорткової операції на рівень з розміром 5x5x1фільтром розміром 3x3x1 з stride=1, padding=0. На ілюстрації зображено роботу трьох різних фільтрів

Частина нейронів рівня, яка має однакову координату за віссю глибини має назву depth slide [2].

Для кожного нейрона, його значення розраховується як сума добутків значень нейронів у зоні згортання попереднього рівня на коефіцієнт у фільтрі. Ця сума включає в себе добутки для всіх depth slide

До кожного рівня застосовується декілька (d') фільтрів. Відповідно на наступному рівні глибина дорівнюватиме d'.

Без використання спеціальних відступів розмір кожного наступного рівня буде зменшуватись по осях ширини і висоти. Для прибирання цього ефекту необхідно умовно додати заповнені нулем нейрони до рівня, для якого проводиться операція згортання. Кількість стрічок і стовпчиків, заповнених нулями, що додаються до рівня, називається padding.

$$(W - F + 2P)/S + 1$$

Формула 1.2.2.1 для розрахунку розміру рівня, що утворюється внаслідок згорткової операції.

Де,

W – ширина рівня,

F – ширина фільтру

P – розмір відступу

S – зміщення фільтра за один крок

Крім цього іноді додають пропуски у фільтр, а гіперпараметр, який за це відповідає, називається dilation.

Якщо припустити, що фільтр, який був корисним у одній частині зображення, може бути не менш корисним в іншій частині зображення, то можна значно зменшити кількість параметрів для вивчення. З інтуїтивного погляду це має сенс, оскільки фільтри дозволяють виокремлювати примітивні форми, що зустрічаються усюди в зображенні.

Цей підхід називається поширенням параметрів (parameter sharing) і полягає у наступному: замість того, щоб визначати коефіцієнт ваги зв'язку для кожного зв'язку між нейронами результату згортання і групою нейронів на попередньому рівні, вважати, що всі зв'язки між групами нейронів на попередньому рівні і нейронами на одному depth slide мають рівну вагу.
Це зменшує кількість коефіцієнтів для вивчення у WxH раз, де W – ширина, H – висота рівня нейронів, що є результатом згортання.
Цей підхід вимагає невеликої зміни у алгоритмі навчання, а саме: для кожного нейрона розрахується градієнт, і зміна коефіцієнтів відбувається за сумарним градієнтом.

1.2.2.2 ReLU

ReLU – це одна з частовживаних активаційних функцій, вона має формулу [Формула 1.2.2.2.1]

$$ReLU(x) = \begin{cases} x, \text{якщо } x > 0 \\ 0, \text{інакше} \end{cases}$$

Формула 1.2.2.2.1 ReLU.

Її особливістю є те, що вона може не передавати градієнт взагалі (при від'ємих значеннях аргументу), що дозволяє наблизитись до біологічної аналогії – нейронів, що не передають сигнал.

Крім цього функція є простою у підрахунку. Функція краще поширює градієнт, тобто відсутня класична проблема зникаючого градієнта (коли похідна активаційної функції наближається до нуля).

У функції є і проблеми, одна з них – «мертві нейрони». Зв'язки між нейронами не навчаються з ReLU, якщо значення вхідного аргументу менше нуля. В процесі навчання можна опинитись в ситуації, де у зв'язків є велика негативна вага, що зробить неможливим подальше навчання для цього нейрона.
Для боротьби з цією проблемою пропонується використовувати Leaky ReLU [Формула 1.2.2.2.2]

$$LeakyReLU(x) = \begin{cases} x, \text{якщо } x > 0 \\ 0.01, \text{інакше} \end{cases}$$

Формула 1.2.2.2.2 Leaky ReLU.

У цьому варіанті функції, градієнт буде повільно поширюватись і на зв’язки з нодами, що не були активовані.

1.2.2.3 Максимізаційне агрегування (Max Pooling)

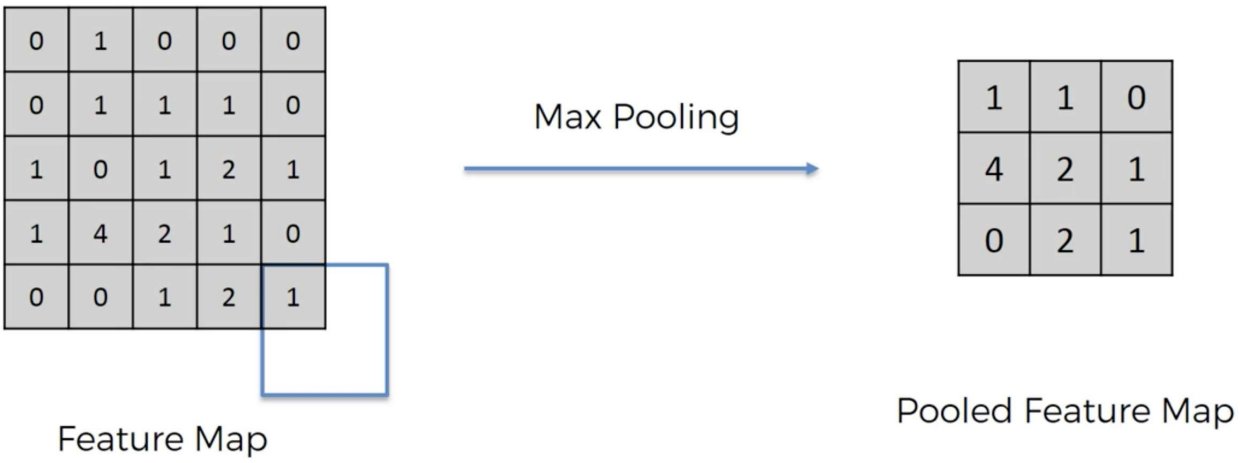

Рисунок 6 Демонстрація операції максимізаційного агрегування

Операція максимізаційного агрегування використовується для зменшення розмірності рівнів моделі зі спробою зберегти цінну інформацію про зображення.

Припускається, що попередні рівні моделі змогли екстрагувати з зображення ознаки, що нам цікаві і передають їх вперед високими значеннями даних. Для виконання агрегування необхідно вибрати розміри зони вибірки (зазвичай, це квадрат 2x2) і крок зміщення зони вибірки (зазвичай це 2).

Зона вибірки по черзі накладається на всі частини зображення рухаючись з вибраним кроком. На кожному етапі з множини значень нейронів, що потрапили у зону вибірки, обирається найбільше і записується у відповідну комірку наступного рівня.

Ця операція в своєму найпоширенішому варіанті (розмір зони виявлення 2x2 і крок зсуву 2) зменшує розмірність рівня мережі по осі ширини і висоти.

1.2.2.4 Метрика IoU

IoU (Intersection over Union) – це метрика, що розраховується як відношення площі перетину двох обмежуючих прямокутників до площі об’єднання обмежуючих прямокутників.

Значення метрики дозволяє відповісти на питання: як добре прогнозоване місцезнаходження об’єкту збігається з реальним.

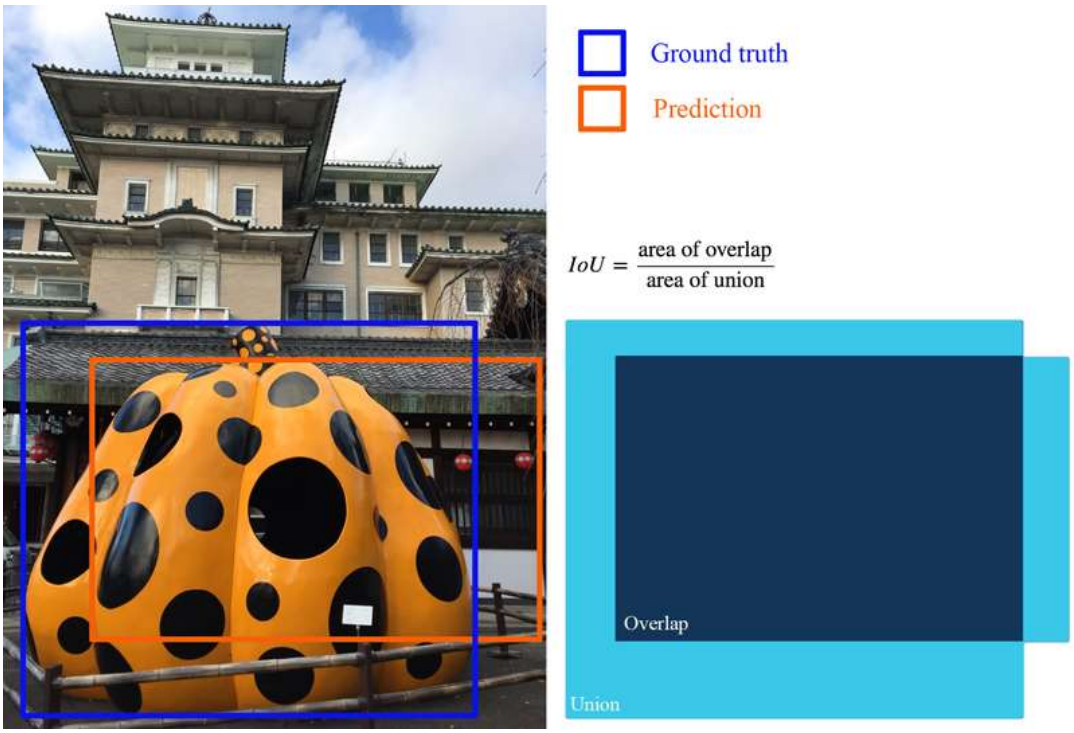

Рисунок 7 Ілюстрація метрики IoU

1.2.3 Структура CNN

Операція згортання є ключовою для CNN і не випадково є невід’ємною складовою будь якої сучасної моделі для виявлення об’єктів.

Операція згортки дала можливість знаходити примітивні патерни на зображенні. Цими патернами є лінії, кути, краплі. Як показали досліди з системою зору котів, це саме роблять і біологічні системи розпізнавання образів.

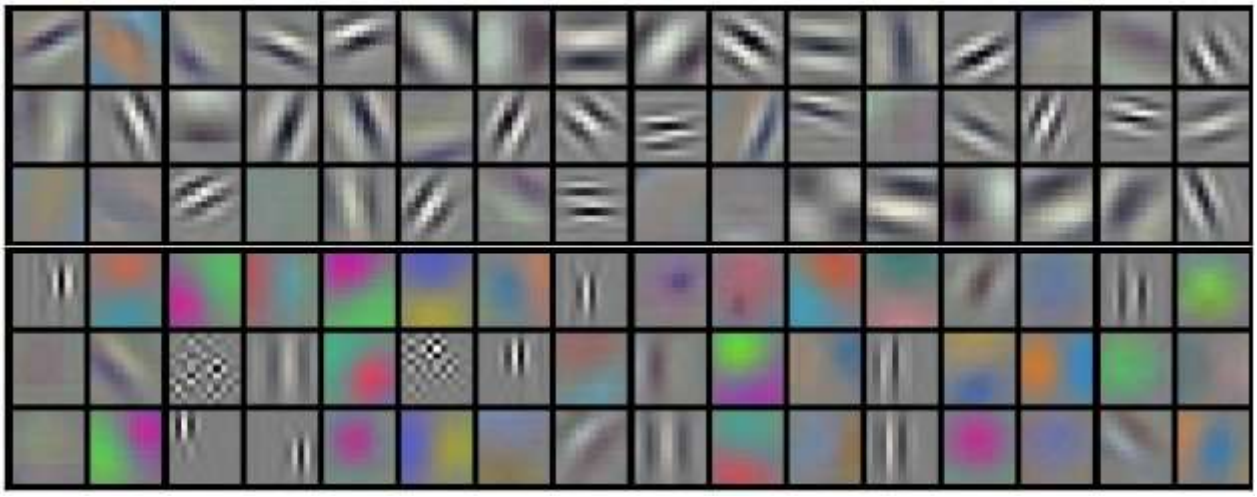

Рисунок 8 Приклад фільтрів 11x11x3 натренованих на перших рівнях моделі

Використання підходу поширення параметрів дозволяє згортковій моделі бути еквіваріантною по відношенню до трансляцій зображення (тобто до переміщення об’єкту по зображенню без зміни розміру і поворотів).

Еквіваріантність до інших видів змін зображення бажана і частково досягається методами модифікації даних для навчання (data augmentation).

Перші рівні CNN - це рівні згортки, вони зменшують розмірність по ширині, висоті але збільшують розмірність по глибині. Їх роль - виявити примітиви на зображенні.

Поширеним є використання ReLU рівнів після рівнів згортки. Часто ближче до останніх рівнів використовують рівні з max pooling. Останніми рівнями часто є FC рівні.

CNN є успішною з точки зору виділення ознак і часто використовується як компонент у складніших архітектурах.

1.3 Cross entropy loss

Cross entropy loss – це функція втрат, що використовується для задач класифікації. В її основі лежить функція перехресної ентропії [Формула 1.3.1].

$$-\sum_{x \in X} p(x)\, log\big(q(x)\big)$$

Формула 1.3.1 Перехресна ентропія між розподілами p і q над простором подій X. Де, x – подія.

Для застосування формули як функції втрат підставляємо замість X множину класів, замість p(x) – 1 якщо x – правильний клас і 0 в інакшому випадку. Замість q(x) необхідно підставити впевненість моделі у тому, що x – правильний клас.

Слід зазначити, що до множини класів завжди належить спеціальний клас, що означає відсутність класу.

2 Будова існуючих архітектур одноетапних детекторів об’єктів

Моделі для виявлення об’єктів (детектори) отримують на вхід зображення, а на виході мають інформацію про те, які об’єкти зображені на фото і де вони знаходяться. Для кожного знайденого об’єкта на зображенні у вихідному векторі міститься 4 числа, що кодують положення і розмір обмежуючого прямокутника і n чисел, що кодують впевненість моделі у тому, що об’єкт належить до кожного з класів.

За особливістю дизайну моделей, було розділено всі моделі детекторів об’єктів на одноетапні і двоетапні. Головна їх відмінність полягає у тому, що двоетапні моделі спочатку знаходять пропозиції обмежуючих прямокутників, а потім класифікують і модифікують обрані пропозиції. Одноетапні моделі ж не мають етапів пропонування і оцінювання пропозицій.

2.2 Single Shot MultiBox Detector

Single Shot MultiBox Detector (SSD) – архітектура одноетапної мережі виявлення об’єктів. Була запропонована у 2016 році [3].

2.2.3 Основні ідеї дизайну моделі

SSD прагне використовувати ознакові мапи (feature maps) різних розмірів для предикторів зображень різного розміру.

Крім цього SSD розділяє предиктори зображень з різним співвідношенням сторін.

SSD використовує як основи для обмежуючих прямокутників, що визначатимуть знайдені об’єкти, набір зразкових прямокутників (default bounding boxes).

2.2.4 Особливості реалізації

2.2.4.1 Базова мережа

В основі моделі знаходиться мережа класифікації зі «зрізаними» рівнями з кінця. Таким чином, використовується перевірений спосіб екстрактувати примітивні ознаки зображення. В статті, що пропонує архітектуру SSD, використовується мережа VGG-16, хоча і зазначається, що можливі успішні заміни базової мережі на іншу.

2.2.4.2 Пірамідальна структура рівнів

З кожним рівнем внаслідок додавання згорткових операцій зменшується розмір ознакових мап по осях ширини і висоти. Таким чином виконується ідея створення ознакових мап з різними рівнями деталізації.

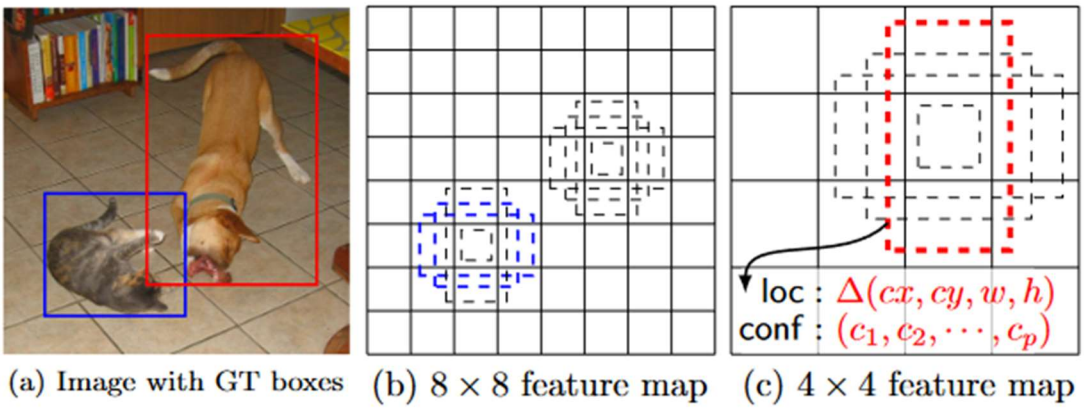

Рисунок 9 Демонстрація ознакових мап з різним рівнем деталізації

Зі зменшенням розмірності ознакової мапи збільшується площа «клітинок», що дозволятиме створювати зразкові прямокутники різних розмірів. Створюючи набори зразкових прямокутників з різним співвідношенням сторін на основі

ознакових мап різного розміру, ми досягаємо непоганого перекриття для всіх можливих обмежуючих прямокутників.

2.2.4.3 Зразкові прямокутники

Для кожної клітини ознакових мап різних розмірів створюється по декілька (k) зразкових прямокутників. До ознакової мапи розміру $m \times n$ з p фільтрами додаються рівні згортки ядрами $3 \times 3 \times p$. Ми очікуємо з кожного зразкового прямокутника отримати вектор довжиною c+4, де c – кількість класів, що мережа може розпізнати. З цього вектора c значень – це ймовірності того, що всередині зразкового прямокутника знаходиться об’єкт певного класу. Інші 4 значення - це коригування зразкового прямокутника для кращого вказування на об’єкт.

Врахувавши вище наведені числа, отримуємо необхідність у $(c + 4)k$ фільтрах і наявність $(c + 4)kmn$ вихідних нейронів (для кожної ознакової мапи).

2.2.4.4 Навчання

2.2.4.4.1 Вибір обмежуючих прямокутників, що вважаються правильно передбаченими

Для того, щоб навчати цю модель, необхідно спочатку обрати зразкові прямокутники, що є близькими до об’єктів на зображенні. Для порівняння близькості використовується метрика IoU. Датасети містять анотовані дані. Існує декілька видів анотацій, але для нас важливий варіант, де для всіх об’єктів на фото є обмежуючі прямокутники і інформація про приналежність до класу. Обмежуючий прямокутник, отриманий з анотації, називають ground bounding box.

Для кожного з анотованих обмежуючих прямокутників обирається зразковий прямокутник найближчий до нього за метрикою IoU. Після цього обираються всі зразкові прямокутники, що містять значення близькості до анотованих обмежуючих прямокутників IoU і вище і відносяться до відповідних анотованих обмежуючих прямокутників.

4.1.2.4.2 Функції втрат

Для навчання використовується функція втрат, що комбінує у собі втрати від локалізації обмежуючих прямокутників і похибку у впевненості про відношення об’єктів до класів. [Формула 4.1.2.4.1]

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

Формула 4.1.2.4.2.1 Функція втрат для SSD. [3]

Де,

x – тривимірна матриця, що вказує на те, чи зв’язані і-тий зразковий обмежуючий прямокутник і j-тий анотований обмежуючий прямокутник класу p ($x_{ij}^{p} = 1{,}0$).

c – матриця з впевненістю моделі у тому, що у і-тому зразковому обмежуючому прямокутнику знаходиться об’єкт класу p (c_i^p).

l – набір параметрів передбаченого обмежуючого прямокутника

g – набір параметрів анотованого обмежуючого прямокутника

N – кількість обмежуючих прямокутників, що були визнані правильно передбаченими

a – вага функції втрат локалізації (гіперпараметр).

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \sum_{m \in \{cx, cy, w, h\}} x_{ij}^{k} \text{smooth}_{\text{L1}}(l_i^m - \hat{g}_j^m)$$

$$\hat{g}_j^{cx} = (g_j^{cx} - d_i^{cx})/d_i^w \qquad \hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy})/d_i^h$$

$$\hat{g}_j^w = \log\left(\frac{g_j^w}{d_i^w}\right) \qquad \hat{g}_j^h = \log\left(\frac{g_j^h}{d_i^h}\right)$$

Формула 4.1.2.4.2.2 Функція втрат локалізації використаний у SSD [3].

Де,

cx, cy – зміщення по 2 координатам центру обмежуючого прямокутника від зразкового обмежуючого прямокутника.

w, h – зміщення ширини, висоти обмежуючого прямокутника від зразкового обмежуючого прямокутника.

d – набір параметрів зразкового обмежуючого прямокутника

cx, cy, w, h – значення передбачені мережею.

$$L_{conf}(x,c) = -\sum_{i\in Pos}^{N} x_{ij}^{p} log(\hat{c}_i^p) - \sum_{i\in Neg} log(\hat{c}_i^0) \quad \text{where} \quad \hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)}$$

Формула 4.1.2.4.2.3 Функція втрат впевненості [3]

Ця функція [Формула 4.1.2.4.2.3] називається softmax loss. Її суть полягає у перетворенні значень c (які є пропорційними впевненості мережі у належності об’єкта до певного класу) у діапазон [0; 1] так, щоб сума c_i^p для кожного конкретного p і всіх i дорівнювала 1.

2.2.4.5 Non-maximum Suppression (NMS)

У багатьох моделей виявлення об’єктів в результаті роботи моделі є багато кандидатів на правильну відповідь, що є схожими між собою. Для коректної роботи програми, потрібно прибрати зайві пропозиції.

Алгоритм NMS працює наступним чином [4]:

1. Обрати пропозицію з найбільшим значенням впевненості моделі, і перемістити її у список відповіді.
2. Порівняти пропозицію, обрану на минулому кроці, з усіма іншими пропозиціями, що не були ще обрані (використовуючи IoU). Всі пропозиції, що є ближчими за певний поріг, видаляються і в подальшому не розглядаються.

3. Повторювати пункти 1, 2 допоки всі пропозиції не будуть переміщені у список відповіді або не будуть відкинутими.

SSD використовує NMS, попередньо відфільтрувавши всі пропозиції з значенням впевненості менше 0.01 [3]. Поріг для NMS в реалізації SSD дорівнює 0.45. Крім цього існує обмеження на кількість пропозицій на виході – 200.

2.3 RetinaNet

2.3.3 Основні ідеї дизайну моделі

Розробники RetinaNet запропонували нову функцію втрат замість функції перехресної ентропії і назвали її Focal Loss [5].

Причиною цього стала помічена розробниками проблема, а саме: через значний дисбаланс частин зображення, що є фоновими до частин зображення з об’єктами, тобто цікавими для нас. Відповідно, при підрахунку функції втрат фонові частини зображення навчаються краще, а нефоновим навчатись дуже складно. Ідеєю Focal Loss є пригнічення ролі добре класифікованих об’єктів при розповсюдженні помилки. Для простоти, розглянемо бінарну класифікацію.

$$CE(q,p) = \begin{cases} -log(q), \text{якщо } p = 1 \\ -log(1 - q), \text{в інших випадках} \end{cases}$$

Формула 2.3.3.1 Перехресна ентропія для бінарної класифікації

Де, q – прогнозована моделлю ймовірність того, що об’єкт є «позитивного» класу, а $p \in \{\pm 1\}$, $p = 1$, коли об’єкт є «позитивного» класу.

Для зручності написання, використаємо скорочення [Формула 2.3.3.2].
Тепер перехресну ентропію можна переписати як [Формула 2.3.3.3]

$$p_t = \begin{cases} q, & \text{якщо } p = 1 \\ 1 - q, & \text{в інших випадках} \end{cases}$$

Формула 2.3.3.2 Скорочення для Focal Loss.

$$CE(q,p) = -log(p_t)$$

Формула 2.3.3.3 Скорочений варіант запису функції перехресної ентропії для бінарної класифікації

Функція Focal Loss [Формула 2.3.3.4] містить гіперпараметр γ, що регулює як сильно необхідно пригнічувати легкі для класифікації частини зображення.

$$FL(p_t) = -(1 - p_t)^{\gamma} log(p_t)$$

Формула 2.3.3.4 Focal Loss

Тож, функція Focal Loss значно зменшує вплив таких прикладів для яких впевненість моделі є високою.

Це не перша спроба боротися з дисбалансом класів вхідних частин зображення. Так, у SSD [3] кількість негативних прикладів штучно обмежувалася в кількості (не більше 3 негативних приклади на 1 негативний).

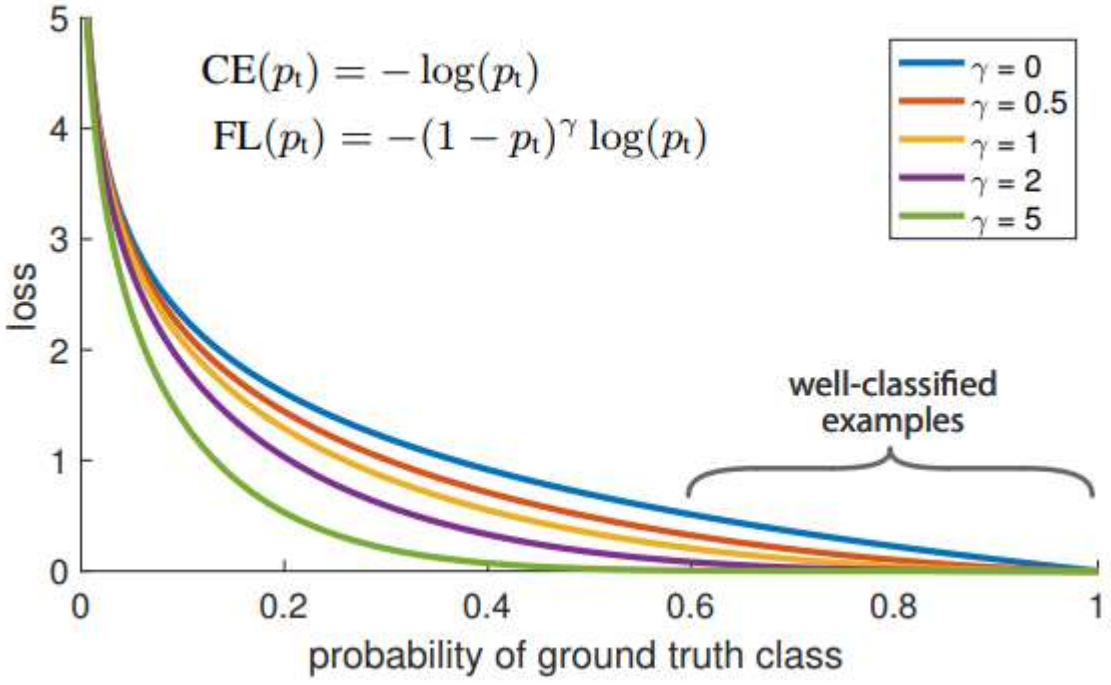

Рисунок 10 Порівняння CE і FL з різними значеннями параметра γ

2.3.4 Особливості реалізації

2.3.4.1 Високорівневий погляд на архітектуру

Як і SSD, RetinaNet використовує існуючу мережу як основу. В даному випадку це – ResNet. На відміну від SSD, RetinaNet додає до ResNet надбудову у вигляді пірамідальної структури з FPN.

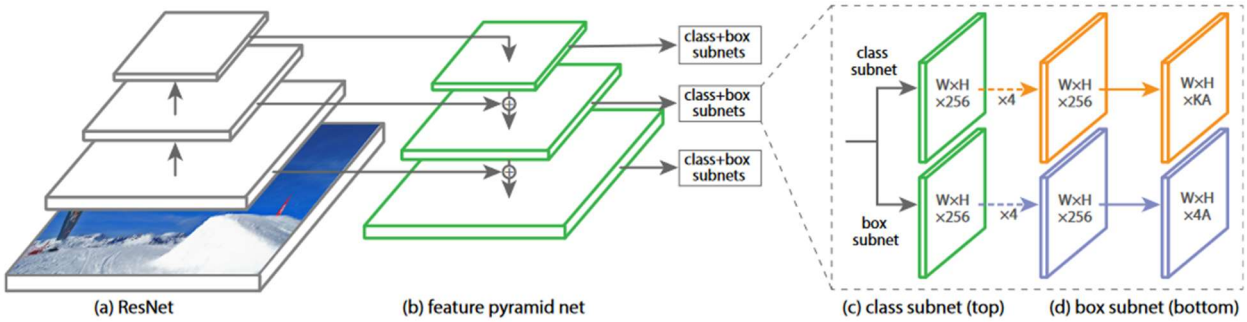

Рисунок 11 Високорівнева структура RetinaNet

Після пірамідальної структури додаються дві підмережі: одна класифікує зображення у якірних прямокутниках (anchor boxes) (аналог зразкових обмежуючих прямокутників у SSD), а інша – підраховує корекцію для якірних прямокутників.

2.3.4.2 Пірамідальна структура рівнів

Причина створення пірамідальних структур з мап ознак – це наслідок прагнення виявляти об’єкти різних розмірів. Подібна ідея використовується і в SSD, де зразкові прямокутники розміщувались на рівнях різного розміру. В той же час не були використані мапи ознак на найбільших рівнях, що спричинило б значне сповільнення роботи моделі.

Пірамідальна структура присутня у RetinaNet запозичена з FPN (Feature Pyramid Network). Взаємодія рівнів мережі зображена на [Рисунок 11]. До вже звичної структури зі зниженням розмірності рівнів (шлях знизу вверх) додаються рівні, що збільшують розмірність (шлях зверху вниз).

Якірні прямокутники знаходяться на кожному пірамідальному рівні на шляху зверху вниз.

Через велику кількість операцій, що змінювали розмірність рівнів, інформація про місцезнаходження об’єктів на зображенні може втратитись. Саме тому було додано зв’язки з рівнями на шляху знизу вверх. Ці зв’язки є подібні до зв’язків у ResNet.

Кожен рівень на шляху знизу вверх стає вдвічі меншим і навпаки – на шляху зверху вниз розмір збільшується вдвічі.

2.4 YOLOv1

YOLO (You Only Look Once) – модель, що відома своєю швидкістю роботи. Модель працює зі швидкістю 45 зображень за секунду, що дозволяє використовувати її для виявлення об’єктів у реальному часі. Вона була запропонована у 2016 році [6] і стала початком серії не менш успішних моделей таких як YOLOv2 і YOLOv3.

2.4.1 Основні ідеї дизайну моделі

YOLOv1 розділяє вхідне зображення на сітку розміру S×S. Для кожної клітини розраховуються C значень, що репрезентують ймовірності того, що якщо у даній клітині знаходиться об’єкт, то він належить до певного класу.

Крім цього кожна клітина містить в собі центри декількох (B) обмежуючих прямокутників. Для кожного прямокутника розраховуються 5 значень: x, y, w, h, conf.

x, y - репрезентують координати центру обмежуючого прямокутника відносно клітини.

w, h – репрезентують ширину і висоту обмежуючого прямокутника відносно всього зображення.
conf – це міра впевненості моделі у тому, що у обмежуючому прямокутнику знаходиться модель, помножена на міру близькості обмежуючого прямокутника до анотованого прямокутника (IoU) [6].

2.4.2 Особливості реалізації

YOLOv1 складається з згорткових рівнів, пронизаних рівнями максимізаційного агрегування. Останніми двома рівнями є повністю з'єднані рівні.

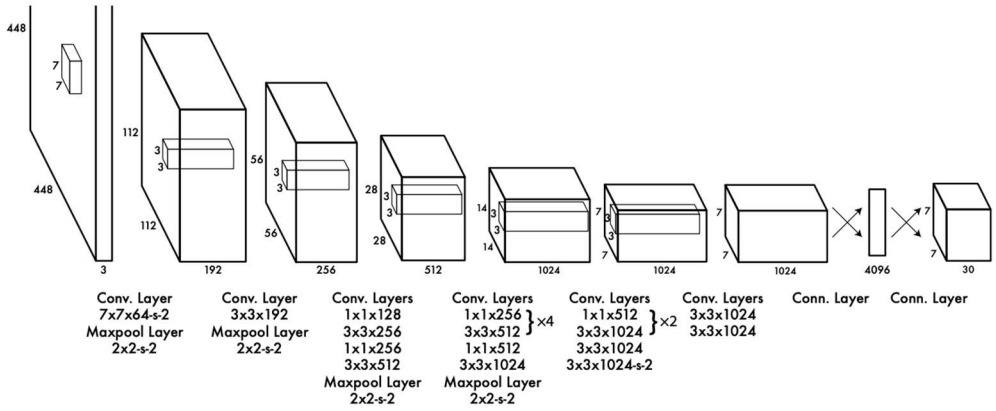

Figure 3: The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224 input image) and then double the resolution for detection.

Рисунок 12 Структура моделі YOLOv1

Згорткові рівні використовують Leaky ReLU в якості активаційної функції.

Останній рівень моделі використовує ReLU замість Leaky ReLU.

Розробники YOLOv1 також вказують на проблему дисбалансу класів, що створює проблему з поширенням градієнту. Ними пропонується зменшити вплив на градієнт нейронів, що відповідають за обмежуючі прямокутники, які не містять об'єктів у 10 разів.

3 Існуючі метрики для оцінки ефективності моделі

Для оцінки точності виявлення об'єктів використовуються такі метрики:

AP, AP50, AP75, AP@[.5: .05: .95], mAP.

Перед тим, як перейти до опису значення кожної з цих метрик, необхідно зазначити, що всі виміри відбуваються на одному з стандартних бенчмарків, що створюються на основі найбільших датасетів. У статтях з описом моделей, розглянутих вище, найчастіше використовується бенчмарк на основі датасету

COCO (Common Objects in Context). Він є одним з найпоширеніших датасетів, для виявлення об’єктів.

Тож, продовжимо про метрики.
Спочатку треба згадати, що значить точність (precision) і повнота (recall).

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

TP = True positive
TN = True negative
FP = False positive
FN = False negative

Рисунок 13 Формули точності і повноти

Точність - це відношення кількості знайдених об’єктів до кількості спроб вгадування.

Повнота – це відношення кількості знайдених об’єктів до загальної кількості об’єктів на зображенні.

Зі збільшенням кількості спроб вгадування зростає повнота і скачкоподібно зменшується точність. Практично неможливо досягти максимальних значень точності і повноти одночасно. Тому при реальному використанні моделі потрібно було б вказати параметр, що вирішував, де піти на компроміс. При великих значеннях повноти неодмінно з’являтимуться помилково позитивні вгадування. Цей поріг визначає наскільки впевненою модель повинна бути, щоб вказати здогадку як правильну. Змінюючи цей параметр і вимірюючи значення точності і повноти, можна отримати графік залежності точності від повноти (Precision Recall Curve).

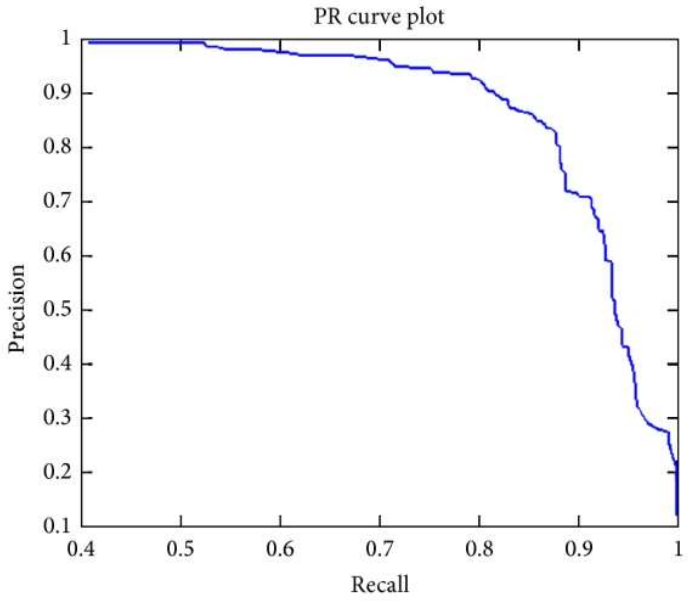

Рисунок 14 Precision Recall Curve

Враховуючи принцип побудови функції, ми розраховуємо лише її апроксимацію (Кожна точка, що описує функцію – це необхідність перераховувати значення точності і повноти). Таким чином, якщо ця функція на певних ділянках зростає, то ми заміняємо ці ділянки на максимальне значення функції на локальній ділянці функції (під ділянкою мається на увазі частина функції між двома розрахованими сусідніми точками).

Площу фігури, що є обмеженням кривої і осей, називають значенням AP (Average Precision). Іншими словами, AP – це інтеграл функції точності по повноті.

Під час оцінки моделей виявлення об’єктів з’являється питання: наскільки близькою має бути здогадка до правильної відповіді, щоб вважати здогадку правильною. Для оцінки близькості здогадки і правильної відповіді використовується метрика IoU.

AP50, AP75 – це метрики AP для чіткого порогу (50%, 75%).

Якщо не вказувати чіткий поріг для IoU, а розглянути низку варіантів, отримаємо низку значень AP.

AP@[.5: .05: .95] – це середнє значення метрик AP для ряду порогових значень IoU від 50% до 95% з кроком 5%. [7]

Метрика mAP і ій подібні – це аналоги метрики AP і їм подібних, але усереднені для всіх класів. Для бенчмарку COCO це неважливо, бо він не рахує метрики для кожного класу окремо. Конкретно для COCO, AP значить AP@[.5: .05: .95]. Ця метрика є основною в бенчмарку COCO.

Також у COCO існують окремі метрики для об'єктів різного розміру, що дозволяє краще зрозуміти, як модель розпізнає об'єкти різних розмірів.

4 Порівняльна характеристика алгоритмів представлених вище

У кожної з розглянутих моделей існує низка модифікацій, що покращують якість виявлення об'єктів або швидкість роботи алгоритмів. Іноді можна покращити і якість, і швидкість. Нижче приведені метрики розглянутих моделей на COCO test-dev датасеті. Метрики отримані з оригінальних статей. Якщо у статті було представлено декілька варіацій з різними метриками, були використані варіації з найбільшим значенням AP.

Найбільша ефективність моделей (AP)			
Назва моделі	AP	AP50	AP75
SSD512	28.8	48.5	30.3
YOLOv2	21.6	44	19.2
RetinaNet	40.8	61.1	44.1

Порівняння метрик є ускладненим, бо з часом змінюються популярні бенчмарки. Так, оригінальна стаття з YOLOv1 не містить інформації про ефективність на COCO датасеті. В той же час стаття з пропозицією RetinaNet не містить інформації про ефективність роботи на датасетах Pascal VOC 2007 чи Pascal VOC 2012. Таким чином без проведення власного тестування неможливо порівняти всі моделі одночасно.

З порівнянням швидкості моделі теж є свої проблеми. Так стаття з пропозицією

SSD не містить інформацію про швидкість роботи на датасеті COCO, але містить таблицю порівняння ефективності і швидкості роботи на датасеті VOC2007. Ця таблиця дозволяє порівняти швидкість і ефективність роботи YOLOv1 (на таблиці записаний як YOLO(VGG16)) і SSD512.

Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	$\sim 1000 \times 600$
Fast YOLO	52.7	155	1	98	448×448
YOLO (VGG16)	66.4	21	1	98	448×448
SSD300	74.3	46	1	8732	300×300
SSD512	76.8	19	1	24564	512×512
SSD300	74.3	59	8	8732	300×300
SSD512	76.8	22	8	24564	512×512

Table 7: **Results on Pascal VOC2007 `test`.** SSD300 is the only real-time detection method that can achieve above 70% mAP. By using a larger input image, SSD512 outperforms all methods on accuracy while maintaining a close to real-time speed.

За результатами порівняння швидкостей, можна прийти до висновку, що хоча YOLOv1 і був сенсацією завдяки своїй швидкості, з часом інші моделі наздогнали його характеристики. В той же час YOLO розвинувся у серію всі більш ефективних алгоритмів YOLOv3, YOLOv4.

ВИСНОВОК

Одноетапні моделі для виявлення об’єктів відрізняються від двоетапних тим, що мають виділяти перспективні ділянки зображення для перевірки на наявність там об’єкта певного класу у один прохід – без етапу пропозицій.

Використання цього підходу має свої позитивні і негативні сторони. Позитивним є те, що можна отримати дійсно швидку модель, що буде достатньо точною.

Одна з поширених проблем це те, що необхідно розглядати на етапі тренування багато частин зображення, що не містять цінної інформації (фон). Часто на них витрачається забагато «уваги градієнта». Кожна з розглянутих в роботі моделей запропонувала свій варіант подолання цієї проблеми. У SSD і YOLOv1 ці підходи є доволі схожими. У той же час мені здається перспективнішим використання функції втрат від RetinaNet (Focal Loss). Вона дозволяє налаштувати, наскільки сильно фонові приклади слід пригнічувати при навчанні.

Як я і писав вище, практично в усіх популярних сьогодні моделях використовуються згорткові операції для виділення ознак з зображення. Ба більше, більшість «складніших» моделей використовують готові згорткові моделі для класифікації зображень як основу для своїх структур.
Цим сучасні моделі зобов’язані унікальним властивостям згорткової операції: виділяти примітивні ознаки зображення, бути еквіваріантними до операції трансляції, використовувати лише локальні дані.

Еквіваріантість моделей до змін у зображенні є надзвичайно важливою характеристикою, яку хотілося б мати не тільки для трансляції, а і для операцій зміни розміру, повороту, віддзеркалення. Наразі, ця проблема частково вирішується модифікацією даних перед тренуванням. Над даними проводять операції, вказані вище перед навчанням, щоб модель була стабільною.

Результати порівняння переконали мене в тому, що хоча і можна вишукувати найкращу модель для всіх завдань з виявлення об’єктів, намагаючись вибрати одночасно швидку і точну модель, але навряд чи у цьому досягнеш успіху. Тож при виборі моделі слід звернути увагу на вимоги по швидкості і обрати найточнішу модель, що їм задовольняють.

Список використаної літератури

1. Backpropagation. Brilliant.org. URL: https://brilliant.org/wiki/backpropagation/ (дата звернення: 14.05.2022).
2. CS231n Convolutional Neural Networks for Visual Recognition. cs231n.github.io. URL: https://cs231n.github.io/convolutional-networks/ (дата звернення: 15.05.2022).
3. Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg. SSD: Single Shot MultiBox Detector. *https://arxiv.org/*. URL: https://arxiv.org/abs/1512.02325 (дата звернення: 19.05.2022).
4. Non-maximum suppression (NMS). towardsdatascience.com. URL: https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c.
5. Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, Piotr Dollár. Focal Loss for Dense Object Detection. *https://arxiv.org*. URL: https://arxiv.org/abs/1708.02002 (дата звернення: 19.05.2022).
6. Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. *https://arxiv.org*. URL: https://arxiv.org/pdf/1506.02640.pdf (дата звернення: 19.05.2022).
7. The confusing metrics of AP and map for object detection / instance segmentation. *yanfengliux.medium.com*. URL: https://yanfengliux.medium.com/the-confusing-metrics-of-ap-and-map-for-object-detection-3113ba0386ef (дата звернення: 18.05.2022).