Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Сервіс сегментації зображень документів на основі згорткових нейронних мереж

Виконав: студентка IV курсу, групи СН-41
спеціальності 122 Комп’ютерні науки
(шифр і назва спеціальності)

_____ Дольна О.І,
(підпис) (прізвище та ініціали)

Керівник _____ Готович В.А.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ Шимчук Г.В.
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

Рецензент _____ Лещишин Ю.З.
(підпис) (прізвище та ініціали)

Тернопіль - 2025

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«__» _____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

Студенту Дольній Олені Ігорівні
(прізвище, ім'я, по батькові)

1. Тема роботи Сервіс сегментації зображень документів на основі згорткових нейронних мереж
і
Керівник роботи Готович Володимир Анатолійович, к.т.н., доц.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «07» 05 2025 року № 4/7-444

2. Термін подання студентом завершеної роботи 18.06.2025 р.

3. Вихідні дані до роботи наукові літературні джерела

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ
1. Аналіз предметної області.
2. Розробка алгоритмів сегментації
3. Розробка сервісу сегментації
4. Безпека життєдіяльності, основи охорони праці
Висновки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Титулка. 2. Актуальність роботи 3. Мета, задачі дослідження. 4. Функціональні можливості модуля алгоритмів семантичної сегментації. 5. Результати отримання маски сегментації з різними параметрами 6. Метрики для навчених моделей нейронних мереж. 7. Результати сегментації за допомогою різних моделей. 8. Програмні засоби, які використані для розробки. 9. Вимоги до сервісу. 10. Діаграма компонентів сервісу. 11. Параметри запиту. 12. Висновки. Основні результати проведеного дослідження

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці	Сенчишин В.С., доц. каф. МТ		

7. Дата видачі завдання 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	08.05 – 09.05.25	*Виконано*
2.	Підбір джерел про автоматичну сегментацію зображень		
	документів	10.05 – 13.05.25	*Виконано*
3.	Опрацювання джерел про автоматичну сегментацію		
	зображень документів	14.05 – 17.05.25	*Виконано*
4.	Виконання дослідження щодо розробки алгоритмів		
	автоматичної сегментації зображень документів	18.05 – 20.05.25	*Виконано*
5	Розробка програмного коду	21.05 – 23.05.25	*Виконано*
6.	Оформлення розділу «Аналіз предметної області»	22.05 – 24.05.25	*Виконано*
7.	Оформлення розділу «Розробка алгоритмів сегментації»	25.05 – 27.05.25	*Виконано*
8.	Оформлення розділу «Розробка алгоритмів сегментації»	28.05 – 30.05.25	*Виконано*
9.	Виконання завдання до підрозділу «Безпека		
	життєдіяльності, основи хорони праці»	31.05 – 03.06.25	*Виконано*
10.	Оформлення кваліфікаційної роботи	20.05 – 03.06.25	*Виконано*
11.	Нормоконтроль	01.06 – 05.06.25	*Виконано*
12.	Перевірка на плагіат	04.06 – 10.06.25	*Виконано*
13.	Попередній захист кваліфікаційної роботи	11.06 – 18.06.25	*Виконано*
14.	Захист кваліфікаційної роботи	25.06.25	

Студент _____ Дольна О.І.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Готович В.В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Сервіс сегментації зображень документів на основі згорткових нейронних мереж // Дольна Олена Ігорівна // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем та програмної інженерії, кафедра комп'ютерних наук, група СН–41 // Тернопіль, 2025 // С. – 54, рис. – 23, табл. – 2, слайдів – 12, бібліогр. – 52.

Ключові слова: енкодер, згорткова нейронна мережа, машинне навчання, метрика IoU, сегментація зображень

Кваліфікаційна робота присвячена розробці сервісу для сегментації зображень документів із використанням згорткових нейронних мереж.

У першому розділі наведено існуючі алгоритми сегментації зображень, здійснено огляд інструментів для сегментації зображень.

Другий розділ присвячено розробці алгоритмів сегментації зображень. Реалізовано енкодери за замовчуванням, VGG16, ResNet50. Також реалізовано архітектури моделей згорткових нейронних мереж U-Net, SegNet, PSPNet. Описано процес навчання моделей. Наведено особливості отримання масок сегментації. Проведено підрахунок складових метрики IoU. Продемонстровано результати навчання раніше описаних моделей для задачі сегментації зображень документів.

У третьому розділі наведено процес розробки сервісу сегментації. Розглянуто існуючі аналоги з їх перевагами та недоліками. Сформульовано вимоги до програмного продукту. Запропонована архітектура рішення. Наведено особливості технічної реалізації.

У четвертому розділі розглянуто важливі питання безпеки життєдіяльності та основ охорони праці.

ANNOTATION

Service for segmentation of document images based on convolutional neural networks // Dolna Olena // Ternopil Ivan Pul'uj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Science // Ternopil, 2025 // P. - 54, Fig. - 23, Table - 2, Slide - 12, References - 52.

Keywords: encoder, convolutional neural network, machine learning, IoU metrics, image segmentation

The thesis deals with the development of a service for document image segmentation using convolutional neural networks.

The first section presents existing image segmentation algorithms, and provides an overview of image segmentation tools.

The second section is devoted to the development of image segmentation algorithms. The default encoders, VGG16, ResNet50, are implemented. The architectures of convolutional neural network models U-Net, SegNet, PSPNet are also implemented. The process of training models is described. The features of obtaining segmentation masks are presented. The components of the IoU metric are calculated. The results of training previously described models for the document image segmentation task are demonstrated.

The third section presents the process of developing a segmentation service. Existing analogues with their advantages and disadvantages are considered. Requirements for the software product are formulated. The solution architecture is proposed. Features of technical implementation are presented.

The fourth chapter discusses important issues of life safety and the basics of occupational health and safety.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ СКОРОЧЕНЬ І ТЕРМІНІВ

CNN (convolutional neural network) – згорткова нейронна мережа.

IoU (Intersection over Union) – метрика в області комп’ютерного зору, для оцінки продуктивності моделей виявлення об'єктів та сегментації зображень.

ReLU (Rectified Linear Unit) – зрізаний (виипрямлений) лінійний вузол, це функція активації (передавальна функція).

MRZ (machine readable zone) — машинозчитувана зона, два рядки тексту внизу паспорта.

OpenCV (Open Source Computer Vision Library) – бібліотека комп'ютерного зору з відкритим кодом, містить функції та алгоритми комп'ютерного зору, обробки зображень і чисельних алгоритмів загального призначення з відкритим кодом.

VGG16 – нейромережа для виділення ознак зображень.

ЗМІСТ

ВСТУП

Актуальність теми. Останнім часом завдання сегментації зображень стало однією з областей, що найбільш активно вивчаються в області комп'ютерного зору і глибинного навчання. Сегментація зображень включає процес поділу зображення на смислові частини, такі як об'єкти і фони, з метою отримання більш детальної інформації про зміст зображення. Це є фундаментальним завданням для багатьох напрямків, включаючи автоматичну обробку документів, медичну діагностику, автоматичне розпізнавання об'єктів, робототехніку та автономні автомобілі.

Актуальність автоматичної сегментації підтверджується великою кількістю алгоритмів, реалізованих для цієї задачі [1].

Одним з важливих напрямків у сегментації є машинний аналіз зображень документів, який полягає в автоматичному розпізнаванні та витягуванні інформації з зображень документів. За темою машинного аналізу зображень документів існує велика кількість конференцій та робіт [2]. Прикладами таких конференцій є міжнародні конференції з аналізу та розпізнавання документів (ICDAR), з розпізнавання образів (ICPR), за машинним зором (ICMV). Також згідно з даними з сайту дослідницької групи Scimago, можна спостерігати постійне зростання цитувань робіт з конференції ICDAR, що показує стійкий інтерес до цієї теми [3].

Існує множина потенційних напрямків для використання машинних алгоритмів сегментації зображень документів, наприклад:

– розпізнавання документів [4];
– класифікація штампів [5];
– вилучення MRZ [6];
– виявлення підроблених паспортів [7].

У зв'язку з цим актуальним є дослідження алгоритмів для завдання сегментації зображень документів, а також реалізація сервісу з сегментації зображень документів на основі досліджуваних алгоритмів.

Мета роботи – розробка сервісу сегментації зображень документів.

Для досягнення мети виділено ряд завдань:

– вивчення підходів, технологій сегментації зображень;

– розробка алгоритмів сегментації документів;

– визначення точності роботи алгоритмів для задачі сегментації зображень документів;

– проектування архітектури сервісу для сегментації зображень документів,

– програмна розробка сервісу.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Алгоритми сегментації зображень

На рис. 1.1 представлено перелік основних алгоритмів [1], котрі можуть бути використані для автоматичної сегментації зображення у залежності від поставленого завдання.

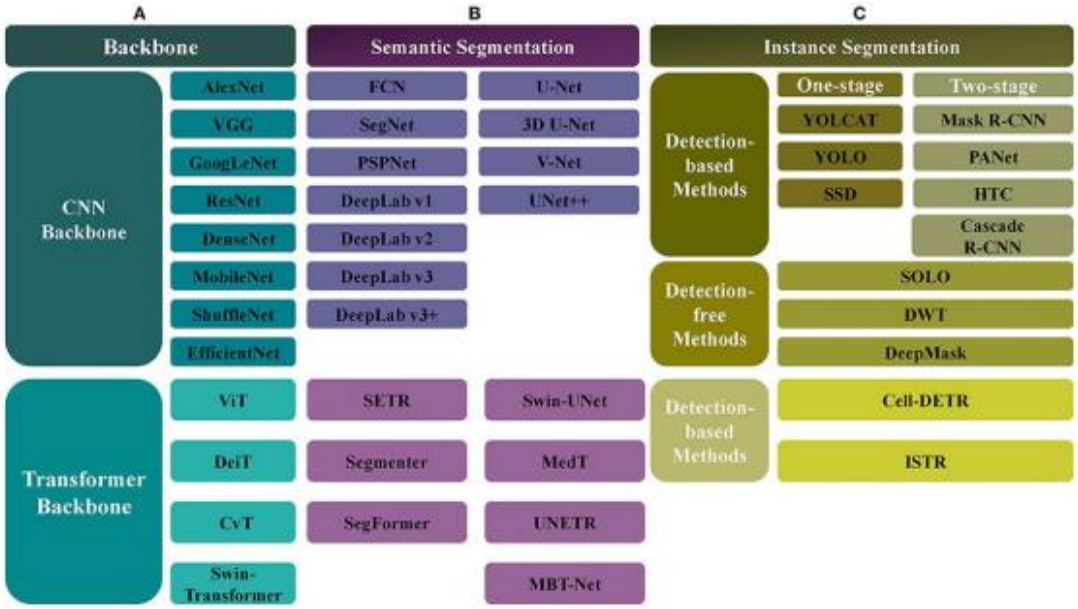

Рисунок 1.1 – Огляд алгоритмів автоматичної сегментації

Однак така широка різноманітність викликає також і складності щодо найбільш ефективного методу для конкретної задачі. Кожен алгоритм має свої особливості, переваги та обмеження, які можуть суттєво впливати на його застосування в конкретному контексті. Факторами, у тому числі, є цілі та вимоги сегментації, тип вхідних даних, обсяг вибірки для навчання, необхідна точність, обчислювальні ресурси.

1.2 Огляд інструментів для сегментації зображень

1.2.1 Мова програмування та бібліотеки

Як мову програмування для розробки було обрано Python. Для цієї мови

програмування реалізовано бібліотеки для машинного навчання TensorFlow та Keras.

TensorFlow - бібліотека з відкритим вихідним кодом для числових обчислень, спроектована для великомасштабного розподіленого навчання. TensorFlow є найпопулярнішим інструментом глибинного навчання. Також дана бібліотека ефективно працює з математичними виразами, включаючи багатовимірні масиви [8].

Keras – популярна бібліотека, обгортка над іншими інструментами глибинного навчання, наприклад TensorFlow. Keras визначає простий шлях швидкого визначення моделей глибинного навчання, абстрагуючись від низько рівневих функцій бібліотек [8].

1.2.2 Структурні елементи архітектур згорткових нейронних мереж

Один з найпопулярніших різновидів нейронних мереж – CNN, що отримала назву від математичної лінійної операції між матрицями, що називається згорткою [9].

CNN мають відмінну продуктивність у завданнях машинного навчання, зокрема при обробці зображень [9]. Структуру CNN можна подати у вигляді набору шарів, що включає у тому числі шари згортки, пулінгу, підвищення розмірності [9].

Вхідний шар. CNN, зазвичай, застосовуються для обробки даних зображення. Кожне зображення можна подати у вигляді матриці значень пікселів.

Крім стандартних вимірів зображення (висота і ширина) також існує третій вимір - колірні канали. Для RGB зображень (червоний, зелений, синій) число каналів дорівнює трьом.

Таким чином, для вхідного RGB зображення розміром, наприклад 256 x 256 пікселів, вхідний шар міститиме тривимірну структуру розміром 256x256x3, що складається з трьох матриць для колірних каналів. Структуру вхідного шару [10] представлено на рис. 1.2.

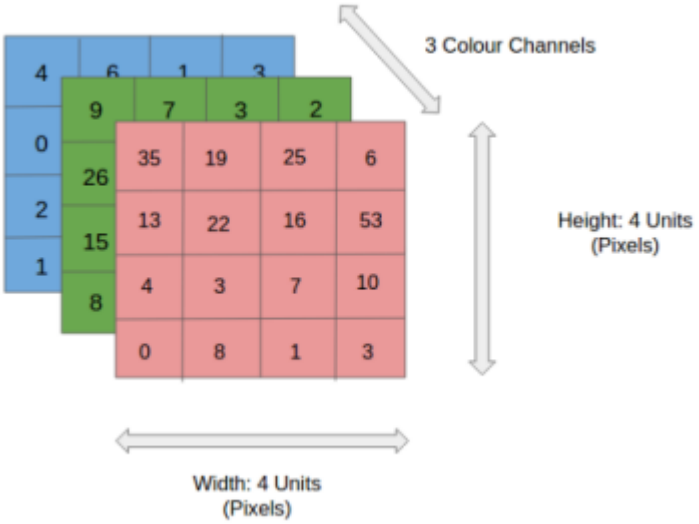

Рисунок 1.2 – Структура вхідного шару

У бібліотеці Keras вхідний шар представлений об'єктом Input [11].

Згортковий (convolution) шар. Шар згортки є основним і найважливішим шаром в CNN. Основна функція даного шару – згортка або множення вхідного шару для створення карти активації зображення [12]. Операція згортки застосовується з використанням ядер, що навчаються (kernel).

Ядра зазвичай невеликі за розмірами, але використовуються по всій глибині вхідних даних. Коли дані потрапляють на згортковий шар, шар згортає кожен фільтр за вхідними даними для створення 2D-карти активації [13].

При ковзанні за вхідними даними для кожного значення в цьому ядрі підраховується скалярний добуток. За допомогою даної операції мережа навчає ядра, які повинні спрацьовувати при знаходженні певної особливості в частині вхідних даних. Дані навчені ядра також називають активаціями [13]. Структура згорткового шару візуально показана на рис. 1.3.

Кожне ядро в результаті має відповідну карту активації, яка буде складена з іншими картами для формування повного обсягу вихідних даних.

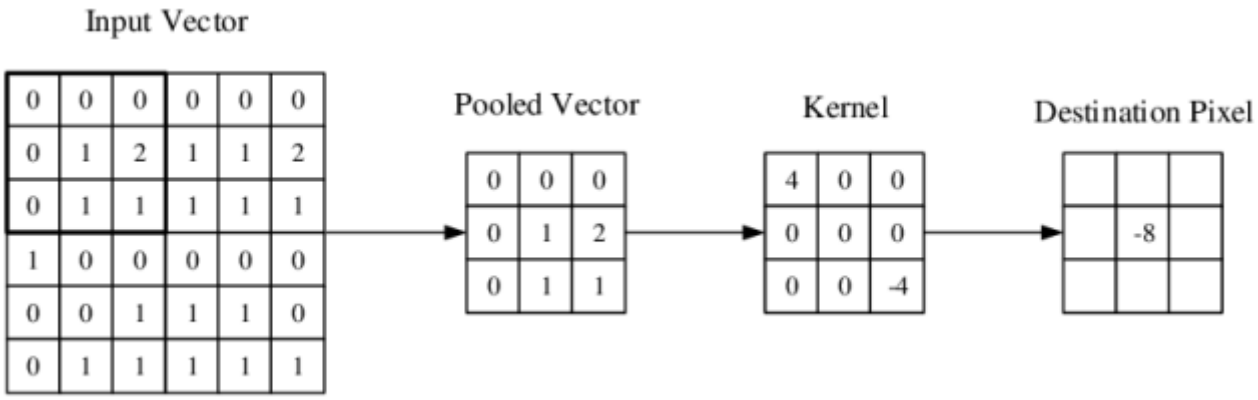

Рисунок 1.3 – Згортковий шар

Для спрощення підрахунків при операції згортки існує можливість вказання меншого розміру рецептивного поля сенсора (у нашому випадку ядра фільтра) [13].

Також існує можливість зменшення складності моделі через оптимізацію вихідних даних за трьома параметрами: глибини (depth, числа вихідних фільтрів), розміру зсуву (stride) та доповнення зображення нулями (zero-padding) [13].

Глибина вихідних даних, створюваних згортковими шарами, може бути встановлена вручну через кількість нейронів усередині шару для області вхідних даних. Зменшення цього параметра може значно мінімізувати загальну кількість нейронів мережі, але також може значно зменшити можливості моделі розпізнавання образів [13].

Також існує можливість вказівки розміру зсуву ядра за вхідними даними. Встановлення кроку більшого ніж 1 дозволить зменшити обсяги карт активації.

Доповнення зображення нулями дозволяє уникнути втрат на краях зображення, оскільки фільтр частіше взаємодіє з кордонними значеннями [9].

У бібліотеці Keras згортковий шар для зображень представлений об'єктом Conv2D , що також дозволяє при ініціалізації вказати параметри kernel_size (розмір ядра фільтра / рецептивного поля сенсора), strides (розмір зсуву по горизонталі та вертикалі), padding = 'same' (доповнення зображення). Доповнення зображення нулями в Keras також представлене у вигляді окремої функції ZeroPadding [15].

Пулінговий (pooling) шар. Використовується для зниження розмірності зображення, таким чином викликаючи скорочення числа параметрів і зменшуючи обчислювальну складність моделі [13]. Вхідні дані розбиваються на блоки із заданою висотою та шириною, після чого для кожного блоку обчислюється деяка функція. Найчастіше використовуються функції максимуму (max pooling) чи середнього (average pooling) [16].

Функція максимуму повертає максимальне значення блоку [17]. Пулінговий шар з такою функцією представлений у бібліотеці Keras як об'єкт MaxPooling2D [18].

Функція середнього повертає середнє значення блоку [17]. Пулінговий шар з такою функцією представлений у бібліотеці Keras як об'єкт AveragePoolmg2D [18].

Робота функцій пулінгового шару візуально представлена на рис. 1.4.

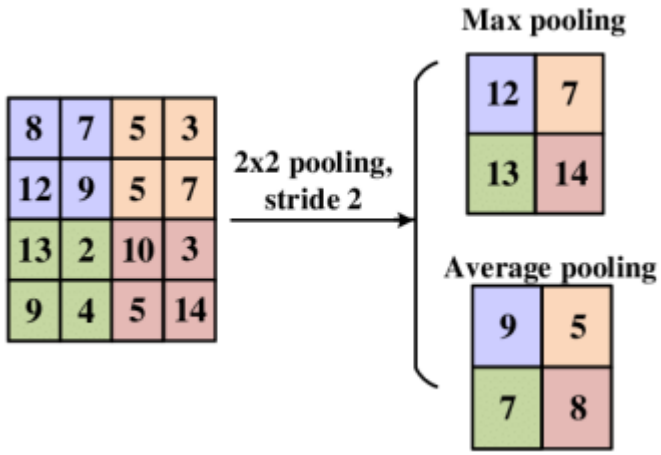

Рисунок 1.4 – Функції Max та Average пулінгового шару [9]

Шар підвищення розмірності (upsampling). Застосовується для збільшення розмірності вхідних даних [20]. Також застосування цього шару дозволяє збільшити розмірність даних до вихідних.

У рамках подальшого дослідження, використовуються такі техніки з підвищення розмірності, як метод найближчого сусіда (Nearest neighbor) і білінійна інтерполяція (Bi-Linear Interpolation) [8].

При використанні методу найближчого сусіда [21] значення вхідних даних

копіюються у всі значення відповідних блоків з певними висотою і шириною . Робота даного методу візуально представлена на рис.1.5.

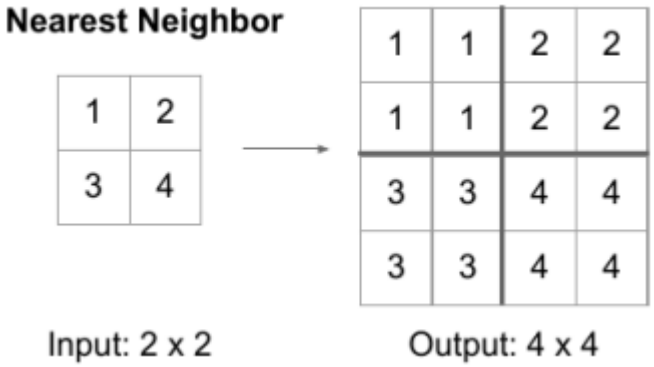

Рисунок 1.5 – Метод найближчого сусіда

При використанні білінійної інтерполяції, для кожного осередку вихідних даних підраховується середньозважене значення чотирьох найближчих значень із вхідних даних, таким чином згладжуючи результат підвищення розмірності. Робота цього методу показана на рис. 1.6.

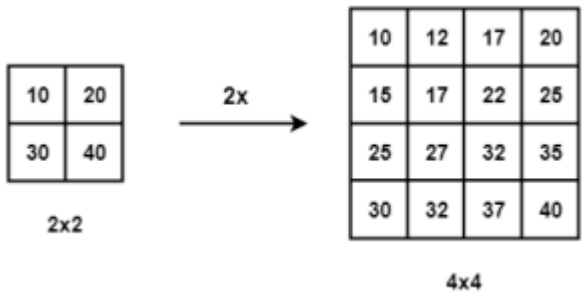

Рисунок 1.6 – Метод білінійної інтерполяції

У бібліотеці Keras шар підвищення розмірності представлений об'єктом UpSampling2D, що при ініціалізації приймає параметри interpolation (наприклад, зі значеннями 'nearest' або 'bilinear'), size (множники висоти та ширини для вихідних даних із вхідних) [22].

Функція активації. Для визначення необхідності активації нейрона у мережі існує можливість застосування функції активації, що визначає спрацьовування для числового результату роботи нейрона [23].

В архітектурах, що досліджуються в рамках даної роботи, використовуються функції ReLU та SoftMax.

ReLU - сама популярна функція активації, серед використовуваних у глибинних нейронних мережах [23], що найчастіше застосовується у прихованих шарах [24]. Функцію ReLU можна формально представити як (1.1):

$$\mathrm{ReLU}(x)=\max(0,x) \tag{1.1}$$

Також функцію ReLU можна подати як графік [25] (рис. 1.7):

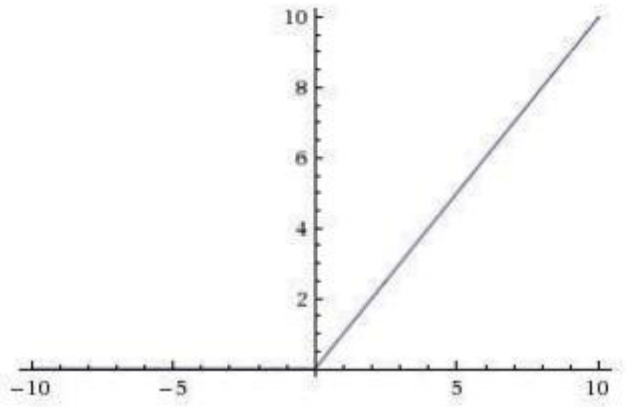

Рисунок 1.7 – Графік функції ReLU

Значною перевагою ReLU є її простота. Завдяки тому, що для негативних значень x функція повертає 0, мережа виходить більш розрідженою та легкою [26].

Softmax - функція активації, застосовувана багатьох вихідних шарів архітектур глибинного навчання, котрі використовують цю функцію [28-29]. Вона застосовується для обчислення розподілу ймовірностей із вектора дійсних чисел. Функція повертає дані, що знаходяться в діапазоні значень від 0 до 1, із сумою ймовірностей, що дорівнює 1 [30]. Softmax можна формально представити як (1.2):

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \quad (1.2)$$

Також функцію Softmax можна подати як графік [31] на рис. 1.8.

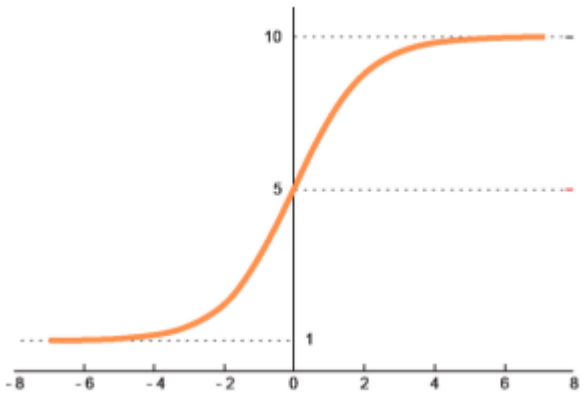

Рисунок 1.8 – Графік функції Softmax

Дана функція, як правило, використовується в багатокласових моделях для повернення ймовірностей кожного класу, при цьому цільовий клас має найбільшу ймовірність [30].

Додати функцію активації в Keras можна за допомогою об'єкта шару Activation (activation_type), або передавши параметром activation при ініціалізації іншого шару [32].

1.2.3 Пакетна нормалізація

Є алгоритмічним методом, котрий робить навчання глибинних нейронних мереж більш швидким та стабільним [33].

Суть даного методу полягає у нормалізації векторів активації із прихованих шарів з використанням першого та другого статистичних моментів (середнього та дисперсії) поточного пакету. Цей крок нормалізації застосовується безпосередньо перед (або одразу після) нелінійної функції [34].

На рис. 1.9 та 1.10 візуально представлена структура нейронної мережі до та після операції пакетної нормалізації [35].

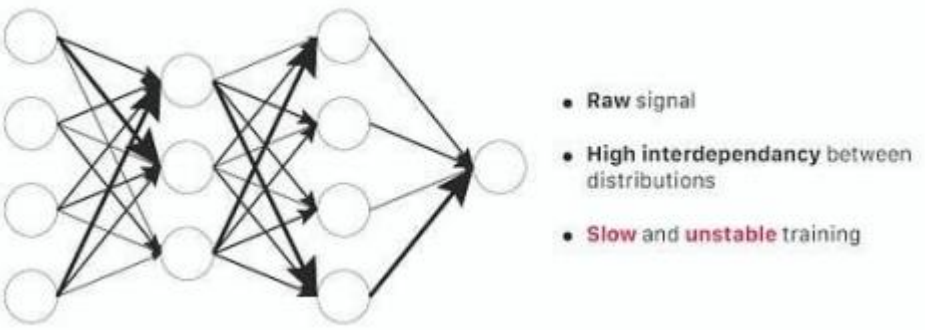

Рисунок 1.9 – Нейронна мережа без пакетної нормалізації

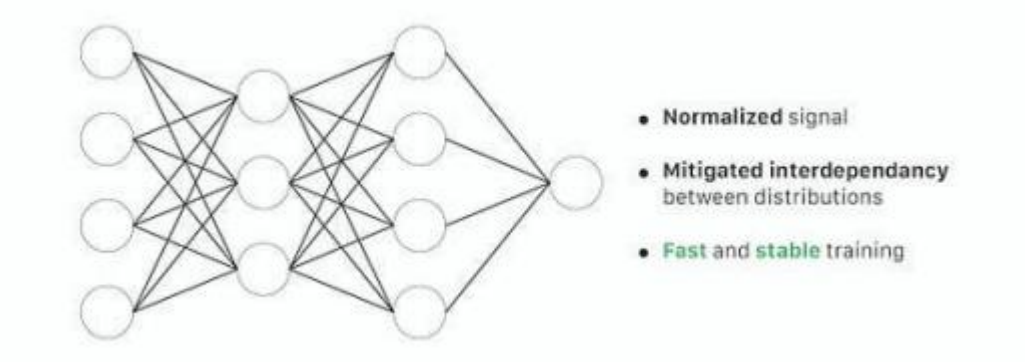

Рисунок 1.10 – Нейронна мережа з пакетною нормалізацією

У Keras даний метод представлений як функція BatchNormalization [35].

РОЗДІЛ 2. РОЗРОБКА АЛГОРИТМІВ СЕГМЕНТАЦІЇ

Для дослідження та подальшого використання алгоритмів семантичної сегментації була реалізований модуль, що включає наступні функціональні можливості:

– навчання моделей з архітектурами U-Net, SegNet, PSPNet з опціональним використанням попередньо навчених моделей з архітектурами VGG16 та ResNet50 як енкодера;

– отримання маски сегментації за зображенням з можливостями використання вхідного зображення як фону і вказання назв класів на зображенні;

– підрахунок середнього та середньозваженого значень метрики IoU, а також значень IoU за класами для навченої моделі за тестовою вибіркою.

2.1 Реалізація енкодерів

Для реалізації різних архітектур моделей з використанням попередньо навчених моделей VGG16 та ResNet50 шари енкодерів були виділені в окремі методи default_encoder(), vgg16(), resnet50().

Перелічені методи приймають на вхід параметри зображення: висоту, ширину та кількість каналів. Результатом роботи є вхідний шар і масив, що складається з вихідних даних 5 шарів для використання декодера.

2.1.1 Енкодер за замовчуванням

Енкодером за замовчуванням, що не використовує попередньо навчені моделі, є шари методу default_encoder(). Для даного енкодера була розроблена спрощена архітектура, що складається з 5 згорткових шарів, що дозволяє спростити навчання вибраної моделі.

Архітектура представлена на рис. 2.1.

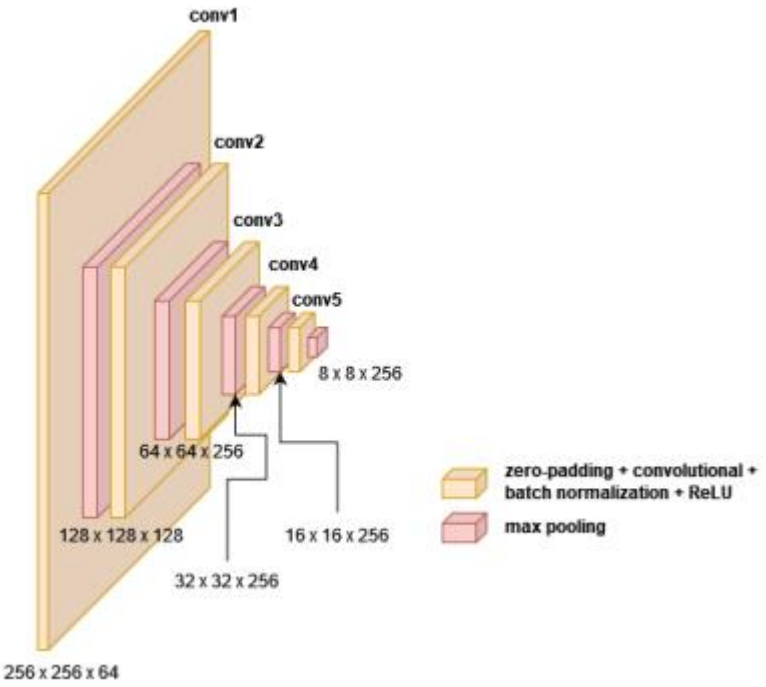

Рисунок 2.1 – Архітектура кодера за замовчуванням

У лістингу 2.1 наведено реалізацію частини архітектури з використанням Keras.

Лістинг 2.1 – Втілення частини архітектури із застосуванням Keras

```
# block 2
pad2 = (ZeroPadding2D((1, 1)))(pool1)
conv2 = (Conv2D (128,    (3, 3), padding = "valid")) (pad2)
norm2 = (BatchNormalization())(conv2)
act2 = (Activation("relu"))(norm2)
pool2 = (MaxPooling2D((2, 2)))(act2)
block_outputs.append(pool2)
```

Вихідними даними default_encoder() є вихідні дані кожного з 5 шарів пулінгу.

2.1.2 VGG16

Як енкодер передбачена можливість використовувати попередньо навчену на датасеті imagenet модель VGG16 з бібліотеки Keras [36]. Показник Top-1 Accuracy для даної моделі в задачі класифікації становить 71.3%, Top-5 Accuracy - 91% [37]. Архітектура моделі VGG16 [38] представлена на рис. 2.2.

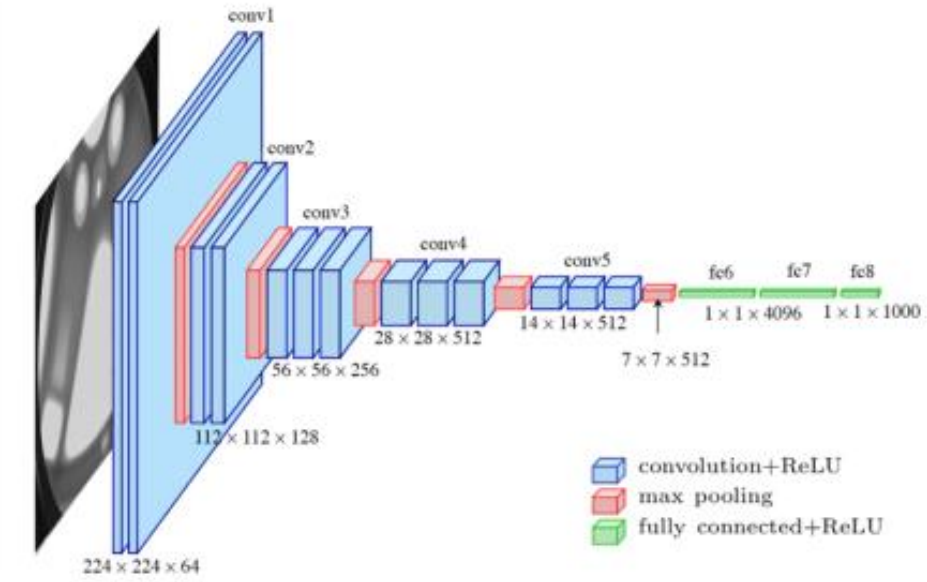

Рисунок 2.2 – Архітектура VGG16

У лістингу 2.2 наведено реалізацію методу vgg16().

Лістинг 2.2 – Втілення методу vgg16()

```
 def vgg16(input_height=224, input_width=224, channels=3):
        vgg = VGG16(input_shape=(input_height, input_width,
channels), include_top=False)
        block_out1 = vgg.get_layer(name="block1_pool").output
        block_out2 = vgg.get_layer(name="block2_pool").output
        block_out3 = vgg.get_layer(name="block3_pool").output
        block_out4 = vgg.get_layer(name="block4_pool").output
        block_out5 = vgg.get_layer(name="block5_pool").output
        return vgg.input, [block_out1, block_out2, block_out3,
block_out4, block_out5]
```

У реалізованому модулі не потрібно використання повнозв'язних шарів. У зв'язку з цим вихідними даними методу vgg16() є вихідні дані кожного з 5 шарів пулінгу.

2.1.3 ResNet50

Як енкодер передбачена можливість використовувати попередньо навчену на датасеті imagenet модель ResNet50 з бібліотеки Keras [39]. Показник Top-1 Accuracy для даної моделі в задачі класифікації становить 74.9%, Top - 5 Accuracy - 92.1% [37].

Архітектура моделі ResNet50 [40] наведена на рис. 2.3.

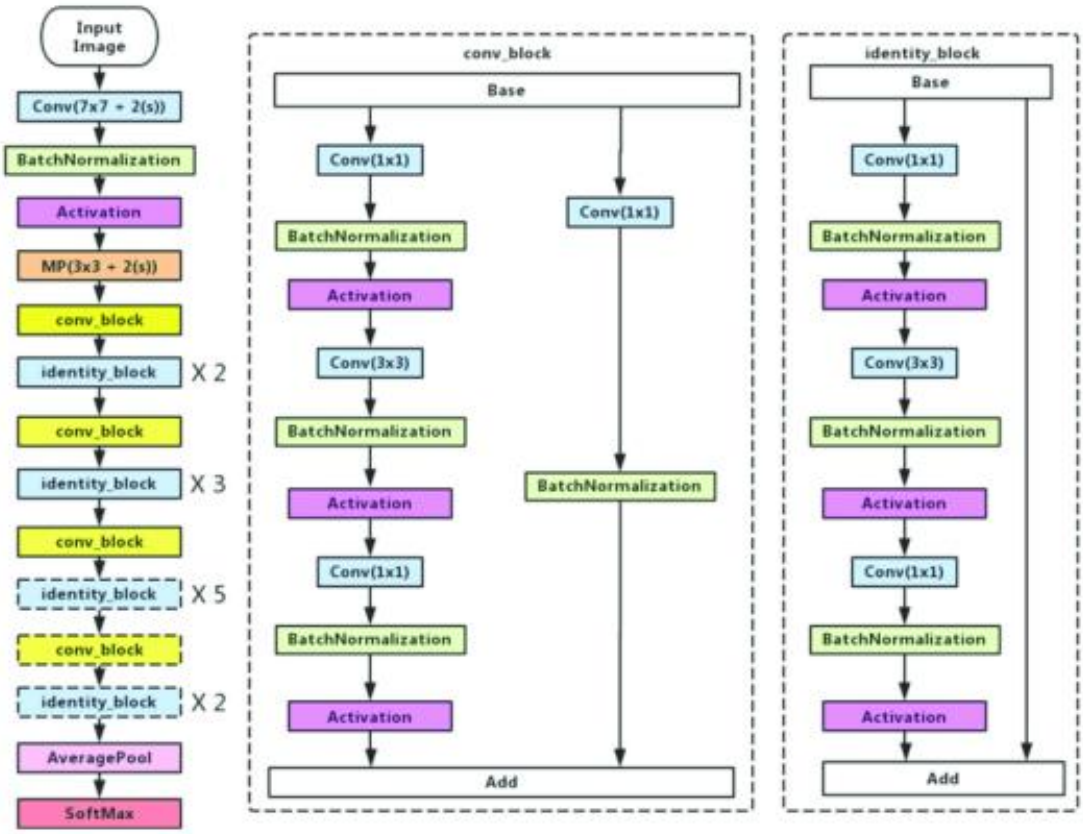

Рисунок 2.3 – Архітектура ResNet50

Особливістю мережі ResNet є використання залишкових з'єднань для навчання нейронних мереж. Вихідними даними блоку шарів буде вважатися конкатенація вихідних даних останнього шару та вхідних даних (у випадку identity_block) або вхідних даних з однією операцією згортки (у випадку conv_b1ock).

Дана особливість дозволяє уникнути погіршення результату при великій кількості шарів через зникаючий градієнт нейронної мережі [41].

У лістингу 2.3 наведено реалізацію методу resnet50().

Реалізований модуль не потребує використання останніх операцій пулінгу та активації SoftMax. Вихідними даними методу resnet50() є вихідні дані шару згортки (початкового), шарів активації після останньої операції конкатенації (Add) для кожного набору conv_block + identity_block * n.

Лістинг 2.3 – Втілення методу resnet50()

```
     def resnet50(input_height=224, input_width=224,
channels=3):
         resnet = ResNet50(
             input_shape=(input_height, input_width, channels),
include_top=False
         )
         block_out1 = resnet.get_layer(name="conv1_conv").output
         block_out2 =
resnet.get_layer(name="conv2_block3_out").outpu
         t block_out3 =
resnet.get_layer(name="conv3_block4_out").outpu
         t block_out4 =
resnet.get_layer(name="conv4_block6_out").outpu
         t block_out5 =
resnet.get_layer(name="conv5_block3_out").output
         return resnet.input, [block_out1, block_out2, block out3,
block out4, block out5]
```

2.2 Реалізація архітектур моделей

Для реалізації архітектур вибраних моделей U-Net, SegNet, PSPNet з використанням попередньо навчених кодерів були розроблені низькорівневі методи _unet(), _segnet(), _pspnet(), що включають додавання шарів декодера та їх зв'язків з результатами роботи шарів кодера.

Основними параметрами даних методів є число класів сегментації; функція кодування; висота, ширина, кількість каналів зображення. Для отримання фінальної моделі в кожному методі викликається утилітарний метод get_segmentation_modelQ, в якому відбувається фінальна активація Softmax, ініціалізація об'єкта Keras Model і додавання методів train(), predict_segmentation(), evaluate_segmentation() в об'єкт моделі.

Також були реалізовані високорівневі методи, що передають відповідні кодери в низькорівневі методи для кожної з архітектур: unet(), vgg16_unet(), resnet50_unet(), segnet(), vgg16_segnet(), resnet50_segnet(), pspnet(), vgg16_pspnet(), resnet50_pspnet().

2.2.1 U-Net

Архітектура реалізованої моделі U-Net [42] представлена на рис.2.4.

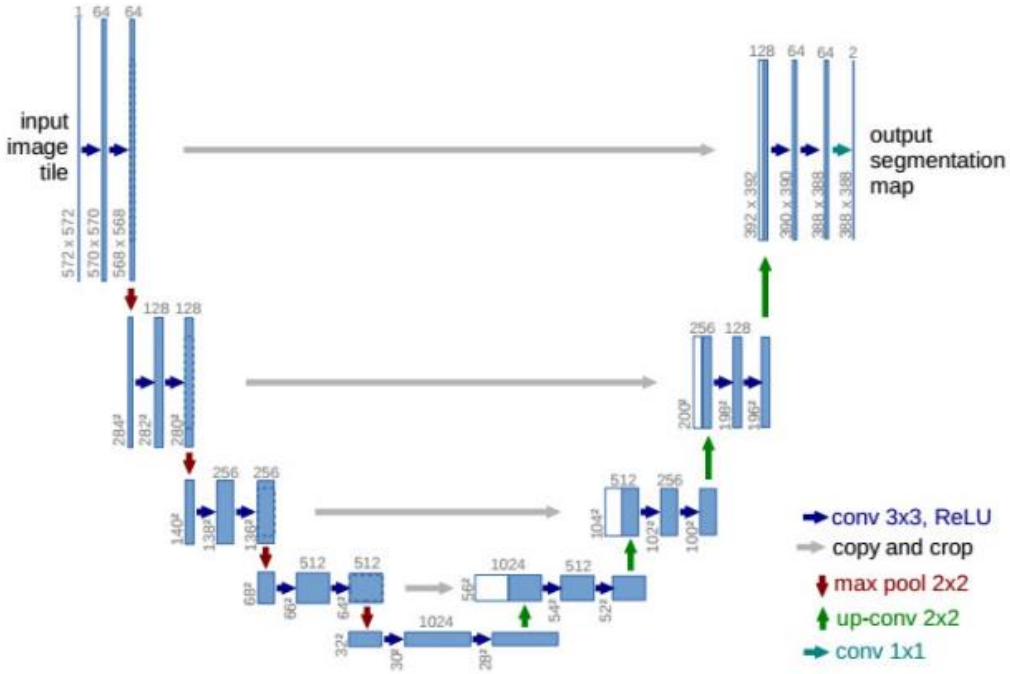

Рисунок 2.4 – Архітектура U-Net

Як шари кодера використовуються раніше описані спрощений кодер за замовчуванням, попередньо навчены моделі VGG16 та ResNet50.

U-Net має кілька відмінних рис:

– використовує з'єднання швидкого доступу (skip connections) між кодером і декодером для зменшення втрат інформації при зниженні розмірності зображення [43];

– має невелике рецептивне поле (розмір ядра), що дає змогу враховувати невеликі деталі зображень.

У лістингу 2.4 наведена частина архітектури U-Net, що включає операцію конкатенації з результатом роботи симетричного шару кодера, таким чином реалізуючи з'єднання швидкого доступу.

Лістинг 2.4 – Частина архітектури U-Net

```
      # block 2
      up2 = (UpSampling2D((2, 2)))(norm1)
      merge2 = concatenate([up2, skip_conn3])
      pad2 = (ZeroPadding2D((1, 1)))(merge2)
      conv2 = (Conv2D (256,    (3, 3), padding = "valid",
activation = "relu")) (pad2)
      norm2 = (BatchNormalization())(conv2)
```

2.2.2 SegNet

Архітектура реалізованої моделі SegNet [28] наведена на рис. 2.5.

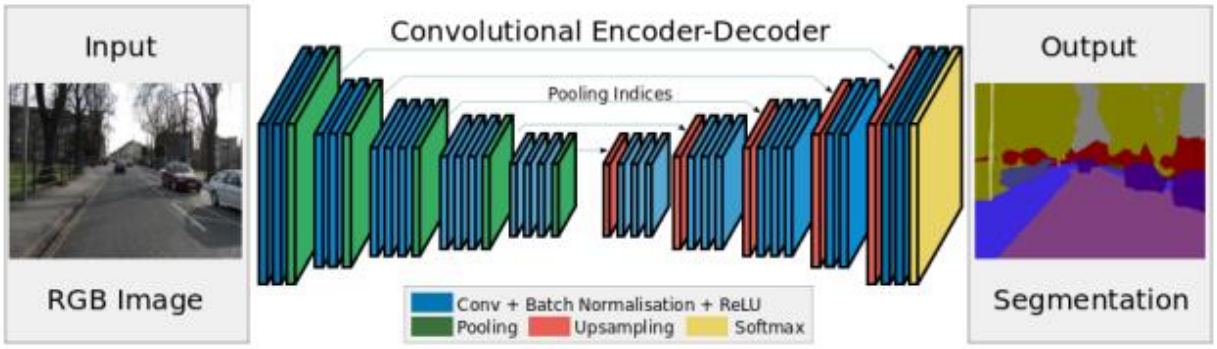

Рисунок 2.5 – Архітектура SegNet

Як шари кодера використовуються раніше описані спрощений кодер за умовчанням, попередньо начені моделі VGG16 та ResNet50.

SegNet має архітектуру схожу на U-Net, але з деякими відмінностями : не передає всю карту ознак для конкатенації, а лише індекси пулінгу; у U-Net немає блоків conv5 та max-pool5, як і в архітектурі VGGNet; але в той же час SegNet використовує всі ваги попередньо навчених згорткових шарів з VGGNet [28].

У лістингу 2.5 наведена частина архітектури SegNet, що містить один із шарів згортки.

Лістинг 2.5 – Частина архітектури SegNet

```
# block 2
up2 = (UpSampling2D((2, 2)))(norm1)
pad2 = (ZeroPadding2D((1, 1)))(up2)
conv2 = (Conv2D (256,    (3, 3), padding = "valid")) (pad2)
norm2 = (BatchNormalization())(conv2)
```

2.2.3 PSPNet

Архітектура реалізованої моделі PSPNet [44] показана на рис. 2.6.

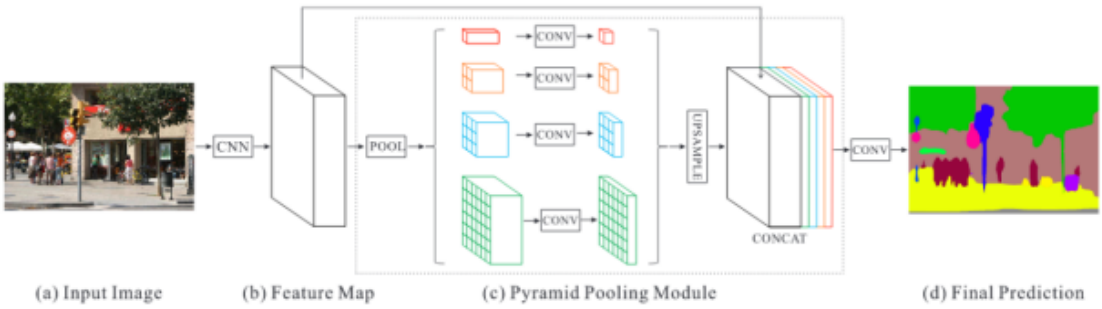

Рисунок 2.6 – Архітектура PSPNet

Як шари кодера використовуються раніше описані спрощений кодер за замовчуванням, попередньо навчені моделі VGG16і ResNet50.

На відміну від U-Net і SegNet, які мають симетричну структуру, як декодер застосовується пірамідальний пулінговий модуль, що містить шари згортки над даними різної розмірності, отриманими з пулінгових шарів. Результатом роботи є згортка з конкатенації вихідних даних шарів згортки (зі збільшеною розмірністю) з даними, отриманими з кодера. У лістингу 2.6 наведена частина реалізації архітектури PSPNet, що відповідає зазначеному розміру даних для отримання з шару пулінгу, приймає на вхід карти ознак, отримані з кодера, потім виконує операції пулінгу, згортки, підвищення розмірності для подальшої кон тенації результатів операцій і підсумкової згортки [44].

Лістинг 2.6 – Частина архітектури PSPNet

```
        def pool_block(encoder_output, pool_factor):
            h = K.int_shape(encoder_output)[1]
            w = K.int_shape(encoder_output)[2]
            pool_size = strides = [
                int(np.round(float(h) / pool_factor)),
                int(np.round(float(w) / pool_factor)),
            ]
            pool = AveragePooling2D(pool_size, strides=strides,
padding="same")(encoder_output)
            conv = Conv2D (512, (1, 1), padding = "same",
use_bias = False (pool)
            norm = BatchNormalization()(conv)
            act = Activation("relu")(norm)
            up = UpSampling2D(strides,
interpolation="bilinear")(act)
            return up
```

2.3 Навчання моделей

Для навчання моделей з різними архітектурами було реалізовано функцію train().

Ця функція приймає на вхід такі параметри:

– model – одна з розроблених моделей як об'єкт keras.Model. Також передбачена можливість виклику методу train() на об'єкті моделі;

– train_images – шлях до вибірки зображень для навчання моделі;

– train_annotations – шлях до вибірки масок сегментації зображень для навчання моделі;

– verify_dataset - необхідність перевірки датасета (приймає True чи False, за умовчанням True). При вказівці True перед навчанням буде проведено перевірку датасетів на відповідність розмірів вихідних та розмічених зображень, відповідність кількості класів, переданих раніше в модель;

– checkpoints_path - шлях (директорія та префікс назви файлу) для збереження конфігурації та чекпоінту (стану) моделі після завершення епохи навчання. Останній чекпоінт є результуючою навченою моделлю. Значення параметра за замовчуванням - None (збереження чекпоінтів не проводиться);

– epochs - кількість епох навчання (за замовчуванням 5), передається безпосередньо у функцію навчання моделі Model.fit() [45];

– steps_per_epoch - число кроків для епохи (ітерацій за датасетом, за замовчуванням 512), після яких епоха вважається завершеною, передається безпосередньо у функцію навчання моделі Model.fit() [45].

Даний метод повертає результат роботи функції ModeLfit(), а саме об'єкт History, що зберігає значення втрат і метрик [45].

Як оптимізатор був обраний алгоритм Adam (Adaptive moment estimation), що комбінує в собі переваги алгоритмів AdaGrad і RMSProp, а також придатний для великих обсягів даних [46].

Як функцію втрат було обрано categorical cross entropy loss, різновид cross entropy loss (одна з найбільш широко використовуваних функцій втрат у глибинному навчанні, що застосовується в тому числі в архітектах U -Net і

SegNet), призначена для багатокласових завдань [47].

Крім цього, при початковій конфігурації моделі за допомогою функції Model.compile () була вказана метрика Accuracy для оцінки навчання моделі [45].

Як параметри вхідних і цільових даних функції Model.fit() використовується генератор нескінченно повторюваного датасета, таким чином число кроків для епохи відповідає значенню steps_per_epoch при будь-якому розмірі датасета [45].

У лістингу 2.7 наведено приклад використання функції навчання системи.

Лістинг 2.7 – Приклад застосування функції навчання системи

```
model = vgg16_pspnet(n_classes=4)
model.train(
    train_images="doc_dataset/img_train/",
    train_annotations="doc_dataset/anno_train/",
    checkpoints_path="./tmp/pspnet",
    epochs=5,
    steps_per_epoch=512, )
```

2.4 Отримання масок сегментації

Для одержання масок сегментації для переданого зображення з використанням навченої моделі було реалізовано функцію predict_segmentation().

Ця функція приймає на вхід один із дванадцяти описаних нижче параметрів:

– model – одна з розроблених моделей як об'єкт keras.Model. Також передбачено можливість виклику функції predict_segmentation() на об'єкті моделі;

– checkpoints_path - шлях (директорія та префікс назви файлу) для отримання чекпоінту (стану) моделі. Для використання як модель вивантажується чекпоінт останньої епохи переданим шляхом. Якщо передано параметр model, дані чекпоінта не вивантажуються;

– inp - вхідне зображення, може бути як шлях до файлу, або зображення як масиву numpy.ndarray, наприклад отриманого за допомогою бібліотеки

OpenCV;

– out_fname – шлях для збереження вихідного файлу – маски сегментації;

– overlay_img – необхідність відображення оригінального зображення на фоні (приймає True або False, за замовчуванням False);

– show_legends – необхідність відображення легенди (приймає True або False, за замовчуванням False). При вказівці True відображає назви класів із параметра class_names (повинен бути обов'язково вказаний) та відповідних кольорів;

– class_names - назви класів сегментації, значення параметра має бути передано за вказівкою show_legends = True;

– colors – перевизначені кольори класів для відображення;

– prediction_width - ширина результуючого зображення (при вказівці результат сегментації буде інтерпольовано до вказаної ширини);

– prediction_height – висота результуючого зображення (при вказівці результат сегментації буде інтерпольований до зазначеної висоти);

– all_visualizations - необхідність візуалізації всіх варіацій зображень (з наявністю/відсутністю підписів та фонового зображення). Приймає True / False;

– out_fnames – шляхи для збереження вихідних файлів – варіацій маски сегментації. Використовується при вказівці параметра all_visualizations, що дорівнює True. Число файлів має відповідати числу варіацій, а саме 4.

Даний метод повертає маску сегментації як масив з вимірюваннями, котрі дорівнюють висоті, ширині та кількості каналів зображення.

У лістингу 2.8 наведено приклад використання функції отримання маски сегментації.

Лістинг 2.8 – Приклад застосування функції одержання маски сегментації

```
out = predict_segmentation(
    checkpoints_path=
"models\unet\vgg16_unet\vgg16_unet",
    inp="doc_dataset/img_test/3832349.jpg",
    out_fname="./tmp/out.png",
    overlay_img=True,
    show_legends=True,
    class_names=["Background", "Passport", "Photo", "MRZ"],
    prediction_width=500,
    prediction_height=500,
    colors=[(0, 0, 0), (255, 0, 0), (0, 255, 0), (0, 0,255)]
)
```

Приклади результатів роботи функції отримання масок сегментації показані на рис. 2.7.

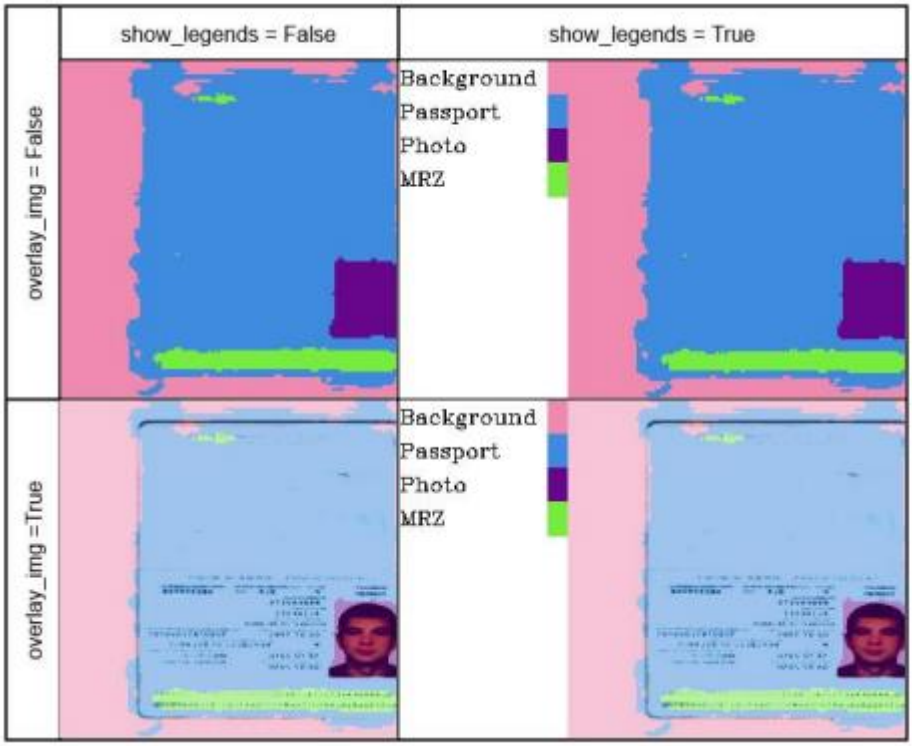

Рисунок 2.7 – Результати роботи функції отримання маски сегментації з різними параметрами

2.5 Підрахунок метрик

Метрика точності (accuracy) є недостатньо якісною в процесі перевіряння

сегментації зображень з великими областями одного класу. За таких умов метрика показуватиме точність більше 30% за наявності тільки апріорного знання [48].

Проблему можна вирішити, у тому числі, використовуючи метрики середнього та середньозваженого IoU [48]. Також зручною метрикою є класовий IoU, що дозволяє оцінити точність роботи моделі за окремими класами.

Таким чином було реалізовано функцію evaluate_segmentation для підрахунку наступних метрик:

– середнє (mean) значення IoU;

– середньозважене (frequency-weighted) значення IoU;

– значення IoU за класами.

Ця функція приймає на вхід такі параметри:

– model – одна з розроблених моделей як об'єкт keras.Model. Також передбачено можливість виклику функції predict_segmentation() на об'єкті моделі;

– checkpoints_path - шлях (директорія та префікс назви файлу) для отримання чекпоінту (стану) моделі. Для використання як модель вивантажується чекпоінт останньої епохи переданим шляхом. Якщо передано параметр model, дані чекпоинта не вивантажуються;

– inp_images – масив із зображеннями для тестової вибірки, котрі можуть бути шляхом до файлу, або зображення як масив numpy.ndarray, наприклад, отриманого за допомогою бібліотеки OpenCV;

– annotations - масив масок сегментації зображень для тестової вибірки, які можуть бути як шлях до файлу, або зображення як масив numpy.ndarray, наприклад отриманого за допомогою бібліотеки OpenCV;

– inp_images_dir - шлях до директорії із зображеннями для тестової вибірки. Використовується, якщо не вказано значення параметра inp_images:

– annotations_dir – шлях до директорії з масками сегментації для вхідних зображень. Використовується, якщо не вказано значення параметра annotations.

Ця функція повертає словник зі значеннями середнього (mean_IU) та

середньозваженого (frequency_weighted_IU) значень метрики IoU, а також значень IoU за класами для вивченої за тестовою вибіркою моделі (class_wise_IU).

У лістингу 2.9 наведено приклад використання функції підрахунку метрик.

Лістинг 2.9 – Приклад застосування функції обчислення метрик

```
metrics = evaluate_segmentation (
    checkpoints_path =
"models\unet\vgg16_unet\vgg16_unet",
    inp_images_dir="doc_dataset/img_test",
    annotations_dir="doc_dataset/anno_test",
)


{
  'frequency_weighted_IU': 0.6733870444911572,
  'mean_IU': 0.5223808806612247,
  'class_wise_IU': array([0.73808964, 0.71159794, 0.371875,
0.26796093])
}
```

2.6 Результати навчання моделей для завдання сегментації зображень документів

Для навчання моделей був використаний розмічений датасет із 200 зображень паспортів та віз з різними розташуваннями документа у кадрі та якістю зйомки.

Даний датасет був випадковим чином розбитий на навчальну вибірку з 160 зображень і на тестову, що включає відповідно 40 зображень. Зображення було розмічено за допомогою 4 класів: фон, документ, фото, MRZ.

Результати навчання оцінили за допомогою метрик, отриманих з реалізованої функції підрахунку. Дані представлені у табл. 1.1.

Таблиця 1.1 – Метрики для навчених моделей нейронних мереж

Модель	Середнє IoU	Середньо - зважен. IoU	IoU Фон	IoU Документ	IoU Фото	IoU MRZ
U-Net	0.585	0.635	0.655	0.648	0.428	0.607
VGG16 + U-Net	0.681	0.771	0.812	0.789	0.522	0.594
ResNet50 + U-Net	0.626	0.532	0.594	0.694	0.313	0.529
SegNet	0.468	0.424	0.309	0.591	0.402	0.395
VGG16 + SegNet	0.643	0.551	0.681	0.667	0.416	0.440
ResNet50 + SegNet	0.677	0.469	0.779	0.719	0.192	0.186
PSPNet	0.697	0.508	0.782	0.744	0.428	0.078
VGG16 + PSPNet	0.671	0.542	0.744	0.691	0.34	0.391
ResNet50 + PSPNet	0.575	0.488	0.51	0.662	0.388	0.391

Кращі метрики для тестової вибірки показала зв'язка VGG16 та U-Net. Даний результат є очікуваним через особливості роботи CNN U-Net. Архітектура U-Net була розроблена і часто застосовується в медичних цілях, де найважливіше розпізнавання невеликих об'єктів [42]. При дослідженні даної архітектури було виявлено, що дана нейромережа добре справляється з невеликими деталями на зображенні завдяки з'єднанням швидкого доступу, що переносять ваги із симетричних шарів, дозволяючи не втрачати корисні дані. Також у даній архітектурі передбачається використання невеликого ядра, що також має позитивий вплив на результати опрацювання. Ця особливість нейронної мережі особливо добре помітна на метриці IoU для MRZ, оскільки інші нейромережі не зовсім точно працюють з такими маленькими деталями як текст на таких документах, що представлено на рис. 2.8.

Рисунок 2.8 – Результати сегментації за допомогою різних моделей

В цілому моделі показали хороші результати при визначенні фону і документа на зображенні. Зокрема хороші метрики на всій вибірці показала модель PSPNet без попереднього навчання. Цей результат також пов'язаний з особливістю архітектури. Завдяки даним різного розміру, що використовуються для згортки, дана нейромережа добре працює при визначенні великих об'єктів. PSPNet була розроблена і застосовується при сегментації об'єктів на зображеннях всередині та поза приміщеннями [44], для чого добре підходять перераховані вище особливості архітектури.

Також варто зазначити, що загалом використання попередньо навчених моделей несе позитивний вплив на результати роботи CNN, покращуючи розпізнавання сегментів.

РОЗДІЛ 3. РОЗРОБКА СЕРВІСУ СЕГМЕНТАЦІЇ

3.1 Опис завдання

Необхідно реалізувати сервіс, що приймає на вхід зображення документа і повертає маски сегментації даного зображення, використовуючи моделі CNN, навчені для сегментації зображень документів.

3.2 Огляд аналогів

На даний момент наявні деякі комерційні рішення для розпізнавання документів.

Основним їх недоліком є те, що дані сервіси призначені для конкретної мети - розпізнавання документів, але крім цього завдання існують і безліч інших, наприклад класифікація штампів [5], вилучення MRZ [6], виявлення підроблених паспортів [7].

3.2.1 Passport Recognition Service by Evergreen

Даний сервіс являє собою просте хмарне API, що приймає на вхід фотографію, посилання на фотографію або фотографію у форматі Base64. Відповіддю на запит є JSON із розпізнаними значеннями полів паспорта [49]. Схема роботи сервісу представлена на рис. 3.1.

Згідно з даними з сайту постачальників даного рішення, рівень розпізнавання понад 96%. Для розпізнавання використовується нейромережа, навчена на великій кількості зображень паспортів громадян різних країн. Фотографії на паспортах у навчальній вибірці представляють людей із різним віком, статтю, національністю, расою, довжиною волосся тощо [49].

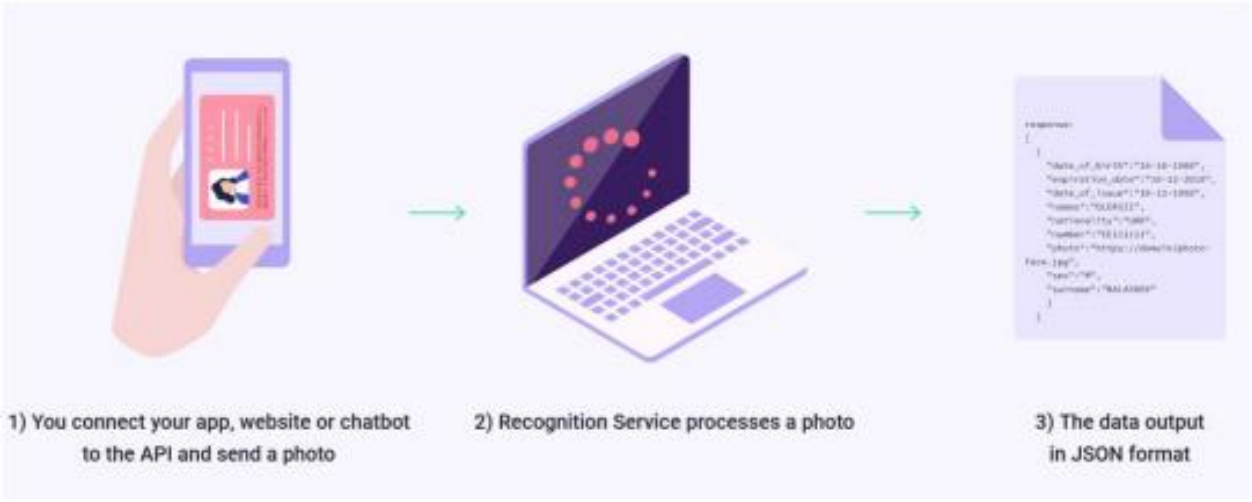

Рисунок 3.1 – Схема роботи Passport Recognition Service by Evergreen

Даний сервіс не підтримує автономну роботу, працюючи тільки через сервер самого сервісу, що призводить до додаткових ризиків у плані безпеки і доступності при недостатньо хорошому з'єднанні.

3.2.2 Smart ID Engine від Smart Engines

Є мультиплатформним автономним сервісом з розпізнавання документів, що засвідчують особу, документів на право власності, віз та інших документів, що використовуються для ідентифікації та контролю доступу, у відеопотоці, на фотографії або зображенні. Також заявляється підтримка розпізнавання широкого числа документів для 193 країн для 101 мови.

Відповідно до специфікації, дане рішення має такі особливості:

– вся обробка даних ведеться в локальній оперативній пам'яті, без використання інтернет-з'єднання та без зберігання даних на сторонніх ресурсах;

– розпізнавання невимогливе до обчислювальних ресурсів, що дозволяє запускати його навіть на слабких пристроях;

– існує можливість розпізнавання відеопотоку, який можна отримати безпосередньо з камери пристрою.

Також представлена можливість інтеграції як SDK для мов програмування C, C++, C#, Java, Python, PHP.

3.2.3 Розпізнавання полів документів від VK Cloud

Рішення з розпізнавання полів документів є одним із застосувань технології розпізнавання осіб, об'єктів, процесів на базі машинного навчання та штучних нейронних мереж Vision від VK Cloud.

Для розпізнавання полів документів достатньо передати зображення у вигляді файлу на відповідний ендпоінт. Відповіддю на запит є JSON із розпізнаними значеннями полів паспорта.

Дане рішення також не підтримує автономну роботу, працюючи лише через сервери VK Cloud, що призводить до додаткових ризиків щодо безпеки та доступності при недостатньо хорошому з'єднанні.

3.2.4 IRISmart security

Сервіс є програмою для персонального комп'ютера на ОС Windows. Цікавими особливостями є підтримка широкого числа моделей сканерів, що дозволяє скоротити кількість необхідних дій від користувача; автоматичне експортування даних паспортів у файли Excel [50].

Схема роботи сервісу [50] представлена на рис. 3.2.

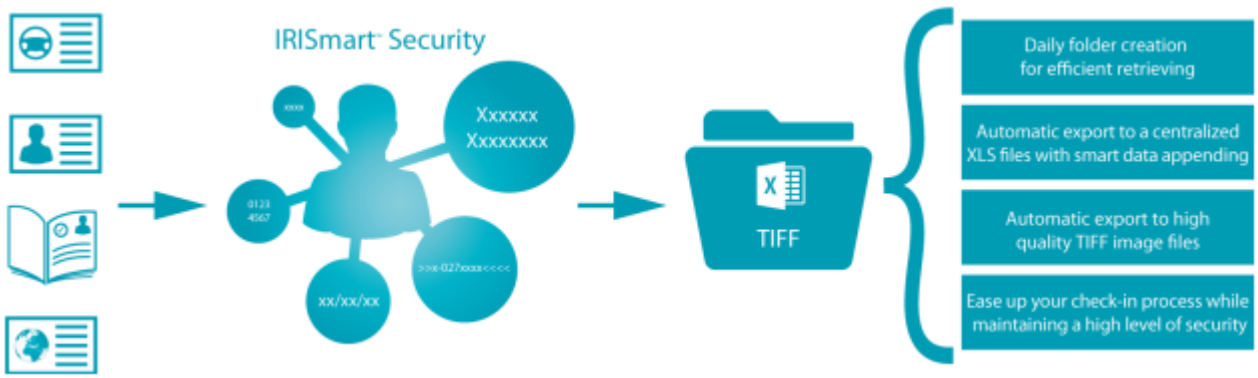

Рисунок 3.2 – Схема роботи IRISmart security

Цільовою аудиторією для даного програмного рішення є приватне підприємництво, наприклад готельний бізнес, кемпінги, спортивні клуби та агенції з оренди автомобілів, що звужує сферу застосування сервісу , але наділяє унікальними відмінними рисами [50].

Крім цього, одним із недоліків даного рішення є відсутність API , що також звужує потенційну сферу застосування.

3.3 Вимоги до системи

За рахунок аналізу аналогів були сформульовані такі вимоги до сервісу, що розробляється:

– можливість використання різних алгоритмів сегментації;

– розширюваність, тобто можливість впровадження додаткових алгоритмів;

– REST API, можливість інтеграції у вже існуючі системи;

– наявність веб-інтерфейсу;

– автономна робота;

– достатня точність для сегментації зображення.

3.4 Використовувані технології

Для розробки веб-програми було обрано мову програмування Python.

Як фреймворк для реалізації REST API був обраний Flask.

Для веб-інтерфейсу вибрано шаблонізатор Jinja та CSS -фреймворк Bootstrap.

3.5 Архітектура рішення

Сервіс використовує класичну клієнт-серверну архітектуру. Для взаємодії з користувачем використовується протокол HTTP/HTTPS.

Архітектуру сервісу можна подати як діаграму компонентів, представлену на рис. 3.3.

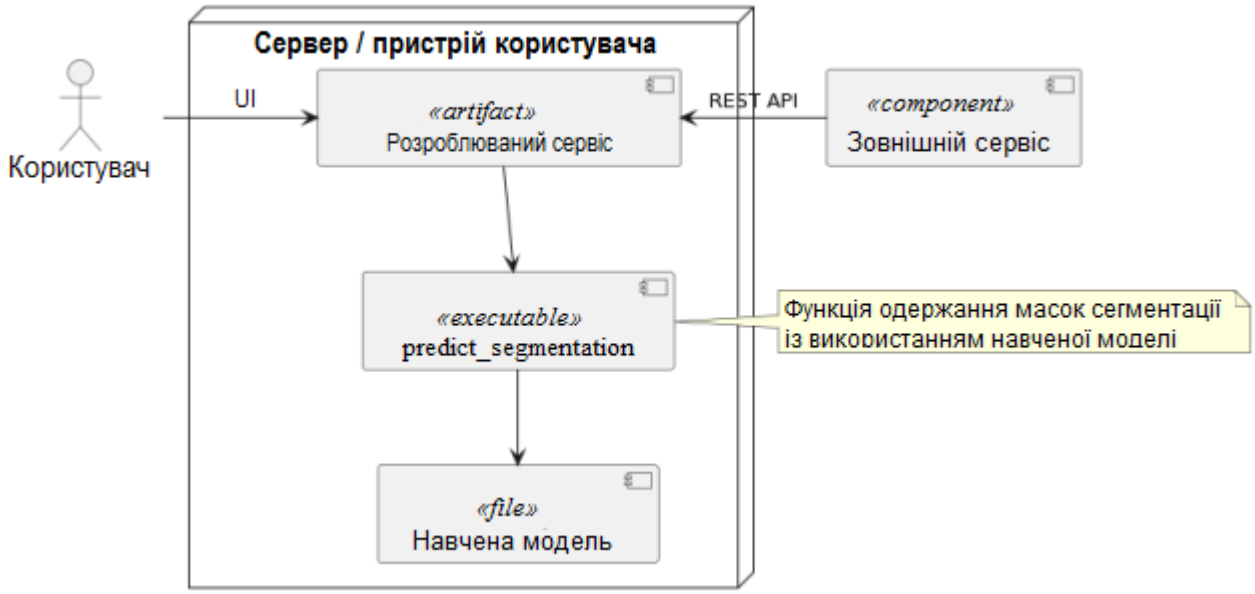

Рисунок 3.3 – Діаграма компонентів сервісу

Для отримання масок сегментації зображення використовується REST API. Нижче наведено опис ендпоінтів API:

POST /api/v1/upload – метод завантаження файлу на сервер.

Для завантаження необхідно передати зображення з розширенням .jpeg, .jpg, .png або .gif у полі файлу тіла запиту (multipart/form-data).

Параметри тіла відповіді у форматі JSON:

file_name - назва файлу передачі в метод сегментації.

Приклад відповіді:

{ "file_name": "ede38718-59b9-45f0-821b-0d6aaec08f96.jpg" }

GET /api/v1/models - метод отримання списку моделей з конфігураційного файлу .env.

Тілом відповіді є JSON з парами ключ моделі - назва моделі. Приклад відповіді напведено у дістингу 3.1.

Лістинг 3.1 – Приклад відповіді

```
{
    "PSPNET": "PSPNet",
    "RESNET50_PSPNET": "ResNet50 + PSPNet", "RESNET50_SEGNET":
    "ResNet50 + SegNet",
    "RESNET50_UNET": "ResNet50 + U-Net",
    "SEGNET": "SegNet",
    "UNET": "U-Net",
    "VGG16_PSPNET": "VGG16 + PSPNet",
    "VGG16_SEGNET": "VGG16 + SegNet",
    "VGG16_UNET": "VGG16 + U-Net"
}
```

GET /api/v1/segmentation - метод для отримання масок сегментації раніше завантаженого зображення.

Параметри запиту представлені у табл. 3.1.

Таблиця 3.1 – Параметри запиту

Назва параметра	**Тип значення**	**Обов'язковість**	**Опис**
file_name	string	+	Назва файлу з тіла відповіді методу завантаження
model_key	string	+	Ключ моделі
show_legends	bool	-	Необхідність вказівки назв класів на зображенні (за замовчуванням False)
overlay_image	bool	-	Необхідність наявності вихідного зображення на фоні (за замовчуванням False)

Тілом відповіді є маски сегментації як зображення з типом image/png.

Для зберігання та подальшого використання інформації про моделі, використовується файл .env, що зберігає в собі конфігурацію про модель у наступному форматі:

```
DOC_SEGMENTATION.MODEL.<KEY>.NAME =< NAME >
DOC_SEGMENTATION.MODEL.<KEY>.PATH=<PATH>
```

Ключ моделі (KEY) використовується в API REST для вказівки моделі для сегментації. Назва моделі (NAME) визначає відображуване значення у полі вибору моделі у веб-інтерфейсі. Шлях до моделі (PATH) позначає абсолютний або відносний шлях до директорії та префікс файлу стану навченої моделі (для сегментації використовується файл стану останньої доби).

Приклад конфігурації для однієї з моделей:

```
DOC_SEGMENTATION.MODEL.VGG16_UNET.NAME='VGG16 + U-Net'
DOC_SEGMENTATION.MODEL.VGG16_UNET.PATH=../models/unet/vgg16
_unet/vgg16_unet_
```

Цей підхід забезпечує розширюваність сервісу, спрощуючи додавання нових навчених моделей у сервіс.

3.6 Технічна реалізація

У рамках рішення було реалізовано сервіс з REST API та веб-інтерфейссом на основі шаблонізатора Jinja. Зображення веб-інтерфейсу представлено на рис. 3.4.

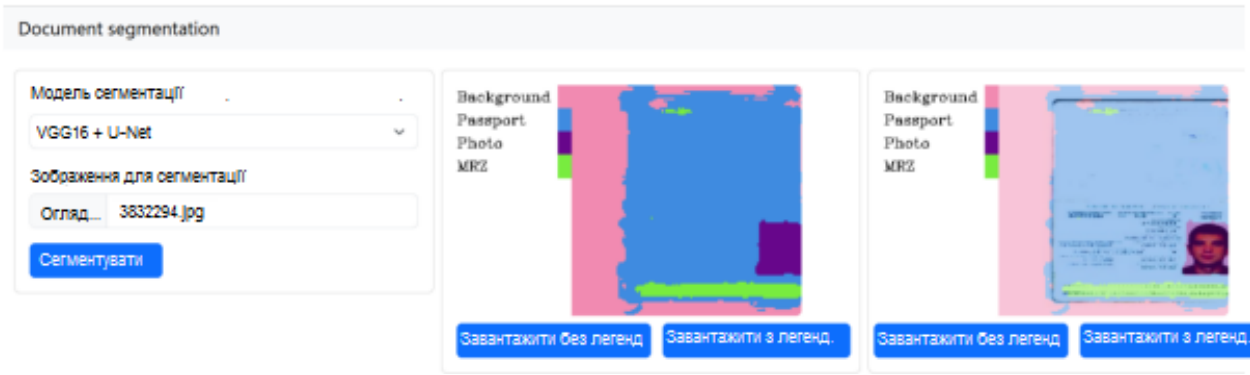

Рисунок 3.4 – Зображення веб-інтерфейсу сервісу

Для роботи сервісу локально необхідно виконати такі кроки:

– встановити Python 3.11 та пакетний менеджер pip відповідної версії;

– завантажити проект із репозиторію;

– завантажити або навчити моделі сегментації (для навчання можна скористатися функцією train із файлу segmentation/tram.py, приклад

використання знаходиться у файлі example.py);

– вказати назви та шляхи до навчених моделей у файлі .env;

– виконати наступну команду в корені проекту для встановлення залежності мостів: pip install -r requirements.txt;

– запустити виконання файлу doc_segmentation_app.py: python -m app.doc_segmentation_app.

РОЗДІЛ 4. БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Долікарська допомога при ураженні електричним струмом

Удар електричним струмом є поширеною травмою і часто може закінчитися летальним випадком. Ураження електричним струмом відбувається підчас контакту тіла людини з джерелом електричної енергії під напругою. Це реакція організму людини на проходження електричного струму через тіло. Вона проявляється по різному, від легких потрясінь до небезпечних здоров'ю травм, які можуть вплинути на тканини в організмі.

Шкода завдана електричним струмом залежать від кількох факторів: наскільки висока була напруга, яка область тіла була вражена і від виду струму. Фізичні наслідки для людини можуть коливатися від опіків частин тіла до серйозних вражень внутрішніх органів [52].

Часто люди стикаються з підвищеним ризиком ураження високою напругою. Низька напруги не наносить серйозних травм для люди, а з іншої сторони висока напруги, яка має більше 500 В, може призвести до серйозного пошкодження тканин. У дітей або підлітків при враженні електричним струмом в діапазоні від 110 до 200 В можуть вникнути значні травми. Зазвичай це трапляється підчас порушення техніки безпеки при роботі з електричними приладами, які є в побуті. Це можуть бити електричні шнури, подовжувачі, несправні розетки та багато чого іншого.

Виділяють чотири основні фактори від яких залежить вплив електричного струму на організм: величина струму, що протікає через тіло; органи, через які проходить струм; час, протягом якого струм вражає тіло; частота струму.

Фізичні наслідки на пряму залежать від величини струму, яка вражає тіло людини. Струм менший за 1 мА не завдає жодної шкоди людина фізичного ефекту. При 1 мА можуть відчуватися слабкі поколювання, а при 5 мА – легкий поштовх. Однак при струмі ведичною від 6 до 25 мА людина може відчувати больовий шок і деяку втрату контролю над м'язами.

При ураженні електричним струмом від 50 до 150 мА може спричинити у людини сильний біль, м’язове скорочення і навіть призвести до зупинку дихання. У певних випадках можливий летальний результат для людини. Струм від 1000 до 4300 мА призводить до ймовірної смерть, тому що дана напруга спричиняє пошкодження нервових зав’язків, м’язових скорочення та порушення ритму серця. До 10 000 мА електричний струм спричиняє важкі опіки шкіри людини та зупинку серця. Висока ймовірність смерті.

Найпоширеніші ознаки та симптоми враження електричним струмом включають:

- втрата свідомості;
- ускладнення або зупинка дихання;
- опіки, які виникають там, де струм входить і виходить з тіла;
- зупинка серця;
- слабкий і непостійний пульс або його припинення.

Перш за все, що потрібно зробити при першій допомозі, це відключити джерело живлення. Вимкнути електропостачання, від’єднати електроприлад від джерела електричного струму або вимкнути блок запобіжників, якщо він знаходиться неподалік. Не потрібно намагатися підходити близько до жертви, якщо не переконані, що це безпечно і живлення вимкнено.

Потрібно бути обережним у вологих місцях, тому що вода є електричним провідником і рятівник може стати теж жертвою. Якщо людина не впевнена щодо вологості поверхні, потрібно відключити основне електропостачання будинку. У випадку коли це неможливо, використати підручний предмет, який не є провідником і відокремити людину від джерела струму. Це може бути дерев’яна або пластмасова річ.

Після того як постраждалого відокремили від джерела електрики, потрібно викликати швидку і надати першу медичну допомогу. Далі потрібно визначити стан жертви. Перевірити, чи людина у свідомості і дихає. У складних випадках у жертви може бути слабкий пульс або його відсутність. Можливо, що дихання зупинилося. Якщо людина втратила свідомість і перестала дихати, потрібно почати серцево-легеневу реанімацію. Руки розташовуємо в центрі грудної, одна

на іншу. Сильно і швидко натиснути 30 раз приблизно до третини діаметра грудної клітки. Після кожного натискання на грудну клітку робиться два рятувальні вдихи. Потрібно відкинути голову потерпілого назад і підняти підборіддя. Затиснути ніс і створити повне ущільнення. Далі подути потерпілому в рот і подивитися, чи підніметься грудна клітка. Потрібно продовжувати робити натискань на грудну клітку та вдих, поки не прибуде медична допомога або людина не почне сама дихати. Якщо потерпілий живий, перемістити його у зручне йому положення подальше від небезпеки. Можна запобігти шоку, поклавши людину рівно на землю, з головою трохи нижче тіла [51].

Якщо людина при свідомості, нормально дихає і на тілі є опіки, потрібно накрити їх звичайною харчовою плівкою або іншою неклейкою пов’язкою, але без мазі чи лосьйону. Якщо кровотеча у потерпілого, може знадобитися компресія та джгут.

При роботі з соціальною мережею користувач повинен знати і вміти як правильно поводитися з ПК, тому що людина перебуває у непосередньому контакті з джерелом напруги. Удар електричним струмом є потенційно смертельною травмою. Негайна медична допомога важлива, щоб запобігти серйозним травмам і смерті. Для запобігання уражень електричним струмом при роботі за ПК слід встановити додаткові захисні пристрої, що забезпечують недоступність токопровідних частин для дотику. Для зменшення небезпеки використовувати розділовий трансформатор для розв'язки з основною мережею.

Удар електричним струмом є потенційно смертельною травмою. Негайна медична допомога важлива, щоб запобігти серйозним травмам і смерті.

4.2 Вимоги ергономіки до організації робочого місця оператора ПК

Робоче місце – це ділянка простору, яка облаштована необхідним обладнанням, відповідно до трудової діяльності, для виконання поставлених завдань.

Правильно побудоване робоче місце повинне забезпечувати:

– найкраще розміщення обладнання і предметів праці;

– не допускати дискомфорту;

– підвищувати продуктивність праці;

– зменшувати втому працівника.

Розмір робочого місця повинен бути таким, щоб людина не виконувала лишніх рухів і не відчувала дискомфорту під час виконання роботи. Також для працівника важливо мати змогу змінити робочу позу, наприклад, положення тулуба, рук або ніг. Потрібно мінімізувати або звести до нуля всі незручності положення тіла [53].

Різні дослідження заявляють, що при правильному проектуванні робочого місця продуктивність людини може зрости від 15-25%. Такі фактори як рівень освітлення, вологість повітря, температура, шум, вібрація, токсичність, мають значний вплив на умови життєдіяльності і працездатності людини.

Антропометричні вимоги визначають відповідність робочого місця до фізіологічних параметрів тіла людини як зріст і розміри тіла. Індикатором цього є правильна робоча поза, відсутність дискомфорту, оптимальні зони досягнення, раціональні рухи. Психофізіологічні та фізіологічні вимоги формулюють відповідність обладнання і робочого місця можливостям співробітника щодо розуміння, обробки даних, пошук і реалізації рішень.

Організація робочого місця передбачає наступні пункти:

– раціональне положення робочого місця у приміщенні;

– вибір робочих меблів відповідно до фізіологічних характеристик працівника;

– правильне компонування і розміщення обладнання на робочих місцях;

– урахування особливостей та характеру професійної діяльності.

До загальних принципів організації робочого місця відносять:

– робоче місце повинне містити тільки ті предмети, які беруть учать у робочому процесі, але не заважати йому;

– предмети, які часто використовуються у роботі, розміщуються ближче, ніж ті речі, якими користуються рідше;

– предмети, які беруться лівою рукою, повинні розміщуватися зліва, а предмети, які використовуються правою рукою — справа;

– якщо при роботі з предметом працівник використовує дві руки, то він розміщується з урахуванням зручності захоплювання його двома руками;

– робоче місце не повинно бути засмічене;

– необхідна оглядовість повинна бути забезпечена при правильній організації робочого місця.

Робоча поза – це найбільш тривале положення тіла працівника протягом робочого дня. При зручній робочій позі забезпечується стійкість положення тулуба, ніг, рук, голови і витрачається мінімальний запас енергії та максимальну продуктивність [53].

Сидячи і стоячи – дві найпопулярніших пози у робочому процесі. При проектуванні робочого місця потрібно враховувати, що з фізичним навантаженням бажана поза стоячи, а при малих зусиллях – сидячи. При роботі стоячи, людина стомлюється більше ніж сидячи. У відсотковому еквіваленті це на 10% більше енергії. При додатковому навантаженні підвищується артеріальний і венозний тиск крові, розширення вен, пошкоджуються ступені та викривляється хребет. У свою чергу при сидячій роботі нижня частина тіла розслаблена, а основне навантаження спрямоване на м'язи шиї, спини, таза, стегон. При неправильній сидячій позі розвивається застій крові у ногах, а якщо пальці виконують багато роботи можливе запалення суглобів.

Організація робочого місця при використанні ПК повинна відповідати усім ергономічним вимогам. Ключові ергономічні вимоги до проектування робочого місця оператора ПК зображені на рисунку 4.1.

При роботі з персональним комп’ютером потрібно:

– зменшувати кількість статичних напружень;

– розподіляти кількість і час статичних напружень;

– змінювати робочі пози під професійної діяльності.

Саме вибір правильної робочої пози визначається від впливу багатьох факторів. Одні з найважливіших це – кількість зусиль яка прикладається, велична робочої зони, відношення висоти робочої поверхні і ростом працівника.

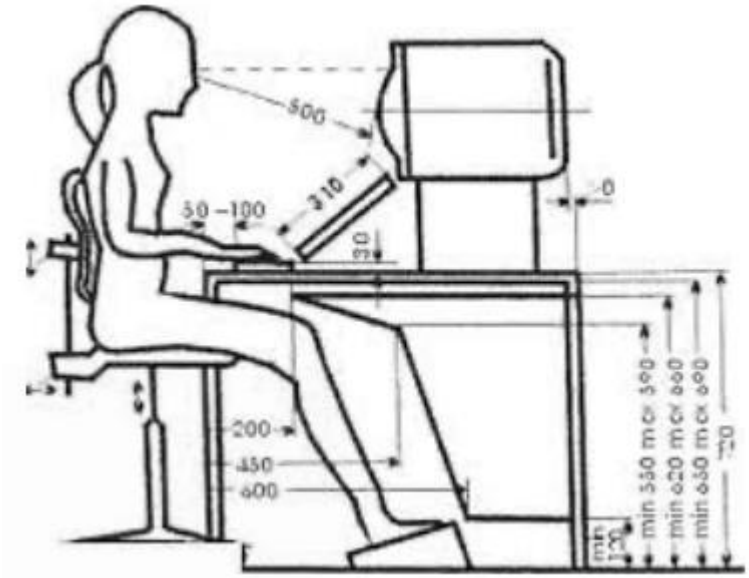

Рисунок 4.1 – Робочий стіл і розміщення користувача ПК

При використанні соціальної мережі користувач повинен дотримуватися вище зазначених правил, які будуть сприяти комфортній і продуктивній роботі. Важливо регулярно робити короткі перерви. Часта зміна заняття – кращий спосіб уникнути можливих неприємностей.

ВИСНОВКИ

При виконанні роботи було отримано такі результати:

– проведено дослідження підходів, технологій сегментації зображень, у тому числі архітектур нейронних мереж, що використовуються при роботі алгоритмів сегментації;

– розроблені алгоритми сегментації документів на основі архітектур CNN, що включають U-Net, SegNet, PSPNet без використання і з використанням попередньо навчених моделей VGG16, ResNet50;

– для реалізованих алгоритмів було підраховано різні метрики IoU, що показують достатню точність для сегментації;

– на основі розроблених алгоритмів було спроектовано архітектуру та реалізовано сервіс для сегментації зображень документів з можливостями використання різних алгоритмів сегментації та впровадження додаткових, можливістю автономної роботи, наявністю REST API та веб- інтерфейсу.

Подальший розвиток розробки вбачається у розширенні сервісу шляхом впровадження додаткових CNN, доопрацювання реалізованих моделей для поліпшення якості сегментації.

ПЕРЕЛІК ДЖЕРЕЛ

1. Luo D. et al. Deep learning for automatic image segmentation in stomatology and its clinical application //Frontiers in Medical Technology. 2021. p. 68

2. Santosh K.C. Document Image Analysis. Springer Nature Singapore Pte Ltd. 2018. P. 174. https://doi.org/10.1007/978-981-13-2339-3.

3. Proceedings of the International Conference on Document Analysis and Recognition, ICDAR [Електронний ресурс] – Режим доступа: https://www.scimagojr.com/journalsearch.php?q=75898&tip=sid (дата звертання: 28.04.25).

4. Attivissimo F. et al. An automatic reader of identity documents //2019 IEEE International Conference on Systems, Man and Cybernetics (SMC). IEEE, 2019. pp. 3525-3530.

5. Zaaboub W. et al. Neural network-based system for automatic passport stamp classification //Information Technology and Control. 2020. V. 49. №. 4. pp. 583-607.

6. Liu Y. et al. MRZ code extraction from visa and passport documents using convolutional neural networks //International Journal on Document Analysis and Recognition (IJDAR). 2022. V. 25. №. 1. pp. 29-39.

7. Othman P. S. et al. Image processing techniques for identifying impostor documents through digital forensic examination //Image Process. Tech. 2020. V. 62. pp. 1781-1794.

8. Nguyen G. et al. Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey //Artificial Intelligence Review. 2019. V. 52. pp. 77-124.

9. Albawi S., Mohammed T. A., Al-Zawi S. Understanding of a convolutional neural network //2017 international conference on engineering and technology (ICET). IEEE, 2017. PP. 1-6.

10. .Convolutional Neural Network (CNN) | by Raycad | Medium [Електронний ресурс] – Режим доступа: https://medium.com/raycad.seedotech/convolutional-neural-network-cnn-8d1908c010ab (дата звертання: 28.04.25).

11. Input object. Keras API reference [Електронний ресурс] – Режим доступа: https://keras.io/api/layers/core_layers/input/ (дата звертання: 28.04.25).

12. Ajit A., Acharya K., Samanta A. A review of convolutional neural networks //2020 international conference on emerging trends in information technology and engineering (ic-ETITE). IEEE, 2020. PP. 1-5.

13. O'Shea K., Nash R. An introduction to convolutional neural networks //arXiv preprint arXiv:1511.08458. 2015.

14. Матійчук Л., Готович В., Бонар В. Порівняння ефективності методів некерованого машинного навчання для виявлення аномалій в OBD2 даних. Вимірювальна та обчислювальна техніка в технологічних процесах. Хмельницький національний університет. (1), 2025. с. 407–414.

15. Wang Y., Xiao Z., Cao G. A convolutional neural network method based on Adam optimizer with power-exponential learning rate for bearing fault diagnosis //Journal of Vibroengineering. 2022. V. 24. №. 4. PP. 666-678.

16. Bieder F., Sandkühler R., Cattin P. C. Comparison of methods generalizing max-and average-pooling //arXiv preprint arXiv:2103.01746. 2021.

17. Zhang Y. D. et al. A five-layer deep convolutional neural network with stochastic pooling for chest CT-based COVID-19 diagnosis //Machine Vision and Applications. 2021. V. 32. PP. 1-13.

18. MaxPooling2D layer. Keras API reference [Електронний ресурс] – Режим доступа: https://keras.io/api/layers/pooling_layers/max_pooling2d/ (дата звертання: 28.04.25).

19. Л.В. Волинець, Н.А. Гарматюк, В.А. Готович. Великі за обсягом набори біомедичних даних та машинне навчання. Збірник тез доповідей XII Міжнародної науково-практичної конференції молодих учених та студентів «АКТУАЛЬНІ ЗАДАЧІ СУЧАСНИХ ТЕХНОЛОГІЙ». – Тернопіль, 6-7 грудня 2023 р. с. 370-371

20. Zhao Y. et al. GUN: Gradual upsampling network for single image superresolution //IEEE Access. 2018. V. 6. PP. 39363-39374.

21. Szandała T. Review and comparison of commonly used activation functions for deep neural networks // Bio-inspired neurocomputing. 2021. PP. 203-224.

22. UpSampling2D layer. Keras API reference [Електронний ресурс] – Режим доступа: https://keras.io/api/layers/reshaping_layers/up_sampling2d/ (дата звертання: 28.04.25).

23. Gerstner W. et al. Neuronal dynamics: From single neurons to networks and models of cognition. – Cambridge University Press, 2014. Ping W. et al. Deep voice 3: Scaling text-to-speech with convolutional sequence learning //arXiv preprint arXiv:1710.07654. 2017.

24. Radu M. D., Costea I. M., Stan V. A. Automatic Traffic Sign Recognition Artificial Inteligence-Deep Learning Algorithm //2020 12th International Conference on Electronics, Computers and Artificial Intelligence (ECAI). IEEE, 2020. PP. 1-4.

25. Rasamoelina A. D., Adjailia F., Sinčák P. A review of activation function for artificial neural network //2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI). IEEE, 2020. PP. 281-286.

26. Krizhevsky A., Sutskever I., Hinton G. E. Imagenet classification with deep convolutional neural networks //Communications of the ACM. 2017. V. 60. №. 6. PP. 84-90.

27. В.І. Саламандра, В.А. Готович. Використання технології комп’ютерного зору для спрощення анімації персонажів. Збірник тез доповідей X Міжнародної науково-практичної конференції молодих учених та студентів "АКТУАЛЬНІ ЗАДАЧІ СУЧАСНИХ ТЕХНОЛОГІЙ". – Тернопіль, 24-25 листопада 2021 р. Том 1, с.118

28. Lin M., Chen Q., Yan S. Network in network //arXiv preprint arXiv:1312.4400. 2013.

29. Nwankpa C. et al. Activation functions: Comparison of trends in practice and research for deep learning //arXiv preprint arXiv:1811.03378. 2018.

30. Es-Sabery F. et al. Sentence-level classification using parallel fuzzy deep learning classifier //IEEE Access. 2021. V. 9. PP. 17943-17985.

31. Layer activation functions. Keras API reference [Електронний ресурс] – Режим доступа: https://keras.io/api/layers/activations/ (дата звертання: 28.04.25).

32. A Gentle Introduction to Batch Normalization for Deep Neural Networks [Електронний ресурс] – Режим доступа: https://machinelearningmastery.com/batch-normalization-fortraining-of-deep-neural-networks/ (дата звертання: 28.04.25).

33. Ioffe S., Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift //International conference on machine learning // Proceedings of Machine Learning Research, 2015. PP. 448-456.

34. Batch normalization in 3 levels of understanding | by Johann Huber | Towards Data Science [Електронний ресурс] – Режим доступа: https://towardsdatascience.com/batch-normalization-in-3-levels-of-understanding14c2da90a338#b93c (дата звертання: 28.04.25).

35. BatchNormalization layer. Keras API reference [Електронний ресурс] – Режим доступа: https://keras.io/api/layers/normalization_layers/batch_normalization/ (дата звертання: 28.04.25).

36. VGG16 and VGG19. Keras API reference [Електронний ресурс] – Режим доступа: https://keras.io/api/applications/vgg (дата звертання: 28.04.25).

37. Keras Applications. Keras API reference [Електронний ресурс] – Режим доступа: https://keras.io/api/applications/ (дата звертання: 28.04.25).

38. The Architecture and Implementation of VGG-16 | by Vaibhav Khandelwal | Towards AI [Електронний ресурс] – Режим доступа: https://pub.towardsai.net/the-architecture-and-implementation-of-vgg-16-b050e5a5920b?gi=e5f8a6174a34 (дата звертання: 28.04.25).

39. ResNet and ResNetV2. Keras API reference [Електронний ресурс] – Режим доступа: https://keras.io/api/applications/resnet/ (дата звертання: 28.04.25).

40. Ji Q. et al. Optimized deep convolutional neural networks for identification of macular diseases from optical coherence tomography images //Algorithms. 2019. V. 12. №. 3. PP. 51-52

41. He K. et al. Deep residual learning for image recognition //Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. PP. 770-778.

42. Ronneberger O., Fischer P., Brox T. 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. – PP. 234-241.

43. Jun T. J. et al. T-net: Nested encoder–decoder architecture for the main vessel segmentation in coronary angiography //Neural Networks. 2020. V. 128. PP. 216-233.

44. Zhao H. et al. Pyramid scene parsing network //Proceedings of the IEEE conference on computer vision and pattern recognition. 2017. PP. 2881-2890.

45. Model training APIs. Keras API reference [Електронний ресурс] – Режим доступа: https://keras.io/api/models/model_training_apis/ (дата звертання: 28.04.25).

46. Wang Y., Xiao Z., Cao G. A convolutional neural network method based on Adam optimizer with power-exponential learning rate for bearing fault diagnosis //Journal of Vibroengineering. 2022. V. 24. №. 4. PP. 666-678.

47. Yeung M. et al. Unified focal loss: Generalising dice and cross entropy-based losses to handle class imbalanced medical image segmentation //Computerized Medical Imaging and Graphics. 2022. V. 95. PP. 1020-1026.

48. Thoma M. A survey of semantic segmentation //arXiv preprint arXiv:1602.06541. 2016.

49. Passport Recognition Service by Evergreen [Електронний ресурс] – Режим доступа: https://ocr.solutions/en (дата звертання: 28.04.25).

50. IRISmart Security - ID and passport recognition software [Електронний ресурс] – Режим доступа: https://www.irislink.com/EN-GB/c2030/IRISmart-Security---ID-and-passport-recognition-software.aspx (дата звертання: 28.04.25).

51. Oleh Pastukh, Volodymyr Stefanyshyn, Ihor Baran, Ihor Yakymenko and Vasyl Vasylkiv. Mathematics and software for controlling mobile software devices based on brain activity signals. CEUR Workshop Proceedings, 2023, 3628, pp. 330–337.

52. Заікіна Д., Глива В. Основи охорони праці та безпека життєдіяльності. 2019. URL: https://doi.org/10.31435/rsglobal/001 (дата звернення: 14.05.2025).

53. Безпека в надзвичайних ситуаціях. Методичний посібник для здобувачів освітнього ступеня «магістр» всіх спеціальностей денної та заочної (дистанційної) форм навчання / укл.: Стручок В. С. Тернопіль: ФОП Паляниця В. А., 2022. 156 с.