

Харківський національний університет імені В.Н. Каразіна

Факультет математики і інформатики

Кафедра прикладної математики

Кваліфікаційна робота

магістра

на тему “Розробка автоматизованої системи для ідентифікації та аналізу характеристик дахів з використанням нейронних мереж для сегментації зображень”

«Development of an automated system for the identification and analysis of roof characteristics using neural networks for image segmentation»

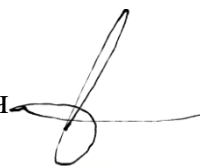
Виконав: студент 2-го курсу магістратури

за ОПІ Магістр

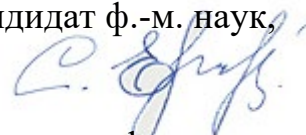
спеціальності 113 «Прикладна
математика»,

освітня програма «Прикладна
математика»

Костенко Віталій Артемович



Керівник: доцент, кандидат ф.-м. наук,
Степанова К. В.



Рецензент: доцент, кандидат ф.-м. наук,
Базілевич К. О.

Харків 2023

Abstract

The thesis focuses on the development and analysis of neural networks for application in image segmentation tasks, with a special emphasis on the analysis of images of building roofs. The paper provides a comprehensive analysis of various aspects of machine learning and computer vision, focusing on the accuracy and efficiency of various segmentation algorithms, including FastInst and Vision Transformer (ViT). The work includes a detailed review of modern deep learning algorithms and models, with an emphasis on their architecture, principles of operation, and application possibilities in the context of image segmentation tasks. The work covers the process of collecting and preparing data necessary for training and testing neural networks. Attention is also paid to the development of an algorithm for determining the angle of inclination of roof slopes. The application of neural networks to these tasks is demonstrated and their accuracy and reliability are analyzed. The work also considers the potential practical applications of the developed methods and algorithms, pointing out the possibilities of their application in various fields, including urban planning, real estate and insurance.

Анотація

Дипломна робота зосереджується на розвитку та аналізі нейронних мереж для застосування в задачах сегментації зображень, з особливим акцентом на аналізі зображень дахів будівель. У роботі проводиться комплексний аналіз різноманітних аспектів машинного навчання та комп'ютерного зору, орієнтуючись на проблематику точності та ефективності різних алгоритмів сегментації, включаючи FastInst та Vision Transformer (ViT). Робота включає детальний огляд сучасних алгоритмів і моделей глибокого навчання, з акцентом на їхній архітектурі, принципах роботи, та можливостях застосування в контексті задач сегментації зображень. Робота покриває процес збору та підготовки даних, необхідних для тренування та тестування нейронних мереж. Також увага приділяється розробці алгоритму для визначення кута нахилу скатів даху. Демонструється застосування нейронних мереж до цих завдань та аналізує їхню точність та надійність. Робота також розглядає потенційні практичні застосування розроблених методів і алгоритмів, вказуючи на можливості їх застосування в різних галузях, включаючи міське планування, нерухомість та страхування.

ЗМІСТ

Abstract	2
Анотація.....	3
ЗМІСТ	3
Розділ 1.....	7
1.1 Основи глибокого навчання	7
1.2 Принципи роботи моделей інстанс сегментації.....	8
1.2.1 Огляд Mask R-CNN.....	10
1.2.2 Огляд PANet.....	12
1.2.3 Огляд Hybrid Task Cascade (HTC).....	14
1.2.4 Огляд SOLOv2: архітектура, основні компоненти, специфікації.....	17
1.2.5 Огляд BlendMask	20
1.2.6 Огляд FastInst	23
1.2.7 Огляд MaskFormer	34
Розділ 2.....	41
2.1 Принципи роботи моделей семантик сегментації.....	41
2.1.1 Огляд UNET	41
2.1.2 Огляд ViT	46
Розділ 3.....	51
3.1 Набір даних.....	51
3.2 Використання моделей інстанс сегментації для аналізу дахів.....	53
Список використаних джерел.....	59

ВСТУП

У сучасному світі комп'ютерного зору, інстанс сегментація відіграє ключову роль в аналізі та розумінні зображень. Ця технологія дозволяє детально виокремлювати об'єкти на зображенні, розпізнаючи їх семантичний зміст. Одним із можливих застосувань цієї технології є аналіз зображень дахів, що має значний потенціал у різноманітних областях, від урбаністики до розвитку інфраструктури.

Останнім часом у сфері комп'ютерного зору відбувається значний прогрес завдяки розвитку глибокого навчання. Зокрема, це стосується розробки ефективних методів інстанс-сегментації. Однією з найновіших моделей у цій області є FastInst [1], яка представляє собою просту, але ефективну рамку на основі запитів для реалізації швидкої інстанс-сегментації в реальному часі.

Особливість цієї моделі полягає в використанні трьох ключових технік: інстанс активаційно-керованих запитів [1], двохвильової стратегії оновлення у декодері Transformer [1, 2] та навчання, керованого масками істинності. Ці техніки дозволяють знизити навантаження на декодер Transformer, використовувати менш важкі піксельні декодери та досягти кращої продуктивності.

FastInst є прикладом того, як сучасні підходи на основі глибокого навчання можуть революціонізувати область інстанс-сегментації, забезпечуючи швидкість та точність, необхідні для реалізації складних задач у реальному часі.

Окрім FastInst, інша значна модель у цій області - Vision Transformer (ViT) [3], яка стає популярною для семантичної сегментації зображень. ViT відрізняється від традиційних підходів завдяки своїм механізмам уваги, які дозволяють моделі ефективно обробляти великі масиви даних і виділяти з них важливі особливості. Це особливо корисно для точної сегментації складних структур, таких як лінії скатів дахів, де потрібна висока точність для розрізнення різних елементів конструкції.

З точки зору продуктивності, ViTs виявилися кращими за традиційні моделі на основі конволюційних нейронних мереж (CNN) [4] у різних завданнях комп'ютерного зору. Ця перевага частково обумовлена їх здатністю обробляти великі набори даних та ідентифікувати важливі особливості в них, що особливо корисно для складних завдань, таких як семантична сегментація.

Додатково, розширення можливостей ViTs для семантичної сегментації включає використання вихідних вбудовувань [3], що відповідають зображенням патчів. Ці вбудовування обробляються для отримання класових міток за допомогою точкового лінійного декодера [3] або декодера трансформера масок. Такий підхід дозволяє більш точно сегментувати елементи зображення, сприяючи ефективності ViTs у таких застосуваннях, як аналіз ліній дахів.

Крім того, Vision Transformers досліджуються за їхній потенціал у самонавчанні в рамках семантичної сегментації. Цей підхід може додатково підвищити їх ефективність, дозволяючи моделям вчитися з великих нерозмічених наборів даних, що є поширеним сценарієм у реальних застосуваннях.

У цій роботі аналізуються можливості використання ViT для семантичної сегментації ліній даху та його внесок у визначення точних характеристик дахів, що включає розрахунок кутів нахилу та форм. В перших двох розділах оглянуті існуючі моделі для сегментації які розглядалися та використовувалися при написанні роботи. Використання цих моделей описано в останньому розділі. Адекватне розуміння цих характеристик має ключове значення для оцінки стану будівель, планування ремонтних робіт, а також для міського планування, де аналіз дахів може слугувати для визначення потреб у розробці міської інфраструктури.

Розділ 1

Моделі інстанс сегментації

1.1 Основи глибокого навчання

Глибоке навчання, яке стає все більш популярним у сучасних дослідженнях, є підмножиною машинного навчання і зосереджено на використанні нейронних мереж. В основі глибокого навчання лежать математичні моделі, які намагаються імітувати роботу людського мозку за допомогою великої кількості "нейронів", які взаємодіють між собою.

Основні концепції глибокого навчання:

• **Нейрони та Шари:** В основі глибоких нейронних мереж лежать штучні нейрони, що організовані у шари. Кожен нейрон приймає вхідні дані, обробляє їх за допомогою вагового коефіцієнта, і застосовує функцію активації до результату.

$$y = f(w \times x + b) \text{ [4]}$$

де y - вихід нейрона, w - ваговий коефіцієнт, x - вхідне значення, b - зсув, f - функція активації.

• **Функція активації:** Ця функція вносить необхідну нелінійність для навчання складних функцій. Найпопулярнішою є ReLU (rectified linear unit), але існують і інші, такі як сигмоїда та гіперболічний тангенс.

$$ReLU(x) = \max(0, x) \text{ [5]}$$

• **Функція втрат (Loss Function):** Це математичний спосіб вимірювання різниці між передбаченнями та реальними мітками. Для класифікації зазвичай використовується перехресна ентропія:

$$H(p, q) = -\sum_x p(x) \log q(x) \text{ [6]}$$

• **Оптимізатор:** Це алгоритм, який коригує ваги моделі, щоб мінімізувати функцію втрат. Найпопулярніший метод - градієнтний спуск:

$$w_{t+1} = w_t - \eta \nabla L \quad [4]$$

де η - крок навчання (learning rate), ∇L - градієнт функції втрат.

У порівнянні з традиційним машинним навчанням, глибоке навчання має здатність автоматично виявляти потрібні особливості з даних, в той час як традиційні методи часто потребують ручного визначення особливостей. Також глибокі моделі мають здатність моделювати набагато більш складні функціональні відношення завдяки своїм багатошаровим структурам.

1.2 Принципи роботи моделей інстанс сегментації

Інстанс сегментація є комбінацією детекції об'єктів та семантичної сегментації. Головна мета полягає в тому, щоб ідентифікувати окремі екземпляри об'єктів в зображенні та визначити, до якого класу вони належать, а також розділити їх від інших об'єктів.

• **Детекція Об'єктів:** Спочатку модель ідентифікує різні об'єкти на зображенні. Це може бути здійснено за допомогою механізмів пропозиції регіону, які визначають області, де ймовірно знаходяться об'єкти.

• **Призначення класу:** Для кожного з ідентифікованих регіонів модель призначає клас. Наприклад, "особа", "автомобіль" тощо.

• **Генерація маски:** Окрім ідентифікації регіону та призначення класу, модель також генерує маску для кожного об'єкта. Маска демонструє форму об'єкта в деталях, дозволяючи розділити його від інших об'єктів та фону.

Математичний контекст:

Припустимо, R — це множина регіонів, які були ідентифіковані на зображенні. Для кожного регіону $r \in R$, модель визначає клас c_r та маску M_r .

Визначення класу може бути розглянуто як класифікація: $c_r = \operatorname{argmax}(f_c(r; \theta))$, де f_c є функцією передбачення, а θ є параметрами моделі.

Маска для регіону r генерується за допомогою декодера, який відновлює форму об'єкта: $M_r = f_m(r; \theta)$.

Інстанс-сегментація бере свій початок від традиційної технології сегментації зображень, яка ділить зображення на окремі, але значущі підрегіони. До традиційних методів сегментації зображень належали методи, засновані на порогах, краях та кластеризації.

З поширенням комп'ютерного зору в складних сценах, таких як інтелектуальне водіння та моніторинг безпеки, традиційні технології сегментації зображень вже не могли задовольнити вимоги до сегментації об'єктів у таких сценах. Завдяки застосуванню глибокого навчання технологія сегментації зображень зробила значний прогрес. Такі алгоритми сегментації зображень, як FCN [7] та Mask R-CNN [8], засновані на глибокому навчанні, підняли технологію сегментації зображень на новий рівень, пропонуючи можливість застосування технології інстанс-сегментації в складних умовах.

Однією з найпопулярніших моделей для інстанс сегментації є Mask R-CNN [8], яка розширює Fastr R-CNN [9], додаючи додаткову гілку для прогнозування масок. Інші сучасні моделі, такі як FastInst [10] та SOLOv2 [11], пропонують новітні підходи до проблеми, спрощуючи архітектуру та покращуючи швидкість і точність.

В інстанс сегментації мета полягає не лише в ідентифікації, де розташовані об'єкти на зображенні, але й у визначенні окремих екземплярів цих об'єктів. Це відрізняє її від семантичної сегментації, де всі об'єкти одного класу об'єднуються в єдиний регіон.

Важливі моделі інстанс-сегментації:

• **Mask R-CNN (2017):** Розроблена He та ін., ця модель базується на Faster R-CNN і додає гілку маски для прогнозування маски об'єкта. Це класичний алгоритм інстанс-сегментації, який ґрунтується на виявленні об'єктів.

• **PA.Net (2019) [12]:** Розроблена Liu та ін., ця модель використовує метод посилення нижнього шляху з точними сигналами позиціонування на нижньому рівні для покращення всієї ієрархії функцій.

• **Hybrid Task Cascade (HTC, 2019) [13]:** Пропонується Chen та ін., ця модель впроваджує каскад в інстанс-сегментацію, переплітаючи виявлення та сегментацію для спільної багатоступінчастої обробки.

• **YOLOACT (2019) [14]:** Розроблена Boiya та ін., ця модель розкладає завдання інстанс-сегментації на два підзавдання: створення набору прототипних масок та прогнозування коефіцієнта маски кожного об'єкта.

• **SOLOv2 (2020):** Розроблена Wang та ін., ця модель безпосередньо прогнозує об'єкт та категорію інстансу, не покладаючись на вилучення пропозиційних коробок та післяобробку.

• **BlendMask (2020) [15]:** Комбінує переваги підходів "зверху-вниз" та "знизу-вгору" для досягнення відмінних результатів інстанс-сегментації.

• **FastInst (2023):** Ця модель є простою та ефективною системою на основі запитів для інстанс-сегментації в реальному часі. FastInst може працювати з швидкістю в реальному часі (тобто 32.5 кадрів в секунду) із точністю AP більше 40 (тобто 40.5 AP) на тестовому наборі даних COCO. Ця модель використовує інноваційні підходи, такі як керовані активацією запити, двошарова стратегія оновлення та навчання, кероване масками реальних даних.

• **MaskFormer (2021) [16]:** Ця модель є багатозадачною архітектурою сегментації зображень, реалізованою за допомогою трансформерів. Вона вважається однією з найсучасніших моделей у цій області.

1.2.1 Огляд Mask R-CNN

Mask R-CNN є розширенням моделі Faster R-CNN, додавши гілку для прогнозування маски, яка допомагає у сегментації окремих об'єктів.

- **Backbone:** Як і в Faster R-CNN, основа мережі, яка зазвичай є згортковою нейронною мережею (CNN), використовується для вилучення особливостей з вхідного зображення.

- **Region Proposal Network (RPN):** Пропонує області зображення, які можуть містити об'єкти.

- **ROIAlign [8]:** Забезпечує вирівнювання регіонів інтересу (ROI) з виходами основної мережі.

- **Класифікація, регресія та маска:** Для кожного ROI виконуються три паралельні операції: визначення класу об'єкта, коректування області об'єкта та визначення маски для об'єкта.

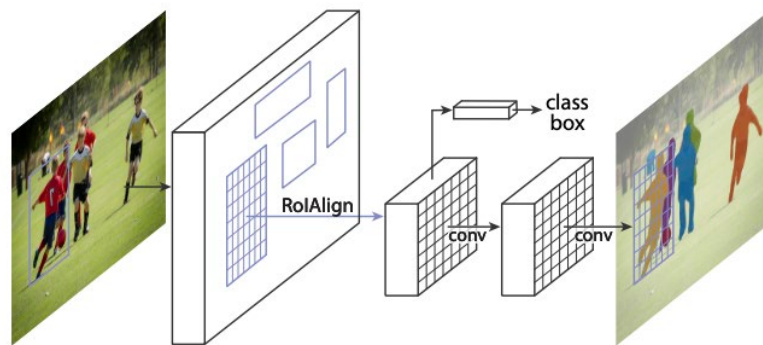


Рисунок 1.1. Структура Mask R-CNN для сегментації екземплярів. [8]

Математичні деталі:

- Згортка: $F_{conv} = Conv(X, W_{conv}) + b_{conv}$

- Активація: $F_{relu} = \max(0, F_{conv})$

- ROIAlign: Ця операція відрізняється від традиційного ROI Pooling [9, 17] тим, що вона забезпечує підгонку, забезпечуючи, що маска відповідає реальному розташуванню об'єкта.

- Маска: Визначення маски використовує декодуючу структуру для відновлення маски з виходів ROIAlign.

Mask R-CNN демонструє вражаючу ефективність в інстанс сегментації, інтегруючи детекцію об'єктів і сегментацію в єдиній моделі.

1.2.2 Огляд PANet

В області інстанс-сегментації, що динамічно розвивається, виникає важлива потреба в підвищенні точності та ефективності ідентифікації та сегментації об'єктів у зображеннях. Серед найновіших проривів у цій сфері є Path Aggregation Network (PANet), що представляє собою новаторську архітектуру, спрямовану на вирішення ключових викликів у інстанс-сегментації. PANet інтегрує низку інноваційних підходів для покращення потоку інформації всередині мережі, сприяючи точнішому та ефективнішому виявленню та сегментації об'єктів. Ця архітектура знаменує собою важливий крок у розвитку технологій комп'ютерного зору, пропонуючи значні переваги у порівнянні з традиційними підходами інстанс-сегментації.

Bottom-up Path Augmentation [12] у PANet є ключовим нововведенням, яке покращує потік інформації в архітектурі мережі. Традиційно, у багатопланових згорткових мережах, вищі рівні зосереджуються на забезпеченні глобального контексту, тоді як нижні рівні надають детальну локалізацію. У більшості випадків, ці нижні рівні не включаються прямо у процес сегментації, що обмежує ефективність інформаційного потоку.

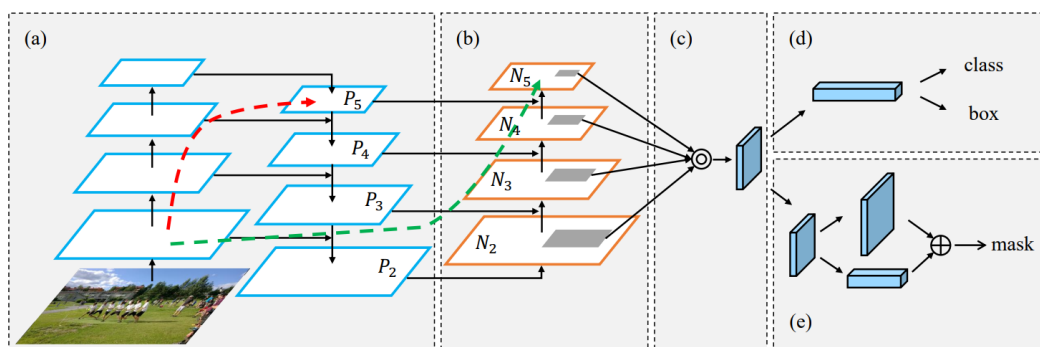


Рисунок 1.2. Ілюстрація рамки PANet. (a) магістраль FPN. (b) Розширення шляху знизу вгору. (c) Адаптивне об'єднання функцій. (d) Коробкова гілка. (e)

Повністю пов'язаний синтез. Зауважте, що для стислості ми опускаємо розмір каналу карт функцій у (a) і (b). [12]

Bottom-up Path Augmentation виправляє це, включаючи нижні рівні безпосередньо у ієрархію функцій, тим самим забезпечуючи більш швидкий і прямий потік інформації від нижніх до верхніх шарів. Це підвищує якість локалізації об'єктів, сприяючи точнішій інстанс-сегментації. Цей підхід особливо корисний у складних візуальних сценах, де важлива точна локалізація об'єктів.

Якщо коротко, то основна ідея полягає у створенні "шляхів" від нижніх рівнів, які містять деталізовану інформацію, до верхніх рівнів, які відповідають за глобальний контекст:

- Використання каскадних зв'язків між різними рівнями мережі.
- Кожен новий шар N_i генерується шляхом об'єднання відповідного шару P_i з попереднього рівня і відповідного вихідного шару P_{i+1} .
- Процес включає використання 3×3 згорткових шарів із наступними операціями ReLU.

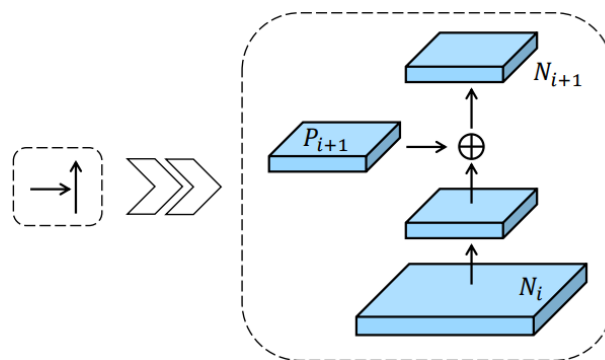


Рисунок 1.3. Ілюстрація нашого будівельного блоку розширення шляху знизу вгору

Adaptive Feature Pooling [12] в PANet є інноваційним компонентом, який покращує спосіб використання та агрегування інформації з різних рівнів мережі. У відповідності до FPN, пропозиції (пропозиційні коробки) призначаються до

різних рівнів функцій в залежності від їхнього розміру. Однак, такий підхід може призводити до неоптимальних результатів, оскільки важливість функцій може не завжди бути тісно пов'язана з рівнем, до якого вони належать.

Adaptive Feature Pooling вирішує цю проблему, дозволяючи кожній пропозиції доступатися до функцій з усіх рівнів для прогнозування. Це означає, що навіть маленькі пропозиції можуть використовувати функції вищих рівнів, які містять більш багатий контекст, а функції нижніх рівнів, які містять більше деталей та високу точність локалізації, доступні для більших пропозицій.

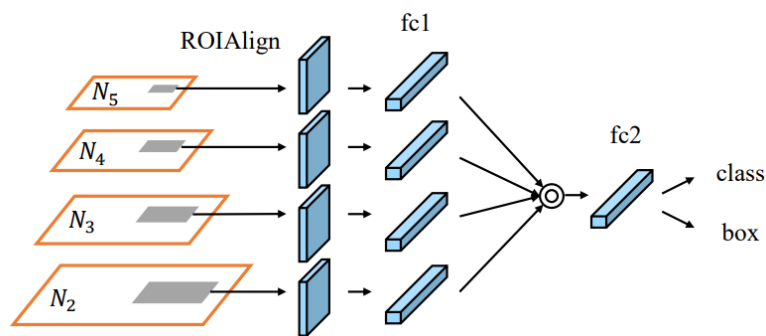


Рисунок 1.4. Ілюстрація адаптивного об'єднання функцій на гілці блоку.

Структура Adaptive Feature Pooling виконується просто: для кожної пропозиції використовується ROIAAlign для збору функцій з кожного рівня, а потім використовується операція злиття (елемент-максимум або сума) для об'єднання функцій з різних рівнів. Цей підхід підвищує точність прогнозування, використовуючи різноманітність інформації, доступної в усій ієрархії функцій.

1.2.3 Огляд Hybrid Task Cascade (HTC)

Основна ідея HTC полягає у тому, що вона ефективно інтегрує каскадну архітектуру в процес сегментації екземплярів, відрізняючись від попередніх методів.

НТС розроблено для вирішення певних недоліків у традиційних підходах до сегментації екземплярів. Особливо важливим є спосіб, яким НТС поєднує завдання виявлення об'єктів та їх сегментації. Замість того, щоб обробляти ці завдання окремо, НТС реалізує їх чергування в рамках одного, багатоступеневого процесу. Таке переплетення завдань не тільки забезпечує більш глибоке та точне розуміння зображень, але й дозволяє кожному етапу вивчати та використовувати інформацію, отриману на попередніх етапах, покращуючи загальні результати сегментації.

Цей підхід дозволяє ефективно обмінюватися інформацією між завданнями виявлення та сегментації на кожному етапі. Завдяки цьому, кожен етап може коригувати та покращувати результати попереднього етапу, використовуючи здобуті знання. Наприклад, інформація, отримана на етапі виявлення, може допомогти уточнити межі сегментації, а зворотно, результати сегментації можуть сприяти більш точному виявленню об'єктів.

Таке взаємодіяння не тільки підвищує точність ідентифікації та сегментації об'єктів, але й робить процес більш адаптивним до складних зображень, де об'єкти можуть бути частково закриті або переплутані з фоном. Впровадження такого інтегрованого підходу в НТС демонструє значний прогрес у вирішенні складних завдань комп'ютерного зору, що робить його важливим внеском у розвиток технологій сегментації екземплярів.

Використання просторових контекстів [13] у НТС є важливою частиною для підвищення ефективності визначення та сегментації об'єктів на зображеннях. Автори вводять додаткову гілку для передбачення семантичної сегментації на рівні пікселів для всього зображення, яка використовує повністю згорткову архітектуру та спільно навчається з іншими гілками. Ця особливість семантичної сегментації є сильним доповненням до існуючих характеристик прямокутника та маски, тому їх комбінують разом для покращення передбачень.

Цей підхід дозволяє моделі краще розрізняти об'єкти переднього плану від заплутаного тла, використовуючи просторові контексти як ефективну підказку. Таке використання семантичної інформації на рівні пікселів поліпшує загальну якість сегментації, оскільки дозволяє моделі використовувати більш глибокі знання про структуру та контекст зображення.

Для покращення розрізнення переднього плану від заплутаного тла використовуються просторові контексти. Додавання додаткової гілки для передбачення семантичної сегментації на рівні пікселів по всьому зображенню, яка використовує повністю згорткову архітектуру, є ключовим елементом у цьому процесі. Ця гілка семантичної сегментації є важливим доповненням до існуючих характеристик прямокутника та маски.

Для покращення розрізнення переднього плану від заплутаного тла використовуються просторові контексти. Додавання додаткової гілки для передбачення семантичної сегментації на рівні пікселів по всьому зображенню, яка використовує повністю згорткову архітектуру, є ключовим елементом у цьому процесі. Ця гілка семантичної сегментації є важливим доповненням до існуючих характеристик прямокутника та маски.

У реалізації використовується простий підхід [13], згідно з яким:

$$F(x_{mask}^t, m^{t-1}) = x_{mask}^t + G_t(m^{t-1})$$

де x_{mask}^t є репрезентацією маски перед деконволюційним шаром з просторовим розміром 14×14 . На етапі t необхідно відправити всі попередні маскові головки з RoI поточного етапу для обчислення m^{t-1} . Далі виконується послідовність обчислень [13]:

$$m^{-1} = M_1^{-}(x_{mask}^t)$$

$$m^{-2} = M_2^{-}(F(x_{mask}^t, m^{-1}))$$

і так далі до:

$$m^{t-1} = M_t^{-1}(F(x_{mask}^t, m^{t-2}))$$

Ці обчислення демонструють взаємодію між різними компонентами моделі для підвищення точності передбачення.

Поступове удосконалення характеристик реалізується шляхом введення потоку інформації між гілками масок, подаючи характеристики маски попередньої стадії на поточну. Цей прямий шлях між гілками масок дозволяє записати пайплайн наступним чином [13]:

$$x_{box}(t) = P(x_{rt-1})$$

$$x_{mask}(t) = P(x_{rt})$$

$$r(t) = B_t(x_{box}(t))$$

$$m(t) = M_t(F(x_{mask}(t), m(t-1)))$$

де $m(t-1)$ позначає проміжну характеристику M_{t-1} і використовується як представлення маски стадії $t-1$. F є функцією, яка поєднує характеристики поточної стадії та попередньої. Такий потік інформації дозволяє поступово удосконалювати маски замість передбачення масок на поступово уточнених обмежувальних рамках.

1.2.4 Огляд SOLOv2: архітектура, основні компоненти, специфікації.

SOLOv2 (Segmenting Objects by Locations, версія 2) представляє собою відносно новий підхід [11] до задачі інстанс сегментації.

Архітектура: SOLOv2 базується на архітектурі одноетапного детектора, яка усуває необхідність в анкерах та пропозиціях регіонів, що робить його швидшим та простішим за інші підходи.

Backbone: SOLOv2 може використовувати будь-яку популярну архітектуру глибокого навчання як основу (наприклад, ResNet [18]). Ця частина відповідає за витягнення ознак з вхідного зображення.

Голова: Ця частина відповідає за генерацію масок і визначення класів. Вона має дві гілки: одна для прогнозування категорії об'єкта і інша для прогнозування маски.

Основні компоненти:

1. Динамічні згорткові ядра: Замість того, щоб намагатися прогнозувати маску для кожного можливого регіону зображення, SOLOv2 генерує динамічне згорткове ядро для кожного об'єкта, базуючись на його розташуванні. Це дає можливість зосередитися на конкретних деталях об'єкта.

2. Matrix NMS: Для більш ефективної обробки масок і уникнення перекриття, SOLOv2 пропонує Matrix NMS - новий підхід до Non-Maximum Suppression для масок, який забезпечує швидке видалення дублюючих об'єктів.

3. Однорідне створення масок: SOLOv2 використовує уніфікований підхід для генерації масок, що дозволяє отримати більш точні контури об'єктів.

SOLOv2 є перспективним і інноваційним підходом до інстанс сегментації, який комбінує в собі швидкість, точність та гнучкість.

Для глибокого розуміння SOLOv2 важливо розглянути математичні аспекти, що лежать в основі його роботи. Розглянемо ключові математичні компоненти:

Формування динамічного ядра: SOLOv2 використовує динамічні згорткові ядра для прогнозування масок. Для об'єкта з координатами (x, y) на зображенні його ядро $K_{x, y}$ визначається на основі вивчених параметрів. При цьому кожне ядро використовується для генерації маски в певному регіоні зображення.

Формування маски: За допомогою динамічного ядра маска об'єкта $M_{x,y}$ отримується шляхом згортки ядра з відповідною частиною зображення: $M_{x,y} = I * K_{x,y}$ де I - це зображення, а $*$ вказує на операцію згортки.

Matrix NMS: Після отримання множини масок для різних об'єктів на зображенні застосовується Matrix NMS. Ця операція дозволяє визначити, які маски є надлишковими і повинні бути видалені. При цьому для кожної пари масок M_i та M_j розраховується міра схожості $S(M_i, M_j)$, і якщо вона перевищує певний поріг, менш імовірна маска видаляється. Matrix NMS є ключовим компонентом в SOLOv2 для покращення ефективності обробки при великій кількості масок. Він використовується для визначення, які з отриманих масок є перекриваючимися, і відсіву таких масок на основі їхньої взаємної схожості.

Основні кроки Matrix NMS:

Обчислення IoU (Intersection over Union): Для кожної пари масок M_i та M_j розраховується їх IoU: $\text{IoU}(M_i, M_j) = (M_i \cap M_j) / (M_i \cup M_j)$ де \cap - це перетин, а \cup - об'єднання двох масок.

Побудова матриці IoU: Створюється матриця розміром $N \times N$ (де N - кількість масок), що містить значення IoU для всіх пар масок.

Обчислення ваг: За допомогою матриці IoU і вихідних оцінок довіри для кожної маски розраховуються ваги: $w_i = \sum_{j=1}^N (1 - \text{IoU}(M_i, M_j)) \times \text{score}_j$ де score_j - це оцінка довіри для маски M_j .

Оновлення оцінок довіри: Оцінки довіри для кожної маски оновлюються: $\text{score}_i' = w_i \times \text{score}_i$

Відсів масок: Маски, для яких оновлена оцінка довіри score_i' є менше певного порогу, видаляються.

Matrix NMS є важливим кроком в SOLOv2, що дозволяє зменшити кількість непотрібних або перекриваючих масок і забезпечити більш точний результат інстанс-сегментації.

Функція втрат: Для навчання моделі SOLOv2 використовується комбінована функція втрат, що враховує як втрати класифікації, так і втрати регресії для правильного розташування масок: $L = L_{cls} + \lambda L_{mask}$ де L_{cls} - втрати класифікації, L_{mask} - втрати регресії для масок, а λ - гіперпараметр, що визначає важливість кожного компонента.

Ці компоненти разом формують основу роботи SOLOv2, дозволяючи йому ефективно виділяти окремі об'єкти на зображеннях та присвоювати їм відповідні класи.

Для підтримки найкращого виконання моделі SOLOv2, вона використовує дві ключові втрати: втрату класифікації L_{cls} та втрату регресії для масок L_{mask} . Давайте розглянемо обидві ці втрати детально.

1.2.5 Огляд BlendMask

BlendMask ефективно комбінує інформацію на рівні інстанцій з семантичною інформацією, використовуючи меншу кількість каналів для передбачення густої позиційно-чутливої інформації на рівні пікселів. Основним внеском є модуль змішування (blender module [15]), який поєднує елементи з підходів зверху-вниз та знизу-вгору у сегментації інстанцій. BlendMask здатний ефективно передбачати густі позиційно-чутливі характеристики інстанцій з дуже невеликою кількістю каналів і вчити карти уваги для кожної інстанції за допомогою всього лише одного шару згортки, що робить його швидким у висновках.

BlendMask може бути легко інтегрований з сучасними одноетапними рамками виявлення об'єктів і перевершує Mask R-CNN за тим самим графіком навчання, будучи при цьому на 20% швидшим.

Архітектура BlendMask включає в себе нижній модуль, який передбачає карти оцінок, які називають базами В. В має форму $N \times K \times H_s \times W_s$, де N - розмір пакета, K - кількість баз, $H \times W$ - вхідний розмір, а s - крок виведення карт оцінок. У експериментах використовувався декодер DeepLabV3+ [19]. Вхід

для нижнього модуля може бути характеристиками з backbone, як у звичайних мережах семантичної сегментації, або з пірамід характеристик, як у YOLACT та Panoptic FPN [20]. Для верхнього шару також додається один шар згортки на кожну з вей виявлення для передбачення уваг верхнього рівня A.

BlendMask базується на передовому одноетапному детекторі об'єктів FCOS [21] з мінімальними змінами. Нижній модуль використовує характеристики з backbone або з FPN для передбачення набору баз. Додається один шар згортки до кожної з вей виявлення для виробництва масок уваги разом з кожним прогнозом обмежувальної рамки. Для кожної передбаченої інстанції модуль змішування обрізає бази з її обмежувальною рамкою і лінійно комбінує їх відповідно до навчених карт уваги.

Архітектура BlendMask включає в себе ряд унікальних елементів та конфігурацій [15]:

- Елементи регіонів R та бали S об'єднуються за допомогою елементного добутку для кожної сутності rd sd, а потім сумуються по виміру K, щоб отримати логіт маски md:

$$md = \sum_{k=1}^K skd \odot rkd \quad \forall d \in \{1, \dots, D\}$$

де k - індекс бази.

Гіперпараметри BlendMask, які можна налаштувати:

- R - роздільна здатність RoI нижнього рівня
- M - роздільна здатність прогнозування верхнього рівня
- K - кількість баз

- Вхідні характеристики нижнього модуля - це можуть бути характеристики з backbone або FPN

- Метод зразкування для нижніх баз - найближчий сусід або білінійний пулінг

- Метод інтерполяції для уваг верхнього рівня - найближчий сусід або білінійне збільшення.

Модуль змішування (blender module) є ключовою частиною BlendMask. Він комбінує бази, чутливі до позиції, відповідно до уваг, для генерації кінцевого прогнозу. Вхідні дані модуля змішування - це нижні бази B , обрані уваги верхнього рівня A та пропозиції обмежувальних рамок P . Спочатку використовується RoIPooler з Mask R-CNN для обрізання баз з кожної пропозиції pd та зміни розміру області до фіксованого розміру карти ознак $R \times R$:

$$rd = \text{RoIPool}_{R \times R}(B, pd) \quad \forall d \in \{1, \dots, D\} [9, 17]$$

де $R \times R$ - розмір фіксованої карти ознак, B - бази, pd - пропозиції обмежувальних рамок, d - індекс пропозиції.

Модуль змішування BlendMask поєднує бази, чутливі до позиції, згідно з увагами, щоб генерувати кінцевий прогноз. На вхід модуля змішування надходять нижні бази B , обрані уваги верхнього рівня A та пропозиції обмежувальних рамок P . Використовується RoIPooler з Mask R-CNN для обрізання баз з кожною пропозицією та зміни розміру регіону до фіксованого розміру карти ознак $R \times R$. Для обох роздільних здатностей, вирівнювання білінійного зразкування може покращити продуктивність майже на 2AP. Використання вирівняних характеристик для нижнього рівня є більш критичним, оскільки саме там відбувається передбачення.

У BlendMask RoIPooler та RoIAlign використовуються для покращення продуктивності шляхом вирівнювання білінійного зразкування, що може підвищити продуктивність майже на 2AP. Використання вирівняних характеристик для нижнього рівня є більш важливим, оскільки саме на цьому рівні відбувається передбачення

1.2.6 Огляд FastInst

FastInst представляє собою новаторську модель для інстанц-сегментації, що використовує підходи, засновані на запитах, для покращення ефективності. Основна архітектура FastInst включає в себе три основні модулі: backbone, декодер пікселів та трансформер-декодер. Модель обробляє вхідне зображення $I \in R^{H \times W \times 3}$ за допомогою backbone та отримує три карти характеристик C3, C4, та C5, роздільні здатності яких становлять 1/8, 1/16, та 1/32 від роздільної здатності вхідного зображення відповідно. Ці карти характеристик проектуються у карти з 256 каналами за допомогою шару згортки 1×1 та подаються до декодера пікселів.

FastInst використовує керовані активацією запити (instance activation-guided queries), що вибираються з вихідних карт характеристик декодера пікселів. Це надихнуто методом Deformable DETR [22], який вибирає області запитів із пірамідальних характеристик для виявлення об'єктів. Конкретно, враховуючи вихідні карти характеристик декодера пікселів, додається допоміжний класифікаційний блок, за яким слідує softmax-активація поверх карти характеристик E4, щоб отримати прогноз ймовірності класу $p_i \in \Delta^{K+1}$ для кожного пікселя, де Δ^{K+1} - це $(K + 1)$ -вимірний симплекс ймовірності, K - кількість класів, додана одиниця для "немає об'єкта" (\emptyset), i - індекс пікселя, а допоміжний класифікаційний блок складається з двох шарів згортки з розмірами ядра 3×3 та 1×1 відповідно.

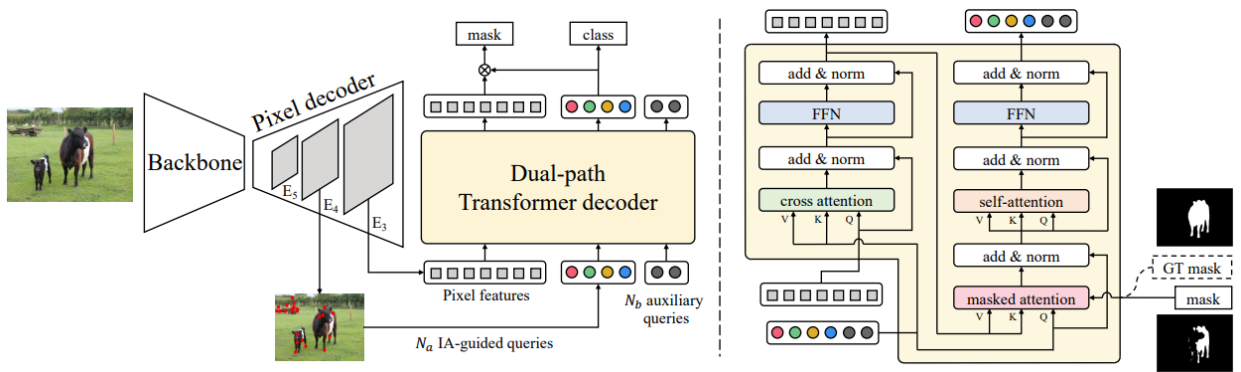


Рисунок 1.2.5. Огляд моделі. FastInst складається з трьох модулів: магістралі, піксельного декодера та трансформаторного декодера. Основний і піксельний декодер витягує й уточнює багатомасштабні функції. Декодер Transformer вибирає запити, керовані активацією екземпляра N_a (запити, керовані IA) із карти функцій E4 і об'єднує їх із N_b допоміжними запитами, які можна вивчати, як початкові запити. Потім, приймаючи початкові запити та зведену карту функцій E3 як вхідні дані, декодер Transformer виконує класифікацію об'єктів і сегментацію на кожному шарі за допомогою стратегії подвійного оновлення. Під час тренінгу ми впроваджуємо базове навчання (GT) під керуванням маски, щоб покращити ефективність замаскованої уваги. [1]

Пишучи про Deformable DETR треба написати про багатоголовий механізм уваги (MultiHeadAttn) [23]. Нейронні мережі, які називаються Трансформерами, базуються на архітектурі мережі з механізмами уваги для машинного перекладу. Для даного елемента запиту (наприклад, цільового слова в вихідному реченні) та набору ключових елементів (наприклад, слів джерела в вхідному реченні), модуль багатоголової уваги адаптивно агрегує вміст ключів відповідно до ваг уваги, які вимірюють сумісність пар запитів та ключів. Для того, щоб модель могла концентруватися на вмісті з різних підпросторів представлення та різних позиціях, виходи різних голів уваги лінійно агрегуються з навчуваними вагами. Нехай $q \in \Omega_q$ індексує елемент запиту з характеристикою представлення $z_q \in R^C$, і $(k \in \Omega_k)$ індексує ключовий елемент з характеристикою представлення $x_k \in R^C$, де C є розмірністю ознак, Ω_q та Ω_k визначають набір елементів запитів

та ключів відповідно. Тоді характеристика багатоголової уваги розраховується як

$$\text{MultiHeadAttn}(z_q, x) = \sum_{m=1}^M W_m \sum_{k \in \Omega_k} A_{mqk} \cdot W_{0m} x_k$$

де m індексує голову уваги, $W_{0m} \in R^{C_v \times C}$ та $W_m \in R^{C \times C_v}$ є навчуваними вагами (за замовчуванням $C_v = C/M$). Ваги уваги $A_{mqk} \propto \exp\left\{\frac{z_q^T U_m^T V_m x_k}{\sqrt{C_v}}\right\}$ нормалізуються як $\sum_{k \in \Omega_k} A_{mqk} = 1$, де $U_m, V_m \in R^{C_v \times C}$ також є навчуваними вагами. Для розрізнення різних просторових позицій, характеристики представлення z_q та x_k зазвичай є конкатенацією/сумою вмісту елементів та позиційних вкладень.

Основна проблема застосування уваги трансформера на картах характеристик зображення полягає в тому, що вона охоплює всі можливі просторові локації. Щоб вирішити це, в моделі пропонується модуль деформовної уваги. Надихнувшись деформовною згорткою, модуль деформовної уваги звертається лише до невеликої кількості ключових точок зразкування навколо опорної точки, незалежно від просторового розміру карт характеристик. Призначаючи лише невелику фіксовану кількість ключів для кожного запиту, можна пом'якшити проблеми збіжності та просторової роздільної здатності характеристик.

Для даної вхідної карти характеристик $x \in R^{C \times H \times W}$, нехай q індексує елемент запиту з вмістовою характеристикою z_q та 2-вимірною опорною точкою p_q , деформовна характеристика уваги розраховується як

$$\text{DeformAttn}(z_q, p_q, x) = \sum_{m=1}^M W_m \sum_{k=1}^K A_{mqk} \cdot W_{0m} x(p_q + \Delta p_{mqk}) \quad [22]$$

де m індексує голову уваги, k індексує вибрані ключі, а K є загальною кількістю вибраних ключів ($K \approx HW$). Δp_{mqk} та A_{mqk} позначають зміщення зразкування та вагу уваги k -ї точки зразкування в m -й голові уваги відповідно. Скалярна вага уваги A_{mqk} лежить в діапазоні $[0, 1]$, нормалізована за допомогою $\sum_{k=1}^K A_{mqk} = 1$. $\Delta p_{mqk} \in \mathbb{R}^2$ є 2-вимірними дійсними числами з неконтрольованим діапазоном. Оскільки $p_q + \Delta p_{mqk}$ є дробовим, застосовується білінійна інтерполяція при обчисленні $x(p_q + \Delta p_{mqk})$. Обидва Δp_{mqk} та A_{mqk} отримуються за допомогою лінійної проєкції над характеристикою запиту z_q . На практиці, характеристика запиту z_q подається на лінійний проєктор з 3МК каналами, де перші 2МК канали кодують зміщення зразкування Δp_{mqk} , а решта МК каналів подається на оператор `softmax` для отримання ваг уваги A_{mqk} .

Модуль деформовної уваги розроблений для обробки згорткових карт характеристик як ключових елементів. Нехай N_q буде кількістю елементів запиту, коли МК є відносно малим, складність модуля деформовної уваги становить $O(2N_q C^2 + \min(HWC^2, N_q KC^2))$. Коли він застосовується в кодері DETR, де $N_q = HW$, складність стає $O(HWC^2)$, що є лінійною складністю з просторовим розміром. Коли він застосовується як модулі перехресної уваги в декодері DETR, де $N_q = N$ (N - кількість запитів об'єктів), складність стає $O(NKC^2)$, що не залежить від просторового розміру HW .

Більшість сучасних рамок виявлення об'єктів отримують вигоду з використання багатомасштабних карт характеристик. Наш запропонований модуль деформовної уваги може бути природно розширений для багатомасштабних карт характеристик.

Нехай $\{x_l\}_{l=1}^L$ будуть вхідними багатомасштабними картами характеристик, де $x_l \in \mathbb{R}^{C \times H_l \times W_l}$. Нехай $\widehat{p}_q \in [0, 1]^2$ будуть нормалізованими координатами

опорної точки для кожного елемента запиту q , тоді багатомасштабний модуль деформовної уваги застосовується як

$$\begin{aligned} \text{MSDeformAttn}(z_q, \widehat{p}_q, \{x_l\}_{l=1}^L) \\ = \sum_{m=1}^M W_m \sum_{l=1}^L \sum_{k=1}^K A_{mlqk} \cdot W_{0m} x_l(\phi_l(\widehat{p}_q) + \Delta p_{mlqk}) \end{aligned}$$

де m індексує голову уваги, l індексує рівень вхідної характеристики, а k індексує точку зразкування. Δp_{mlqk} та A_{mlqk} позначають зміщення зразкування та вагу уваги k -ї точки зразкування на l -му рівні характеристик та m -й голові уваги відповідно. Скалярна вага уваги A_{mlqk} нормалізується за допомогою $\sum_{l=1}^L \sum_{k=1}^K A_{mlqk} = 1$. Тут використовуються нормалізовані координати $\widehat{p}_q \in [0,1]^2$ для ясності формулювання масштабу, в якому нормалізовані координати $(0, 0)$ та $(1, 1)$ вказують на верхній лівий та нижній правий кути зображення відповідно. Функція $\phi_l(\widehat{p}_q)$ в рівнянні 3 перераховує нормалізовані координати \widehat{p}_q до вхідної карти характеристик l -го рівня. Багатомасштабна деформовна увага дуже схожа на попередню версію одного масштабу, за винятком того, що вона вибирає LK точок з багатомасштабних карт характеристик замість K точок з одномасштабних карт характеристик.

Запропонований модуль уваги деградує до деформовної згортки, коли $L = 1$, $K = 1$, і $W_{0m} \in R^{C_v \times C}$ фіксується як тотожність матриці. Деформовна згортка розроблена для одномасштабних вхідних даних, зосереджуючись лише на одній точці зразкування для кожної голови уваги. Однак, наша багатомасштабна деформовна увага звертається до декількох точок зразкування з багатомасштабних входів. Запропонований (багатомасштабний) модуль деформовної уваги можна також сприймати як ефективний варіант уваги трансформера, де вводиться механізм попереднього фільтрування завдяки деформовним місцям зразкування. Коли точки зразкування охоплюють усі

можливі локації, запропонований модуль уваги стає еквівалентним увазі трансформера.

Деформований Трансформер Кодер. В моделі замінюються модулі уваги Трансформера, що обробляють карти характеристик у DETR, на запропонований багатомасштабний модуль деформовної уваги. Як вхідні, так і вихідні дані кодера є багатомасштабними картами характеристик з однаковими роздільними здатностями. У кодері витягуються багатомасштабні карти характеристик $\{x_l\}_{l=1}^{L-1}$ (де $L = 4$) з вихідних карт характеристик етапів C3 до C5 у ResNet (трансформовані за допомогою згортки 1×1), де C_l має роздільну здатність на 2^l нижче, ніж вхідне зображення. Карта характеристик найнижчої роздільної здатності x_L отримується за допомогою згортки 3×3 з кроком 2 на останньому етапі C5, позначеному як C6. Усі багатомасштабні карти характеристик мають $C = 256$ каналів. Зазначимо, що верхньо-нижня структура у FPN не використовується, оскільки наш запропонований багатомасштабний модуль деформовної уваги сам по собі може обмінюватися інформацією між багатомасштабними картами характеристик.

При застосуванні багатомасштабного модуля деформовної уваги у кодері, вихідні дані є багатомасштабними картами характеристик з тими ж роздільними здатностями, що й на вході. Як ключові, так і елементи запиту є пікселями з багатомасштабних карт характеристик. Для кожного пікселя запиту опорна точка є сама себе. Щоб ідентифікувати, на якому рівні характеристик знаходиться кожен піксель запиту, додається вкладення рівня масштабу, позначене як e_l , до представлення характеристик, на додаток до позиційного вкладення. На відміну від позиційного вкладення з фіксованим кодуванням, вкладення рівня масштабу $\{e_l\}_{l=1}^L$ випадково ініціалізуються та спільно навчаються з мережею.

У декодері є модулі перехресної уваги та самоуваги [2]. Елементи запиту для обох типів модулів уваги є запитами об'єктів. У модулях перехресної уваги

запити об'єктів витягують характеристики з карт характеристик, де ключові елементи є вихідними картами характеристик від кодера. У модулях самоуваги запити об'єктів взаємодіють один з одним, де ключові елементи є запитами об'єктів. Оскільки наш запропонований модуль деформовної уваги розроблений для обробки згорткових карт характеристик як ключових елементів, далі замінюється кожен модуль перехресної уваги на багатомасштабний модуль деформовної уваги, залишаючи модулі самоуваги без змін. Для кожного запиту об'єкта 2-вимірна нормалізована координата опорної точки \widehat{p}_q передбачається з вкладення запиту об'єкта через навчану лінійну проекцію, за якою слідує функція сигмоїда.

Оскільки багатомасштабний модуль деформовної уваги витягує характеристики зображення навколо опорної точки, детектору дозволяється передбачати обмежувальну рамку як відносні зміщення відносно опорної точки, щоб додатково зменшити складність оптимізації. Опорна точка використовується як початкове припущення центру рамки. Детектор передбачає відносні зміщення відносно опорної точки. Таким чином, навчений декодер уваги матиме сильний зв'язок з передбаченими обмежувальними рамками, що також прискорює збіжність навчання.

Замінивши модулі уваги Трансформера на модулі деформовної уваги у DETR, створюється ефективна та швидко збіжна система виявлення, яка називається Deformable DETR.

Повертаючись до FastInst, опишемо Двошляховий Трансформер Декодер [1]. Після вибору N_a ІА-керованих запитів з підкладеної карти характеристик, вони конкатенуються з N_b додатковими навчуваними запитамі, щоб отримати загальні запити Q , де додаткові навчувані запити використовуються для сприяння групуванню фонових піксельних характеристик і надання загальної незалежної від зображення інформації в подальшому двошляховому процесі оновлення. Потім загальні запити Q разом з розгорнутими піксельними

характеристиками високої роздільності $1/8$ X подаються до Трансформер декодера. У Трансформер декодері додається позиційні вкладення для запитів Q та піксельних характеристик X , за якими слідує послідовні шари Трансформер декодера для їх оновлення. Один шар Трансформер декодера містить одне оновлення піксельних характеристик і одне оновлення запиту. Весь процес схожий на алгоритм кластеризації EM (Expectation–Maximization) [24]. Крок E: оновлення піксельних характеристик згідно з центрами (запитами), до яких вони належать; Крок M: оновлення центрів кластерів (запитів). Порівняно з одношляховою стратегією оновлення, двошляхова стратегія оновлення співоптимізує як піксельні характеристики, так і запити, зменшуючи залежність від важких декодерів пікселів та отримуючи більш деталізовані вбудовані характеристики. Нарешті, використовуються уточнені піксельні характеристики та запити для передбачення класів об'єктів та масок сегментації на кожному шарі.

Інформація про розташування є критичною для відрізнення різних інстанцій зі схожою семантикою, особливо для об'єктів одного класу. Замість непараметричних синусоїдальних позиційних вкладень, використовується навчувані позиційні вкладення, які можуть покращити швидкість висновку моделі без погіршення продуктивності. Конкретно, використовується навчуване просторове позиційне вкладення фіксованого розміру $P \in R^{S \times S \times 256}$, де S - просторовий розмір, який емпірично встановлюється як округлений квадратний корінь з кількості IA-керованих запитів N_a . Під час передачі інтерполюється P до двох різних розмірів. Один має той же розмір, що й E3, який потім розгортається як позиційні вкладення для піксельних характеристик X ; інший має той же розмір, що й E4, з якого вибираються позиційні вкладення для IA-керованих запитів згідно з їхніми розташуваннями $\{(x_i, y_i)\}_{i=1}^{N_a}$ на карті характеристик E4. Додаткові навчувані запити використовують додаткові N_b навчувані позиційні вкладення.

Оновлення піксельних характеристик. Спочатку оновлюються піксельні характеристики. Враховуючи розгорнуті піксельні характеристики X та запити Q , трубопровід оновлення піксельних характеристик складається з шару перехресної уваги та шару подачі вперед. Позиційні вкладення додаються до запитів та ключів на кожному шарі перехресної уваги. Для оновлення піксельних характеристик не використовується самоувага, яка вводить значний обчислювальний та пам'ятний витрати через довгу довжину послідовності піксельних характеристик. Глобальні характеристики можуть бути агреговані через перехресну увагу на запитах.

В моделі використовується увага з маскою, за якою слідує самоувага та мережа прямого розповсюдження для оновлення запиту, як у Mask2Former. Увага з маскою обмежує увагу кожного запиту лише у межах переднього плану передбаченої маски з попереднього шару, і припускається, що контекстна інформація збирається через наступну самоувагу. Такий дизайн значно покращив продуктивність моделей на основі запитів у завданнях сегментації зображень. Тут позиційні вкладення також додаються до запитів та ключів на кожному шарі уваги з маскою та самоуваги.

Застосовуються дві окремі 3-шарові MLP на верхівці уточнених ІА-керованих запитів на кожному шарі декодера для передбачення класів об'єктів та вбудовування масок відповідно. Кожен ІА-керований запит повинен передбачити ймовірність усіх класів об'єктів, включаючи клас "немає об'єкта" (\emptyset). Лінійна проекція додається до уточнених піксельних характеристик для отримання характеристик маски. Потім вбудовування масок множиться з характеристиками маски, щоб отримати маски сегментації для кожного запиту. Тут параметри MLP та лінійної проекції на кожному шарі Трансформер декодера не діляться, оскільки запити та піксельні характеристики оновлюються по черзі, і їх характеристики можуть бути в різних просторах представлення на різних шарах декодера. Крім того, інстанц-сегментація вимагає оцінки впевненості для кожного прогнозу.

Хоча увага з маскою вводить заздалегідь розріджені знання про увагу, які прискорюють збіжність моделі та покращують продуктивність, вона обмежує поле охоплення кожного запиту і може змусити Трансформер декодер впасти в субоптимальний процес оновлення запиту. Щоб пом'якшити цю проблему, вводиться навчання, кероване маскою Ground Truth (GT). По-перше, використовується маска Ground Truth, яка була співставлена у двосторонньому порядку на останньому шарі, для заміни передбаченої маски, яка використовувалася в увазі з маскою на 1-му шарі. Для запитів, які не співпадають з жодною інстанцією на останньому шарі (включаючи додаткові навчувані запити), використовується стандартна перехресна увага, тобто,

$$M_{li} = M_{gt_j} \text{ якщо } (i, j) \in \sigma, \text{ інакше}$$

де M_{li} є маскою уваги для i -го запиту на 1-му шарі, $\sigma = \{(i, j) \mid i \in \{1, \dots, N_a\}, j \in \{1, \dots, N_{obj}\}\}$ є співставленням останнього шару декодера, і M_{gt} є співставленою маскою Ground Truth для i -го запиту на останньому шарі. Тут N_{obj} позначає кількість цілей Ground Truth. Потім використовується замінена маска уваги M_l разом з оригінальними вихідними запитами та піксельними характеристиками 1-го шару, які були уточнені для кращого керівництва, як вхідні дані для повторного проходження 1-го шару Трансформер декодера. Новий вихід контролюється згідно з фіксованим співставленням σ , узгодженим з результатами двостороннього співставлення останнього шару. Це фіксоване співставлення забезпечує послідовність прогнозів на кожному шарі Трансформер декодера і зберігає обчислювальні витрати на співставлення під час навчання. Завдяки такому керованому навчанню дозволяється кожному запиту бачити весь регіон свого цільового передбаченого об'єкта під час навчання, що допомагає увазі з маскою зосереджуватися на більш відповідному регіоні переднього плану.

Загальна функція втрат для FastInst може бути записана як:

$$L = L_{IA-q} + L_{pred} + L'_{pred}$$

де L_{IA-q} є втратою активації інстанцій допоміжної класифікаційної голови для ІА-керованих запитів, L_{pred} та L'_{pred} є втратою передбачення та втратою, керованою маскою GT, відповідно.

Втрата активації інстанцій. Втрата L_{IA-q} визначається як:

$$L_{IA-q} = \lambda_{cls-q} L_{cls-q}$$

де λ_{cls-q} є гіперпараметром, а L_{cls-q} є втратою перехресної ентропії. Далі використовується алгоритм угорського методу для пошуку оптимального двостороннього співставлення між наборами прогнозів та істинних даних. Для вартості співставлення додається додаткова вартість розташування L_{loc} з вагою λ_{loc} до вищезазначеної класифікаційної вартості.

Втрата передбачення. За попередньою роботою, втрата передбачення L_{pred} для Трансформер декодера визначається як:

$$L_{pred} = \sum_{i=0}^D (\lambda_{ce} L_{ice} + \lambda_{dice} L_{idice}) + \lambda_{cls} L_{icls}$$

де D позначає кількість шарів Трансформер декодера, і $i = 0$ представляє втрату передбачення для ІА-керованих запитів перед подачею в Трансформер декодер, L_{ice} та L_{idice} позначають втрату бінарної перехресної ентропії та втрату кубика для масок сегментації відповідно, а L_{icls} є втратою перехресної ентропії для класифікації об'єктів з вагою "немає об'єкта". λ_{ce} , λ_{dice} та λ_{cls} є гіперпараметрами для збалансування трьох втрат. Аналогічно, використовується алгоритм Угорського алгоритму для пошуку кращого двостороннього співставлення для призначення цілей. І для вартості співставлення додається додаткова вартість розташування $\lambda_{loc} L_{loc}$ для кожного запиту.

Втрата, керована маскою GT. Втрата, керована маскою GT L'_{pred} , подібна до рівняння. Єдина різниця полягає в тому, що вона не враховує втрату 0-го шару і використовує фіксовану стратегію призначення цілей, яка узгоджується з результатом двостороннього співставлення останнього шару Трансформер декодера.

1.2.7 Огляд MaskFormer

Модель MaskFormer виходить з основної ідеї, що класифікація масок є достатньо універсальним методом для вирішення як семантичних, так і інстанц-рівневих завдань сегментації. Традиційно семантична сегментація розглядається як задача класифікації на рівні кожного пікселя, де кожному пікселю присвоюється клас. Однак, метод класифікації масок замість цього передбачає набір бінарних масок, де кожна маска асоційована з одним класовим прогнозом. Такий підхід дозволяє вирішувати завдання сегментації більш гнучко.

MaskFormer використовує декодер Transformer для обчислення набору пар, кожна з яких складається з прогнозу класу та вектора вбудовування маски. Вектор вбудовування маски використовується для отримання бінарного прогнозу маски шляхом скалярного добутку з вбудовуванням на рівні пікселів, отриманим з повністю згорткової мережі. Нова модель вирішує як семантичні, так і інстанц-рівневі завдання сегментації уніфікованим способом: не потрібно змінювати втрати моделі та процедуру навчання. Зокрема, для завдань семантичної та паноптичної сегментації MaskFormer керується однаковою втратою бінарної маски на рівні пікселів та однією втратою класифікації на маску.

Для класифікації на рівні кожного пікселя модель сегментації прагне передбачити розподіл ймовірностей серед усіх можливих K категорій для кожного пікселя зображення $H \times W$: $y = \{p_i | p_i \in \Delta_K\}^{H \cdot W}$. Тут $\Delta_K \in K$ -вимірним симплексом ймовірності. Навчання моделі класифікації на рівні кожного пікселя є прямолінійним: враховуючи мітки істинних категорій $y_{gt} =$

$\{y_{gt_i} | y_{gt_i} \in \{1, \dots, K\}\}^{H \cdot W}$ для кожного пікселя, зазвичай застосовується втрата перехресної ентропії на пікселі (негативна лог-ймовірність), тобто, $L_{pixel-cls}(y, y_{gt}) = \sum_{i=1}^{H \cdot W} -\log p_i(y_{gt_i})$. [16]

Класифікація масок розділяє завдання сегментації на 1) поділ/групування зображення на N регіонів (де N не обов'язково дорівнює K), представлених бінарними масками $\{m_i | m_i \in [0,1]^{H \times W}\}_{i=1}^N$; та 2) асоціювання кожного регіону в цілому з деяким розподілом серед K категорій. Для того, щоб одночасно групувати та класифікувати сегмент, тобто виконувати класифікацію масок, визначається бажаний вихід z як набір з N пар ймовірності-маски, тобто, $z = \{(p_i, m_i)\}_{i=1}^N$. На відміну від передбачення класової ймовірності на рівні кожного пікселя, для класифікації масок розподіл ймовірностей $p_i \in \Delta_{K+1}$ містить додаткову мітку "немає об'єкта" (\emptyset) на додаток до K міток категорій. Мітка \emptyset передбачається для масок, які не відповідають жодній з K категорій. Зауважимо, класифікація масок дозволяє декілька передбачень масок з одним і тим самим асоційованим класом, роблячи її застосовною як до семантичних, так і до інстанц-рівневих завдань сегментації.

Щоб навчити модель класифікації масок, потрібне співставлення σ між набором передбачень z та набором N_{gt} істинних сегментів $z_{gt} = \{(c_{gt_i}, m_{gt_i}) | c_{gt_i} \in \{1, \dots, K\}, m_{gt_i} \in \{0,1\}^{H \times W}\}_{i=1}^{N_{gt}}$. Тут c_{gt_i} є істинним класом i -го істинного сегмента. Оскільки розмір набору передбачень $|z| = N$ і розмір набору істинних даних $|z_{gt}| = N_{gt}$ зазвичай відрізняються, припускається $N \geq N_{gt}$ і доповнюємо набір істинних міток мітками "немає об'єкта" \emptyset для дозволу один-до-одного співставлення.

Для семантичної сегментації, просте фіксоване співставлення є можливим, якщо кількість прогнозів N відповідає кількості міток категорій K . У цьому випадку i -й прогноз співставляється з істинним регіоном з міткою класу i і з \emptyset , якщо регіон з міткою класу i відсутній в істинних даних. У наших експериментах було виявлено, що співставлення на основі двостороннього співставлення

демонструє кращі результати, ніж фіксоване співставлення. На відміну від DETR, який використовує обмежувальні рамки для обчислення вартості призначення між прогнозом z_i та істинними даними z_{gt_j} для задачі співставлення, безпосередньо використовуються класові та маскові прогнози, тобто, $-p_i(c_{gt_j}) + L_{mask}(m_i, m_{gt_j})$, де L_{mask} є втратаю бінарної маски.

Для навчання параметрів моделі, враховуючи співставлення, основна втрата класифікації масок [16] $L_{mask-cls}$ складається з втрати перехресної ентропії класифікації та бінарної втрати маски L_{mask} для кожного передбаченого сегмента:

$$L_{mask-cls}(z, z_{gt}) = \sum_{i=1}^N \left(-\log p_{\sigma(j)}(c_{gt_i}) + 1_{gt_i} L_{mask}(m_{\sigma(j)}, m_{gt_i}) \right)$$

Зауважте, що більшість існуючих моделей класифікації масок використовують допоміжні втрати (наприклад, втрату обмежувальної рамки або втрату дискримінації інстанцій) на додаток до $L_{mask-cls}$.

MaskFormer – нова модель класифікації масок, яка обчислює N пар ймовірність-маска $z = \{(p_i, m_i)\}_{i=1}^N$. Модель містить три модулі: 1) модуль на рівні пікселів, який витягує вбудовування на рівні кожного пікселя, що використовуються для генерації бінарних прогнозів маски; 2) трансформер модуль, де стек шарів Трансформер декодера обчислює N вбудовувань на рівні сегментів; та 3) модуль сегментації, який генерує прогнози $\{(p_i, m_i)\}_{i=1}^N$ з цих вбудовувань. Під час виводу, p_i та m_i збираються у фінальний прогноз.

Модуль на рівні пікселів приймає зображення розміром $H \times W$ на вході. Основна мережа генерує карту характеристик зображення (зазвичай) низької роздільності $F \in R^{C_F \times H_S \times W_S}$, де C_F - кількість каналів, а S - крок карти характеристик. Потім декодер пікселів поступово збільшує роздільність характеристик, щоб генерувати вбудовування на рівні кожного пікселя $E_{pixel} \in R^{C_E \times H \times W}$, де C_E - розмірність вбудовування. Зауважте, що будь-яка модель

сегментації, заснована на класифікації на рівні пікселів, відповідає дизайну модулю на рівні пікселів, включаючи недавні моделі на основі Трансформера. MaskFormer безшовно перетворює таку модель на класифікацію масок.

Трансформер модуль використовує стандартний Трансформер декодер для обчислення з характеристик зображення F та N навчуваних позиційних вкладень (тобто запитів) свого виходу, тобто N вбудовувань на рівні сегментів $Q \in R^{C_Q \times N}$ з розмірністю C_Q , які кодують глобальну інформацію про кожен сегмент, який передбачає MaskFormer. Декодер видає всі прогнози паралельно.

Модуль сегментації застосовує лінійний класифікатор, за яким слідує активація softmax, на верхівці вбудовувань на рівні сегментів Q , щоб отримати прогнози ймовірності класу $\{p_i \in \Delta_{K+1}\}_{i=1}^N$ для кожного сегмента. Зауважте, що класифікатор передбачає додаткову категорію "немає об'єкта" (\emptyset) у випадку, якщо вбудовування не відповідає жодному регіону. Для передбачення маски багат шаровий перцептрон (MLP) з 2 прихованими шарами перетворює вбудовування на рівні сегментів Q на N вбудовувань масок $E_{mask} \in R^{C_E \times N}$ з розмірністю C_E . Нарешті, кожне бінарне передбачення маски $m_i \in [0,1]^{H \times W}$ отримується через скалярний добуток між i -м вбудовуванням маски та вбудовуваннями на рівні кожного пікселя E_{pixel} , обчисленими модулем на рівні пікселів. Скалярний добуток слідується активацією сигмоїда, тобто, $m_i[h, w] = \text{sigmoid}(E_{mask}[:, i]^T \cdot E_{pixel}[:, h, w])$. [16]

Зауважте, що емпірично виявлено, що не варто змушувати передбачення масок бути взаємно виключними одне до одного, використовуючи активацію softmax. Під час навчання втрата $L_{mask-cls}$ поєднує втрату перехресної ентропії класифікації та бінарну втрату маски L_{mask} для кожного передбаченого сегмента. Для спрощення використовується та ж L_{mask} , що й у DETR, тобто лінійна комбінація фокальної втрати та втрати кубика, помножена на гіперпараметри λ_{focal} та λ_{dice} відповідно.

Спочатку представляється проста загальна процедура виводу, яка перетворює результати класифікації масок $\{(p_i, m_i)\}_{i=1}^N$ на формати виходу паноптичної або семантичної сегментації. Потім описується процедура семантичного виводу, спеціально розроблена для семантичної сегментації. Зауважується, що конкретний вибір стратегії виводу значною мірою залежить від метрики оцінки, а не від завдання.

Загальний вивід розділяє зображення на сегменти, призначаючи кожен піксель $[h, w]$ одній з N передбачених пар ймовірності-маски через $\arg \max_{i: c_i \neq \emptyset} p_i(c_i) \cdot m_i[h, w]$. Тут c_i є найімовірнішою міткою класу $c_i = \arg \max_{c \in \{1, \dots, K, \emptyset\}} p_i(c)$ для кожної пари ймовірності-маски i . Інтуїтивно, ця процедура призначає піксель у розташуванні $[h, w]$ парі ймовірності-маски i , лише якщо як ймовірність найімовірнішого класу $p_i(c_i)$, так і ймовірність передбачення маски $m_i[h, w]$ є високими. Пікселі, призначені одній і тій же парі ймовірності-маски i , формують сегмент, де кожен піксель позначений c_i . Для семантичної сегментації сегменти, які мають однакову мітку категорії, об'єднуються; тоді як для завдань сегментації на рівні інстанцій індекс i пари ймовірності-маски допомагає розрізнити різні інстанції одного класу. Нарешті, для зменшення частоти хибнопозитивних результатів у паноптичній сегментації слідує попереднім стратегіям виводу. Зокрема, відфільтровуються прогнози низької впевненості перед виводом та видаляються передбачені сегменти, які мають великі частини своїх бінарних масок ($m_i > 0.5$), затінені іншими прогнозами.

Семантичний вивід спеціально розроблений для семантичної сегментації та виконується через просте матричне множення. Емпірично виявлено, що маргіналізація за парами ймовірність-маска, тобто,

$$\arg \max_{c \in \{1, \dots, K\}} \sum_{i=1}^N p_i(c) \cdot m_i[h, w], [16]$$

дає кращі результати, ніж жорстке призначення кожного пікселя парі ймовірність-маска i , яке використовується у загальній стратегії виводу. Функція

argmax не включає категорію "немає об'єкта" (\emptyset), оскільки стандартна семантична сегментація вимагає, щоб кожен вихідний піксель мав мітку.

Ця стратегія повертає ймовірність класу кожного пікселя $\sum_{i=1}^N p_i(c) \cdot m_i[h, w]$. Однак, спостерігається, що пряме максимізування ймовірності класу кожного пікселя призводить до поганої продуктивності. Гіпотеза полягає в тому, що градієнти рівномірно розподіляються на кожен запит, що ускладнює навчання.

MaskFormer сумісний з будь-якою архітектурою основної мережі. У нашій роботі використовуються стандартні основні мережі на основі згорток ResNet (R50 і R101 з 50 та 101 шарами відповідно) та нещодавно запропоновані основні мережі на основі трансформера Swin-Transformer [25]. Крім того, використовується модель R101c, яка замінює перший шар згортки 7×7 в R101 на 3 послідовні згортки 3×3 , що є популярним у спільноті семантичної сегментації.

Декодер пікселів може бути реалізований за допомогою будь-якого декодера семантичної сегментації. Багато методів класифікації на рівні пікселів використовують модулі, подібні до ASPP або PSP, для збору та розподілу контексту по розташуваннях. Модуль Трансформера звертає увагу на всі характеристики зображення, збираючи глобальну інформацію для генерації класових прогнозів. Ця конфігурація зменшує потребу в модулі на рівні пікселів для важкої агрегації контексту. Тому для MaskFormer розроблений легкий декодер пікселів на основі популярної архітектури FPN.

На основі FPN, вони виконують $2 \times$ збільшення роздільності карти характеристик низької роздільності в декодері та сумують її з проектованою картою характеристик відповідної роздільності з основної мережі; проектування виконується для відповідності розмірів каналів карт характеристик за допомогою шару згортки 1×1 , за яким слідує GroupNorm (GN) [26]. Далі, вони зливають сумовані характеристики з додатковим шаром згортки 3×3 , за яким слідує GN

та активація ReLU. Цей процес повторюється, починаючи з карти характеристик з кроком 32, доки не отримають кінцеву карту характеристик з кроком 4. Нарешті, вони застосовують один шар згортки 1×1 для отримання вбудовувань на рівні кожного пікселя. Усі карти характеристик у декодері пікселів мають розмірність 256 каналів.

Трансформер декодер. Використовується той же дизайн Трансформер декодера, що й у DETR. N вбудовувань запитів ініціалізуються як нульові вектори, і кожному запиту асоціюється навчуване позиційне кодування. За замовчуванням використовуються 6 шарів Трансформер декодера з 100 запитами, і, слідуючи за DETR, вони застосовують ту ж втрату після кожного декодера. У експериментах спостерігається, що MaskFormer є конкурентоспроможним для семантичної сегментації навіть з одним шаром декодера, тоді як для сегментації на рівні інстанцій необхідно кілька шарів, щоб видалити дублікати з кінцевих передбачень.

Багатошаровий перцептрон (MLP) має 2 приховані шари з 256 каналами для передбачення вбудовувань масок E_{mask} , аналогічно до голови коробки у DETR. Обидва вбудовування на рівні пікселів E_{pixel} та масок E_{mask} мають 256 каналів.

Використовується фокальна втрата та втрата кубика для маскової втрати: $L_{mask}(m, m_{gt}) = \lambda_{focal} L_{focal}(m, m_{gt}) + \lambda_{dice} L_{dice}(m, m_{gt})$, і гіперпараметри встановлені на $\lambda_{focal} = 20.0$ та $\lambda_{dice} = 1.0$. Слідуючи за DETR, вага для категорії "немає об'єкта" (\emptyset) у втраті класифікації встановлена на 0.1.

Розділ 2

Моделі семантик сегментації

2.1 Принципи роботи моделей семантик сегментації

Семантична сегментація є завданням групування частин зображень, які належать до одного класу об'єктів. Такий тип алгоритму має кілька випадків використання, таких як виявлення дорожніх знаків, виявлення пухлин, виявлення медичних інструментів під час операцій, сегментація крипт кишечника, класифікація використання та покриття землі. На відміну від цього, несемантична сегментація лише групує пікселі разом на основі загальних характеристик окремих об'єктів. Тому завдання несемантичної сегментації не визначено добре, оскільки багато різних сегментацій можуть бути прийнятними.

Виявлення об'єктів, на відміну від семантичної сегментації, має розрізняти різні інстанції одного і того ж об'єкта. Хоча наявність семантичної сегментації безумовно є великою перевагою при спробі отримати інстанції об'єктів, є кілька проблем: сусідні пікселі одного класу можуть належати до різних інстанцій об'єкта, і регіони, які не з'єднані, можуть належати до однієї інстанції об'єкта. Наприклад, дерево перед автомобілем, яке візуально розділяє автомобіль на дві частини.

Спільнота комп'ютерного зору опублікувала широкий спектр алгоритмів сегментації на сьогоднішній день. Ці алгоритми можна групувати за типом даних, з якими вони працюють, та за типом сегментації, яку вони здатні виробляти. Семантичну сегментацію краще за все розглядати на прикладах, починаючи з UNET [27].

2.1.1 Огляд UNET

Протягом останніх двох років глибокі згорткові мережі перевершили попередні досягнення в багатьох задачах візуального розпізнавання.

Незважаючи на те, що згорткові мережі існували вже довгий час, їхній успіх

був обмежений розміром доступних навчальних наборів даних та розміром самої мережі. Прорив відбувся завдяки навчанню великої мережі з 8 шарами та мільйонами параметрів на наборі даних ImageNet [28] з 1 мільйоном навчальних зображень, проведеному Крижевським та ін. З того часу були навчені ще більші та глибші мережі.

Типове використання згорткових мереж полягає у завданнях класифікації, де вихідними даними для зображення є одна мітка класу. Однак у багатьох візуальних завданнях, особливо в обробці біомедичних зображень, бажаним виходом є локалізація, тобто кожному пікселю має бути призначена мітка класу. Однак наявність тисяч навчальних зображень зазвичай є недосяжною у біомедичних завданнях. Тому Цирсан та ін. навчили мережу у конфігурації "ковзне вікно" для прогнозування мітки класу кожного пікселя, надаючи локальний регіон (патч) навколо цього пікселя.

Архітектура мережі [27] складається з усадочного шляху (ліва частина) та розширювального шляху (права частина). Усадочний шлях слідує типовій архітектурі згорткової мережі. Він складається з повторюваного застосування двох 3×3 згортки (без заповнення), кожна з яких слідується прямолінійною ректифікованою одиницею (ReLU) та операцією максимального об'єднання 2×2 з кроком 2 для зниження розміру зображення. На кожному етапі зменшення розміру подвоюється кількість каналів характеристик. Кожен крок на розширювальному шляху складається з збільшення розміру карти характеристик, за яким слідує 2×2 згортка («згортка збільшення»), яка зменшує кількість каналів характеристик навпіл, конкатенація з відповідно обрізаною картою характеристик з усадочного шляху, та дві 3×3 згортки, кожна з яких слідується ReLU. Обрізка необхідна через втрату крайових пікселів у кожній згортці. На останньому шарі використовується 1×1 згортка для відображення кожного 64-компонентного вектора характеристик на бажану кількість класів. Всього в мережі 23 згорткових шари.

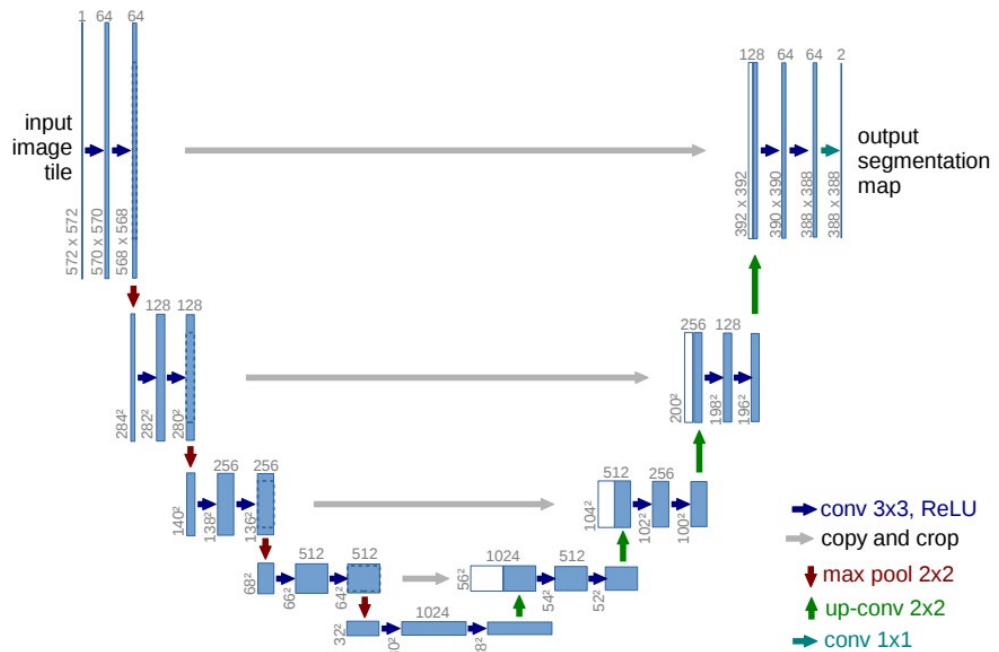


Рисунок 2.1. Архітектура U-net (приклад для 32x32 пікселів у найнижчому дозволі). Кожне синє поле відповідає карті багатоканальних функцій. Кількість каналів вказана у верхній частині вікна. Розмір x-у вказано в нижньому лівому куті поля. Білі поля представляють скопійовані карти об'єктів. Стрілки позначають різні операції. [27]

Для безперервного замощення вихідної карти сегментації важливо вибрати розмір вхідної плити так, щоб усі операції максимального об'єднання 2x2 застосовувалися до шару з парним розміром x та y.

Вхідні зображення та їх відповідні карти сегментації використовуються для навчання мережі за допомогою стохастичного градієнтного спуску. Через згортки без заповнення вихідне зображення менше вхідного на постійну ширину кордону. Щоб мінімізувати надмір і максимально використати пам'ять GPU, віддають перевагу великим вхідним плиткам замість великого розміру пакету, тому розмір пакету зменшують до одного зображення. Відповідно використовують високий момент (0.99), так що велика кількість раніше переглянутих навчальних зразків визначає оновлення на поточному етапі оптимізації.

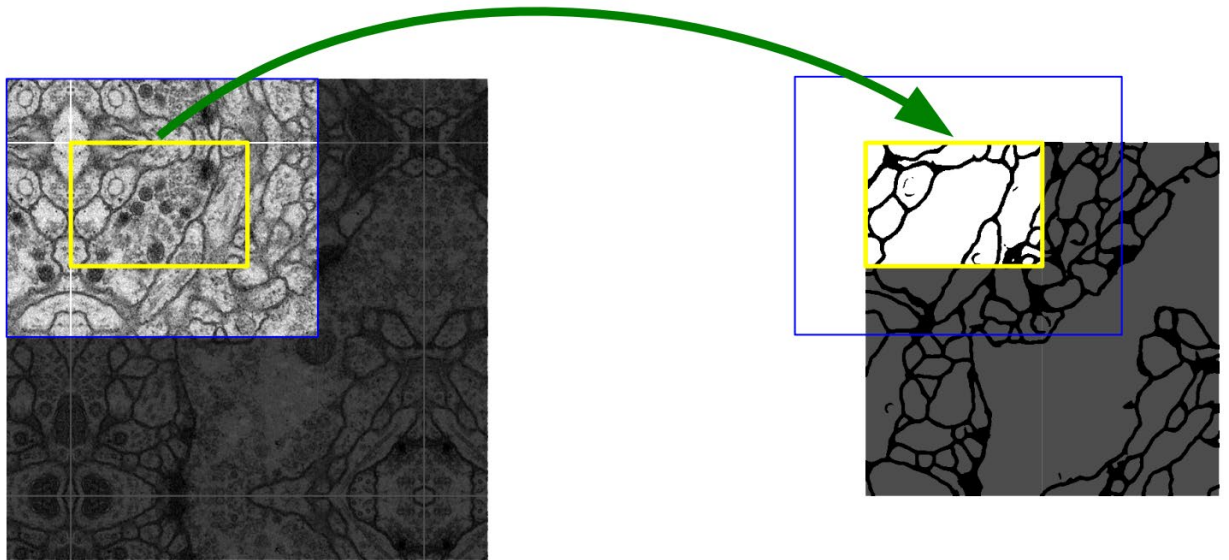


Рисунок 2.2. Стратегія перекривання плиток для безшовної сегментації довільних великих зображень (тут сегментація нейронних структур у стеках ЕМ). Прогнозування сегментації в жовтій області потребує вхідних даних зображення в синій області. Відсутні вхідні дані екстраполюються шляхом віддзеркалення [27]

Функція енергії обчислюється за допомогою піксельної soft-max над останньою картою характеристик у поєднанні з функцією втрати перехресної ентропії. Soft-max визначається як $p_k(x) = \exp(a_k(x)) / \sum_{k'=1}^K \exp(a_{k'}(x))$, де $a_k(x)$ позначає активацію в каналі характеристик k в позиції пікселя $x \in \Omega$ з $\Omega \subset Z^2$. K є кількістю класів, а $p_k(x)$ є наближеною функцією максимуму. Тобто $p_k(x) \approx 1$ для k , який має максимальну активацію $a_k(x)$, і $p_k(x) \approx 0$ для всіх інших k . Потім **перехресна ентропія** карає на кожній позиції відхилення $p_{l(x)}(x)$ від 1, використовуючи

$$E = \sum_{x \in \Omega} w(x) \log(p_{l(x)}(x)) \quad [6]$$

де $l(x)$ позначає істинний клас в позиції x , та $l : \Omega \rightarrow \{1, \dots, K\}$ є істинною міткою кожного пікселя, а $w : \Omega \rightarrow R$ є картою ваг, яку введено для надання деяким пікселям більшої важливості під час навчання.

Карту ваг обчислюють заздалегідь для кожної істинної сегментації, щоб компенсувати різну частоту пікселів певного класу в навчальному наборі даних, а також щоб змусити мережу вчитися малим межам розділення, які вводяться між стикаючимися клітинами.

Межу розділення обчислюють за допомогою морфологічних операцій. Потім карту ваг обчислюють як

$$w(x) = w_c(x) + w_0 \cdot \exp\left(-\frac{(d_1(x)+d_2(x))^2}{2\sigma^2}\right) [6]$$

де $w_c: \Omega \rightarrow R$ є картою ваг для балансування частот класів, $d_1: \Omega \rightarrow R$ позначає відстань до межі найближчої клітини, а $d_2: \Omega \rightarrow R$ - відстань до межі другої найближчої клітини. У експериментах встановлено $w_0 = 10$ і $\sigma \approx 5$ пікселів.

У глибоких мережах з багатьма згортковими шарами та різними шляхами через мережу дуже важливою є хороша ініціалізація ваг. В іншому випадку частини мережі можуть давати надмірні активації, тоді як інші частини ніколи не беруть участь. Ідеально ініційовані ваги повинні бути адаптовані таким чином, щоб кожна карта характеристик у мережі мала приблизно одиничну дисперсію. Для мережі з нашою архітектурою (чергуються шари згортки та ReLU) це можна досягти, вибираючи початкові ваги з гаусівського розподілу зі стандартним відхиленням $\sqrt{2/N}$, де N позначає кількість вхідних вузлів одного нейрона. Наприклад, для 3×3 згортки та 64 каналів характеристик у попередньому шарі $N = 9 \cdot 64 = 576$.

Аугментація даних є важливою для навчання мережі бажаної інваріантності та робастності, коли доступна лише невелика кількість навчальних зразків. У випадку з мікроскопічними зображеннями, перш за все, потрібна інваріантність до зсувів та обертань, а також робастність до деформацій та варіацій сірих значень. Зокрема, випадкові еластичні деформації

навчальних зразків здаються ключовим концептом для навчання мережі сегментації з дуже невеликою кількістю анотованих зображень. Генерація гладких деформацій виконується за допомогою випадкових векторів зміщення на грубій сітці 3 на 3. Зміщення вибираються з гаусівського розподілу зі стандартним відхиленням 10 пікселів. Перепіксельні зміщення потім обчислюються за допомогою бікубічної інтерполяції. Шари відкидання на кінці усадочного шляху виконують подальшу неявну аугментацію даних.

2.1.2 Огляд ViT

У світі обробки природної мови (NLP) архітектура Transformer заслужила статус стандарту, демонструючи видатні досягнення завдяки своїм великим моделям з сотнями мільярдів параметрів. Ці моделі, ефективно навчені на величезних текстових корпусах, показали, що масштабованість і обчислювальна ефективність можуть призвести до небувалого прогресу в NLP. Втім, у світі комп'ютерного зору, згорткові нейронні мережі (CNN) [4] досі вважаються золотим стандартом. Незважаючи на спроби інтегрувати елементи самоуваги в ці мережі, повноцінне застосування архітектури Transformer в комп'ютерному зорі залишалось недосяжним.

У цьому контексті пропонується Vision Transformer (ViT) [3], модель, яка відходить від традиційної залежності від CNN у комп'ютерному зорі. ViT застосовує чисту архітектуру Transformer [2] безпосередньо до послідовностей патчів зображень, обробляючи їх подібно до того, як обробляються слова в тексті. Такий підхід демонструє, що використання CNN не є необхідним для ефективного розпізнавання зображень. Коли модель ViT попередньо навчається на великих датасетах та застосовується до різноманітних завдань класифікації зображень, вона показує вражаючі результати, часто перевершуючи найсучасніші CNN, при цьому вимагаючи значно менше ресурсів для навчання.

У проектуванні моделі слідується за оригінальним Transformer якомога точніше. Перевагою цього навмисно простого підходу є те, що масштабовані

архітектури Transformer для NLP – і їхні ефективні реалізації – можуть бути використані майже без змін.

Стандартний Transformer отримує на вхід одновимірну послідовність токен-вбудовувань. Для обробки двовимірних зображень, зображення $x \in R^{H \times W \times C}$ трансформується у послідовність сплюснених двовимірних патчів $x_p \in R^{N \times (P^2 \cdot C)}$, де (H, W) - розмірність оригінального зображення, C - кількість каналів, (P, P) - розмірність кожного патча зображення, а $N = HW/P^2$ - кількість патчів, яка також виступає як ефективна довжина вхідної послідовності для Transformer. Transformer використовує постійний розмір прихованого вектору D на всіх своїх рівнях, тому патчі сплющуються і відображаються до D вимірів за допомогою навчованої лінійної проєкції.

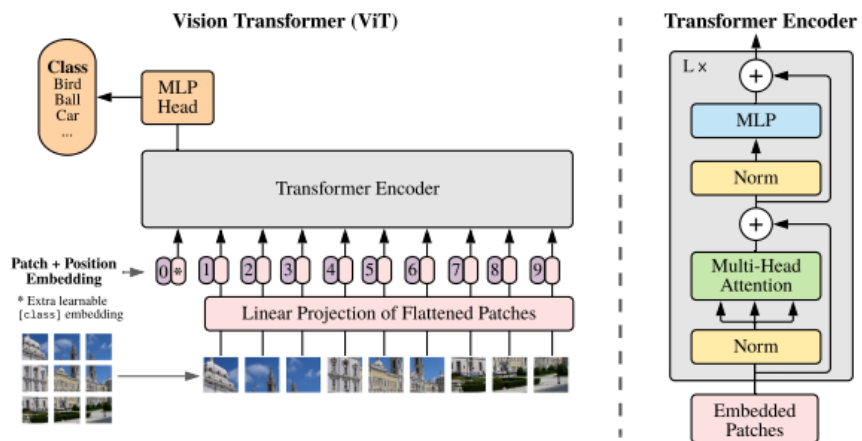


Рисунок 2.3. Огляд моделі. Зображення розбивається на фрагменти фіксованого розміру, лінійно вставляємо кожен із них, додаємо вбудовані позиції та подаємо отриману послідовність векторів у стандартний енкодер Transformer. Щоб виконати класифікацію, використовується стандартний підхід додавання додаткового «маркера класифікації» до послідовності. [3]

Аналогічно до токєну [class] у BERT [29], на початок послідовності вбудованих патчів додається навчований вбудовування ($z_0 = x_{class}$), стан якого на виході кодєра Transformer (z_0^L) слугує як представлення зображення у. Як під

час переднього навчання, так і під час доопрацювання, до z_0^L приєднується головка класифікації. Головка класифікації реалізується за допомогою MLP [30] з одним прихованим шаром на етапі переднього навчання та одним лінійним шаром на етапі доопрацювання.

Позиційні вбудовування додаються до вбудовувань патчів для збереження позиційної інформації. Використовуються стандартні навчані одновимірні позиційні вбудовування, оскільки не спостерігалось значних зростань продуктивності від використання більш розвинених двовимірних позиційних вбудовувань. Результуюча послідовність векторів вбудовувань служить як вхід до кодера.

Кодер Transformer складається з чергування шарів багатоголової самоуваги (MSA) [31] та блоків MLP. LayerNorm (LN) [32] застосовується перед кожним блоком, а зворотні зв'язки - після кожного блоку.

MLP містить два шари з нелінійністю GELU [33].

$$z_0 = [x_{class}; x_1^p E; x_2^p E; \dots; x_N^p E] + E_{pos}, z'_l = \text{MSA}(\text{LN}(z_{l-1})) + z_{l-1},$$

$$z_l = \text{MLP}(\text{LN}(z'_l)) + z'_l, y = \text{LN}(z_0^L)$$

$$E \in R^{(p^2 \cdot C) \times D}, E_{pos} \in R^{(N+1) \times D}$$

$$l = 1 \dots L$$

[3]

Зауважується, що Vision Transformer має значно менше зображення-специфічного індуктивного упередження, ніж CNN. У CNN локальність, двовимірна структура сусідства та еквівалентність перекладу закладені у кожен шар по всій моделі. У ViT лише шари MLP є локальними та перекладно еквівалентними, тоді як шари самоуваги є глобальними. Двовимірна структура

сусідства використовується дуже обмежено: на початку моделі шляхом розрізання зображення на патчі та під час доопрацювання для налаштування позиційних вбудовувань для зображень різної роздільності. Окрім цього, позиційні вбудовування на момент ініціалізації не несуть інформації про 2D позиції патчів, і всі просторові відносини між патчами мають бути навчені з нуля.

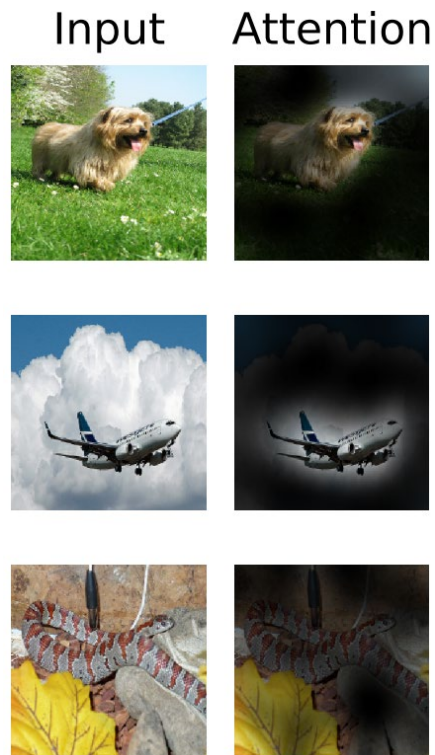


Рисунок 2.4. Типові приклади уваги від вихідного маркера до вхідного простору. [3]

Як альтернатива сирих патчів зображень, вхідна послідовність може бути сформована з карт ознак CNN. У цій гібридній моделі проєкція вбудовування патчів E застосовується до патчів, вилучених з карти ознак CNN. Як спеціальний випадок, патчі можуть мати просторовий розмір 1×1 , що означає, що вхідна послідовність отримується просто шляхом вирівнювання просторових вимірів карти ознак і проєктування до розміру Transformer. Вбудовування для класифікації та позиційні вбудовування додаються як описано вище.

Зазвичай ViT передньо навчається на великих наборах даних і доопрацьовується для (менших) завдань з низхідним потоком. Для цього видаляється попередньо навчений блок прогнозування і приєднуємо нульово ініціалізований шар $D \times K$ зворотного поширення, де K - кількість класів у низхідному потоці. Часто буває корисним доопрацьовувати за більшою роздільною здатністю, ніж під час переднього навчання. Коли зображення більшої роздільної здатності подаються на вхід, розмір патча залишається тим самим, що призводить до більшої ефективної довжини послідовності. Vision Transformer може обробляти довільні довжини послідовностей (обмежені лише пам'яттю), однак передньо навчені позиційні вбудовування вже можуть бути неактуальними. Тому виконується двовимірну інтерполяцію передньо навчених позиційних вбудовувань відповідно до їх розташування в оригінальному зображенні. Зазначимо, що це налаштування роздільної здатності та вилучення патчів - єдині моменти, коли індуктивне упередження щодо 2D структури зображень вручну вводиться у Vision Transformer.

Розділ 3

Використання нейронних мереж для сегментації зображень

3.1 Набір даних

Набір даних для тренування, це зображення дахів з супутника взяті з різних джерел – Bing Maps, Nearmap, Google Maps, але в решті решт для подальших експериментів використовується Nearmap через високу якість зображення, хоча і для кращого застосування потрібно було складний алгоритм отримання зображення з різних тайлів (система впроваджена Google Maps). Дані були взяті за допомогою API для цих сервісів використовуючи базу даних адрес різних приватних будинків у США. Первинний набір даних з цих даних створений для того, щоб нейронна мережа була натренована на дахах усіх будівель на зображеннях. Наступним кроком ця модель використовується щоб знайти дах який знаходиться найближче до центру зображення і сформувати новий набір даних на якому буде видно тільки дах по центру.



Рисунок. 3.1 Розмітка окремих дахів на зображенні за допомогою онлайн сервісу SuperVisely

Цей новий набір даних використовується для розмітки набору даних окремих скатів дахів (рис. 3.2). Кінцева задача полягає в тому, щоб в підсумку знаходити кут нахилу даху використовуючи знімки тих же будівель які були взяті під кутом 45 градусів, для цього і збирались подібні дані.

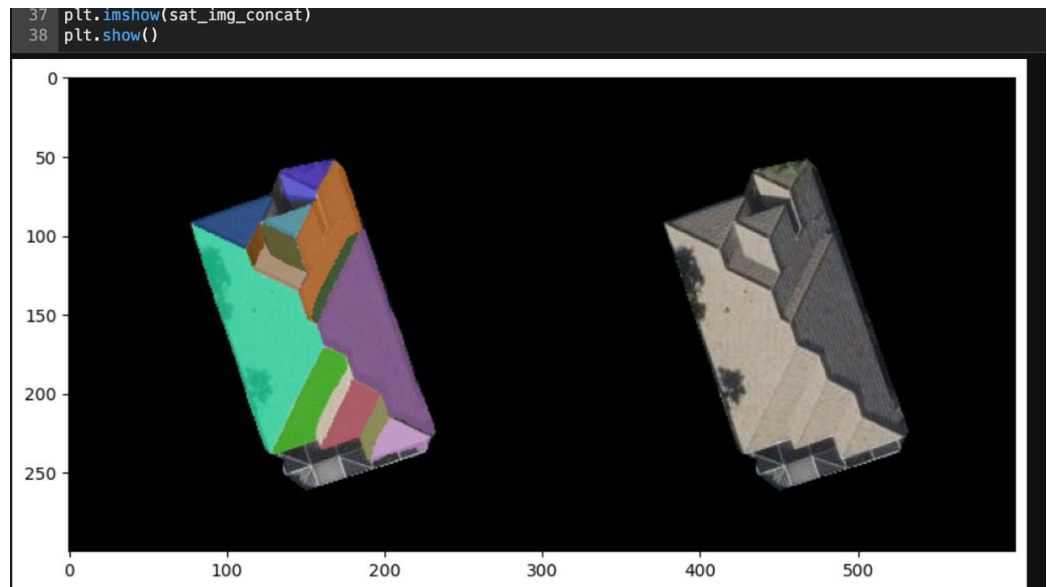


Рисунок. 3.2 Ліворуч - результат роботи моделі

Також, є окремий набір даних, який використовується для семантик сегментації ліній даху (рис. 3.3). Подібний набір даних збирається для написання нового підходу по аналізу характеристик дахів. За допомогою різниці геометрії ліній між зображенням під кутом 90 градусів та 45 градусів теоретично можна рахувати кут нахилу елементів даху.

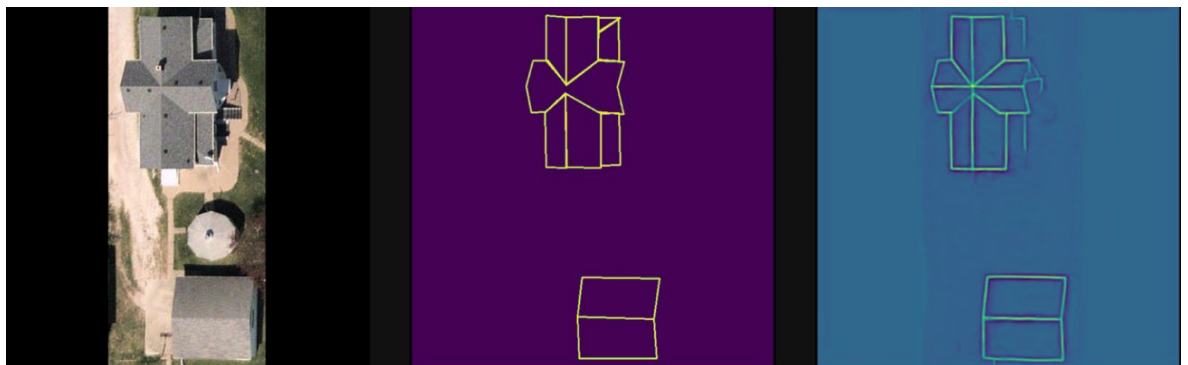


Рисунок. 3.3. Оригінальна фотографія – розмічене зображення – результат обробки моделі

Збирання наборів даних та їх розмітка займає найбільшу кількість часу в даній роботі.

3.2 Використання моделей інстанс сегментації для аналізу дахів

Були протреновані декілька моделей інстанс сегментації, які були оглянуті в [частині 1.2](#). Після тренування моделей з підбором гіперпараметрів по всім показникам найкращою моделлю для нашого датасету є FastInst, що можна побачити на табл. 1

Модель	Час інференції (с)	mAP@0.5	mAP@0.75	mAP (загальний)	IoU	Швидкість навчання (епох/год)	Пам'ять (GB)
Mask R-CNN	0.060	0.45	0.39	0.43	0.57	15	8
PANet	0.055	0.57	0.42	0.52	0.64	14	9
Hybrid Task Cascade	0.050	0.59	0.43	0.54	0.66	16	10
SOLOv2	0.045	0.61	0.44	0.55	0.67	17	9
BlendMask	0.048	0.60	0.45	0.56	0.68	16	8
FastInst	0.035	0.63	0.48	0.59	0.71	21	7
MaskFormer	0.055	0.58	0.46	0.57	0.69	15	9

Таблиця 3.1. Результати порівняння моделей інстанс сегментації

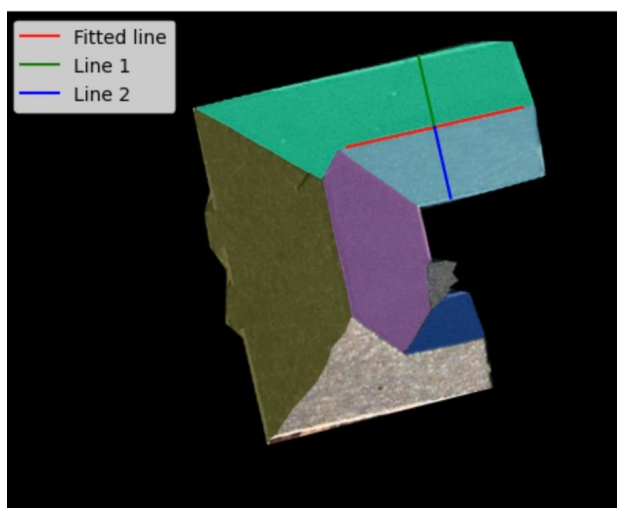


Рисунок. 3.4 . Використання моделі для подальшого визначення двох найближчих скатів даху та її лінії перетину

На зображенні (рис. 3.4) можна побачити, як використовується модель для подальшого визначення двох найближчих скатів даху та її лінії перетину щоб потім знайти 3 точки які за допомогою алгоритмів співвідношення зображень (SIFT, SURF, KAZE, AKAZE, FAST, тощо) знаходити ці ж три точки на зображенні під кутом 45 градусів та наступному знаходженню кута нахилу скатів. Лінія перетину знаходиться за допомогою методу найменших квадратів дані для якого сформовані з координат точок з зони перетину двох знайдених скатів.

Також модель може використовуватись для визначення матеріалів скатів даху, стану, внутрішньої геометрії, фіксації пошкоджень (що є актуальним для регіонів США з екстремальною погодою), тощо.

Експерименти з моделями для співвідношення зображень не дали бажаних результатів. Ось які моделі були протестовані:

- SIFT (Scale-Invariant Feature Transform) [34]: Цей метод виявляє та описує локальні особливості в зображеннях, які інваріантні до масштабування, повороту та частково до зміни перспективи та освітлення.

- SURF (Speeded Up Robust Features) [35]: Подібний до SIFT, але швидший. SURF використовує інтегральні зображення для прискорення обчислення характеристик і робить акцент на швидкості та ефективності.
- ORB (Oriented FAST and Rotated BRIEF) [36]: Цей алгоритм поєднує модифікований FAST детектор ключових точок та BRIEF описник. ORB є ефективним та швидким, при цьому він інваріантний до повороту.
- FAST (Features from Accelerated Segment Test) [37]: Цей алгоритм швидко виявляє кути у зображенні, але не є інваріантним до масштабу та повороту.
- AKAZE (Accelerated-KAZE) [38]: Розвиток KAZE, який використовує швидші методи обчислення для виявлення різноманітних локальних особливостей в зображеннях.
- BRISK (Binary Robust Invariant Scalable Keypoints) [39]: Цей метод створює швидкі двійкові описувачі та виявляє ключові точки, які є інваріантними до масштабування та повороту.

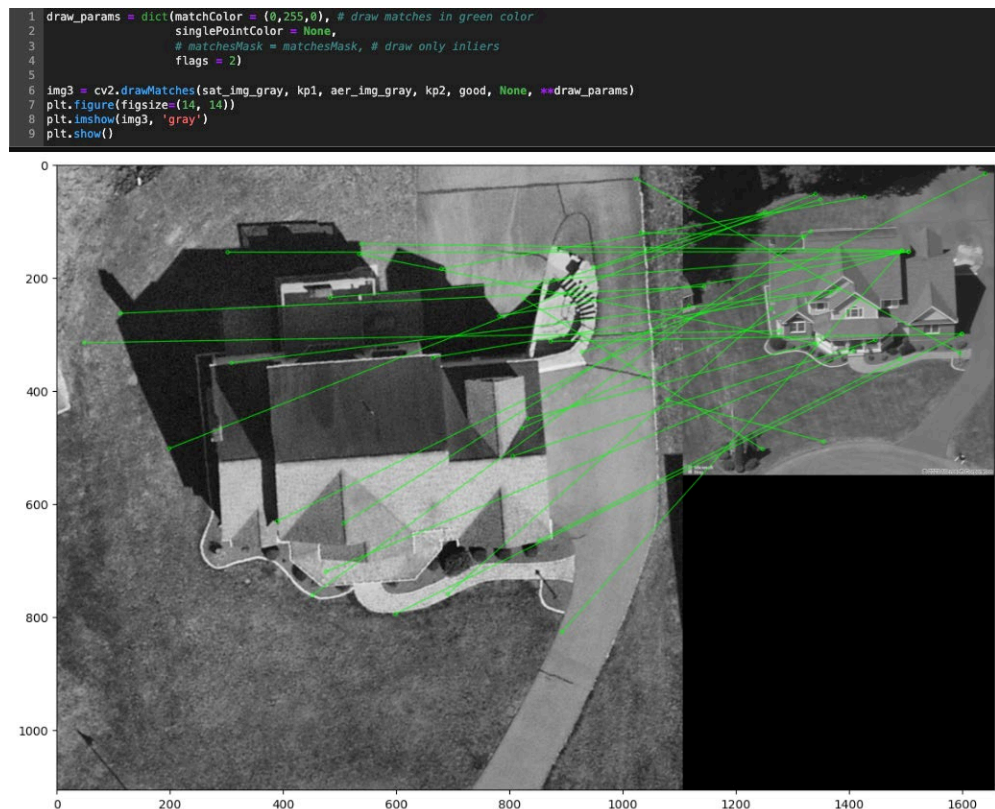


Рисунок 3.5. Приклад використання SIFT

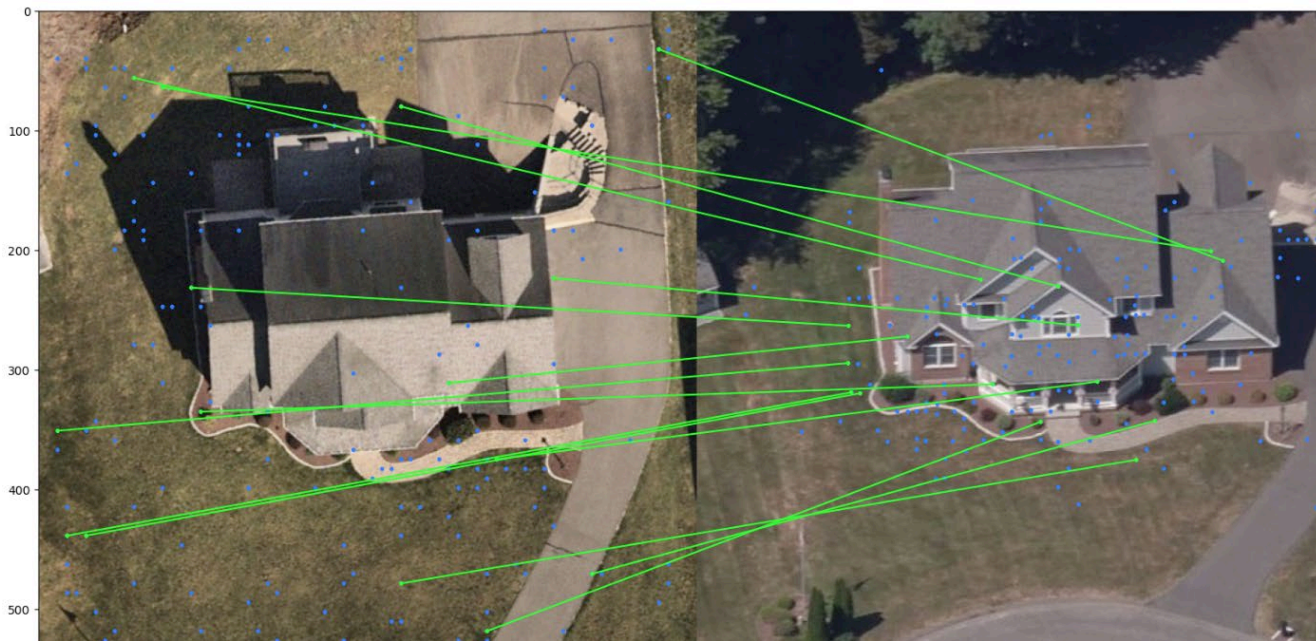


Рисунок 3.6. Приклад використання ORB

Планується проведення експериментів з моделями глибокого навчання для задачі співвідношення зображень, так як вони (як і інші алгоритми машинного навчання) тренуються на специфічних даних і можуть мати набагато вищу точність ніж раніше зазначені моделі. Приклади таких моделей:

- DFM (Deep Feature Matching) [40]: Цей метод використовує навчені характеристики, отримані за допомогою готової глибокої нейромережі (наприклад, архітектури VGG [41]), для досягнення обіцяючої продуктивності. Він використовує густий матчинг найближчих сусідів на кінцевому шарі виходів мережі VGG зображень, які потрібно зіставити. Після початкового вирівнювання та ієрархічного повторення процесу між посиленням та вирівняними зображеннями досягається хороша локалізація та продуктивність зіставлення.
- SuperGlue [42]: Цей метод використовує дескриптори, отримані завдяки використанню оригінальних дескрипторів та розташування ключових точок через багат шаровий перцептрон. SuperGlue навчається робити відповідні зіставлення, використовуючи ці дескриптори.

- **Інтегровані Підходи:** Деякі новітні роботи в літературі відмовилися від класичної послідовності та почали пропонувати поєднання багатьох етапів разом у своїх рішеннях, щоб подолати це обмеження. Такі методи обробляють проблему зіставлення зображень із єдиною архітектурою, безпосередньо оцінюючи геометричне перетворення між двома зображеннями за допомогою регресії. Нещодавно були зроблені спроби знайти відповідності між двома зображеннями, використовуючи глибокі характеристики з єдиною мережею, замість окремого виявлення, опису та зіставлення їх.

Для проведення цього експерименту також треба розмічати та підготовлювати велику кількість даних.

ВИСНОВКИ

Ефективність та Застосування Нейронних Мереж у Сегментації Зображень: Робота підтвердила, що нейронні мережі, зокрема моделі FastInst і Vision Transformer (ViT), є ефективними у задачах сегментації зображень. Ці технології демонструють високу точність і адаптивність, особливо у вирішенні складних завдань, таких як аналіз зображень дахів з різних кутів. Дослідження вперше застосувало глибоке навчання для аналізу характеристик дахів, включаючи визначення кута нахилу та виявлення пошкоджень. Методика включає розробку спеціалізованих алгоритмів для обробки та аналізу зображень, що дозволяє виокремлювати індивідуальні особливості дахів. Незважаючи на успіхи, в дослідженні виявлені певні технічні виклики. Зокрема, важливість великої кількості якісних даних для ефективного навчання моделей та складності інтерпретації отриманих результатів. Робота відкриває нові можливості для подальших досліджень та покращенню алгоритмів обробки зображень та розробки нових моделей для більш точного і швидкого аналізу характеристик дахів. Результати дослідження мають важливе практичне значення, зокрема у сферах урбаністики, нерухомості та страхування. Можливість точного аналізу дахів з різних кутів і при різному освітленні відкриває нові шляхи для оцінки стану будівель, планування ремонтних робіт та оцінки ризиків у страхуванні. Важливою частиною дослідження є інтеграція розроблених методів із іншими технологіями, такими як різні алгоритми машинного навчання для співвідношення зображень. Ця інтеграція має потенціал значно підвищити точність та ефективність аналізу зображень.

Список використаних джерел

- [1] He, Junjie, et al. "FastInst: A Simple Query-Based Model for Real-Time Instance Segmentation." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023.
- [2] Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).
- [3] Dosovitskiy, A., et al. "Transformers for image recognition at scale." arXiv preprint arXiv:2010.11929 (2020).
- [4] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." nature 521.7553 (2015): 436-444.
- [5] Jason Brownlee. A Gentle Introduction to the Rectified Linear Unit (ReLU). URL: <https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/> (дата звернення: 02.09.2023).
- [6] Jason Brownlee. Cross Entropy for Machine Learning. Machine Learning Mastery. URL: <https://machinelearningmastery.com/cross-entropy-for-machine-learning/> (дата звернення: 02.09.2023).
- [7] Long, Jonathan, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.
- [8] He, Kaiming, et al. "Mask r-cnn." Proceedings of the IEEE international conference on computer vision. 2017.
- [9] Girshick, Ross. "Fast r-cnn." Proceedings of the IEEE international conference on computer vision. 2015.
- [10] He, Junjie, et al. "FastInst: A Simple Query-Based Model for Real-Time Instance Segmentation." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2023.

- [11] Wang, Xinlong, et al. "Solov2: Dynamic and fast instance segmentation." *Advances in Neural information processing systems* 33 (2020): 17721-17732.
- [12] Liu, Shu, et al. "Path aggregation network for instance segmentation." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018.
- [13] Chen, Kai, et al. "Hybrid task cascade for instance segmentation." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019.
- [14] Bolya, Daniel, et al. "Yolact: Real-time instance segmentation." *Proceedings of the IEEE/CVF international conference on computer vision*. 2019.
- [15] Chen, Hao, et al. "Blendmask: Top-down meets bottom-up for instance segmentation." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2020.
- [16] Cheng, Bowen, Alex Schwing, and Alexander Kirillov. "Per-pixel classification is not all you need for semantic segmentation." *Advances in Neural Information Processing Systems* 34 (2021): 17864-17875.
- [17] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*. 2014.
- [18] He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
- [19] Chen, Liang-Chieh, et al. "Encoder-decoder with atrous separable convolution for semantic image segmentation." *Proceedings of the European conference on computer vision (ECCV)*. 2018.
- [20] Kirillov, Alexander, et al. "Panoptic feature pyramid networks." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019.
- [21] Tian, Zhi, et al. "Fcos: Fully convolutional one-stage object detection." *Proceedings of the IEEE/CVF international conference on computer vision*. 2019.

- [22] Zhu, X., et al. "Deformable transformers for end-to-end object detection." arXiv preprint arXiv:2010.04159.
- [23] Cordonnier, Jean-Baptiste, Andreas Loukas, and Martin Jaggi. "Multi-head attention: Collaborate instead of concatenate." arXiv preprint arXiv:2006.16362 (2020).
- [24] Moon, Todd K. "The expectation-maximization algorithm." IEEE Signal processing magazine 13.6 (1996): 47-60.
- [25] Liu, Ze, et al. "Swin transformer: Hierarchical vision transformer using shifted windows." Proceedings of the IEEE/CVF international conference on computer vision. 2021.
- [26] Wu, Yuxin, and Kaiming He. "Group normalization." Proceedings of the European conference on computer vision (ECCV). 2018.
- [27] Ronneberger, Olaf, Philipp Fischer, and Thomas Brox. "U-net: Convolutional networks for biomedical image segmentation." Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18. Springer International Publishing, 2015.
- [28] Russakovsky, Olga, et al. "Imagenet large scale visual recognition challenge." International journal of computer vision 115 (2015): 211-252.
- [29] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).
- [30] Popescu, Marius-Constantin, et al. "Multilayer perceptron and neural networks." WSEAS Transactions on Circuits and Systems 8.7 (2009): 579-588.
- [31] Wang, Dongsheng, et al. "Multi-Head Self-Attention with Role-Guided Masks." Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part II 43. Springer International Publishing, 2021.

- [32] Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton. "Layer normalization." arXiv preprint arXiv:1607.06450 (2016).
- [33] Hendrycks, Dan, and Kevin Gimpel. "Gaussian error linear units (gelus)." arXiv preprint arXiv:1606.08415 (2016).
- [34] Lindeberg, Tony. "Scale invariant feature transform." (2012): 10491.
- [35] Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool. "Surf: Speeded up robust features." *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9*. Springer Berlin Heidelberg, 2006.
- [36] Rublee, Ethan, et al. "ORB: An efficient alternative to SIFT or SURF." 2011 International conference on computer vision. Ieee, 2011.
- [37] Viswanathan, Deepak Geetha. "Features from accelerated segment test (fast)." *Proceedings of the 10th workshop on image analysis for multimedia interactive services, London, UK, 2009*.
- [38] P. F. Alcantarilla and T. Solutions, "Fast explicit diffusion for accelerated features in nonlinear scale spaces," *IEEE Trans. Patt. Anal. Mach. Intell*, vol. 34, no. 7, pp. 1281–1298, 2011
- [39] Leutenegger, Stefan, Margarita Chli, and Roland Y. Siegwart. "BRISK: Binary robust invariant scalable keypoints." 2011 International conference on computer vision. Ieee, 2011.
- [40] Efe, Ufuk, Kutalmis Gokalp Ince, and Aydin Alatan. "Dfm: A performance baseline for deep feature matching." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021.
- [41] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).

[42] Sarlin, Paul-Edouard, et al. "Superglue: Learning feature matching with graph neural networks." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020.