

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ
ПОЛІПШЕННЯ ЯКОСТІ ЗОБРАЖЕНЬ В УМОВАХ НИЗЬКОЇ РОЗДІЛЬНОЇ
ЗДАТНОСТІ
RESEARCH AND SOFTWARE IMPLEMENTATION OF METHODS FOR
IMPROVING IMAGE QUALITY IN LOW-RESOLUTION CONDITIONS

Виконав(ла): студент(ка) 4 курсу, групи КНТ-120

Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

АЗМАНОВ І. П.

(ПРИЗВИЩЕ та ініціали)

Керівник СУББОТІН С.О.

(ПРИЗВИЩЕ та ініціали)

Рецензент ШИТКОВА О.В.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2024 року

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

АЗМАНОВА Іллі Павловича
(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація методів поліпшення якості зображень в умовах низької роздільної здатності
Research and Software Implementation of Methods for Improving Image Quality in Low-resolution Conditions

керівник проєкту (роботи) д.т.н., професор, СУББОТІН Сергій Олександрович
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “ 24 ” квітня 2024 року № 168

2. Строк подання студентом проєкту (роботи) 10 червня 2024 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз проблемної області та постановка завдань дослідження. 2. Матеріали і методи. 3. Розробка програмного забезпечення. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)
Слайди презентації

6. Консультанти розділів проєкту (роботи)

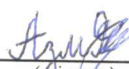
Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	СУББОТІН С.О., професор		
Нормоконтроль	ЛИПОВЕЦЬ М.В., асистент		

7. Дата видачі завдання “ 15 ” квітня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

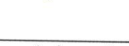
№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка архітектури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділ 3, 4
6	Тестування та експериментальне дослідження програмного забезпечення.	5 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	6 тиждень	Додатки
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка)


(підпис)

Ілля АЗМАНОВ
(Ім'я ПРІЗВИЩЕ)

Керівник проєкту (роботи)


(підпис)

Сергій СУББОТІН
(Ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
83 с., 4 табл., 22 рис., 3 дод., 81 джерело.

PYTHON, PYTORCH, ГЛИБОКЕ НАВЧАННЯ, НАДРОЗДІЛЬНА
ЗДАТНІСТЬ ЗОБРАЖЕНЬ, SRCNN, VDSR, SRGAN, EDSR, НЕЙРОННІ
МЕРЕЖІ.

Об'єкт дослідження – процес опрацювання зображень для підвищення
їхньої роздільної здатності.

Предмет дослідження – алгоритми та методи поліпшення якості
зображень із низькою роздільною здатністю.

Мета роботи – дослідити, проаналізувати та програмно реалізувати
моделі підвищення роздільної здатності зображень.

Матеріали, методи та технічні засоби: методи глибокого навчання,
нейронні мережі, Python, бібліотека PyTorch, метрики оцінки якості зображень.

Результати: реалізовано та протестовано чотири моделі надроздільної
здатності зображень. Проведено порівняльний аналіз їхньої ефективності на
основі метрик PSNR і SSIM. Встановлено, що EDSR демонструє вищі показники
якості зображення порівняно з класичними методами.

Практична цінність роботи полягає в можливості використання
результатів дослідження для поліпшення якості зображень у різних додатках.

Висновки: досягнуто мети роботи, підтверджено ефективність моделей
глибокого навчання в задачах надроздільної здатності зображень.
Рекомендовано використання EDSR для практичних застосувань у галузі
медицини, відеоспостереження та супутникової зйомки.

Галузь використання: результати роботи можуть бути застосовані в
різних галузях, де потрібне поліпшення якості зображень, включно з медичною
візуалізацією, відеоспостереженням і аналізом супутникових даних, а також для

розробників програмного забезпечення та дослідників у галузі комп'ютерного зору.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 83 pages, 4 tables, 22 figures, 3 appendix, 81 sources.

PYTHON, PYTORCH, DEEP LEARNING, SUPER-RESOLUTION IMAGES, SRCNN, VDSR, SRGAN, EDSR, NEURAL NETWORKS.

The object of research is the process of image processing to increase their resolution.

The subject of research is algorithms and methods for improving the quality of low-resolution images.

The purpose of the study is to investigate, analyze, and programmatically implement models for improving image resolution.

Materials, methods, and techniques: deep learning methods, neural networks, Python, PyTorch library, image quality assessment metrics.

Results: four models of ultra-high-resolution images were implemented and tested. A comparative analysis of their effectiveness based on the PSNR and SSIM metrics was performed. It is established that EDSR demonstrates higher image quality indicators compared to classical methods.

The practical value of the work lies in the possibility of using the results of the study to improve image quality in various applications.

Conclusions: the aim of the work has been achieved, the effectiveness of deep learning models in ultra-high-resolution image processing has been confirmed. We recommend the use of EDSR for practical applications in the fields of medicine, video surveillance, and satellite imaging.

Applications: The results of this work can be used in a variety of fields where image quality improvement is required, including medical imaging, video surveillance, and satellite data analysis, as well as for software developers and researchers in the field of computer vision.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	9
Вступ.....	10
1 Аналіз проблемної області та постановка завдань дослідження	12
1.1 Поняття суперроздільності. Сучасний стан області.....	12
1.2 Задача суперроздільності	14
1.3 Огляд відомих методів підвищення роздільної здатності	16
1.3.1 Алгоритми однокадрової надроздільної здатності.....	17
1.3.2 Алгоритми багатокадрової надроздільної здатності.....	25
1.3.3 Алгоритми надвисокої роздільної здатності відеоданих.....	29
1.4 Висновки за розділом 1	31
2 Матеріали і методи.....	33
2.1 Вибір мови програмування	33
2.2 Вибір середовища розробки реалізації моделей надвисокої роздільної здатності	37
2.3 Вибір фреймворку	40
2.4 Підсумки за розділом 2.....	43
3 Розробка програмного забезпечення.....	44
3.1 Структура програмного забезпечення	44
3.2 Функціонування програмного забезпечення для побудови та навчання моделей для задач надвисокої роздільної здатності зображень.....	46
3.3 Висновки за розділом 3	48
4 Експлуатація, тестування та експериментальне дослідження програми	49
4.1 Звернення до програми.....	49
4.2 Вхідні дані для тестування.....	50
4.3 Оцінювання якості моделей.....	51
4.4 Результати обробки зображень.....	53
4.5 Висновки за розділом 4	55
Висновки	56

Перелік джерел посилань	58
Додаток А Технічне завдання	68
Додаток Б Фрагмент тексту програми	72
Додаток В Слайди презентації	76

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

CNN – Convolutional Neural Network;

EDSR – Enhanced Deep Residual Networks;

GAN – Generative Adversarial Network;

HR – High Resolution;

JPEG – Joint Photographic Experts Group;

LR – Low Resolution;

MSE – Mean Squared Error;

PNG – Portable Network Graphics;

PSNR – Peak Signal-To-Noise Ratio;

SR – Super-resolution;

SRCNN – Super Resolution Convolutional Neural Network;

SRGAN – Super Resolution Generative Adversarial Network;

VDSR – Very Deep Super Resolution;

ВР – висока роздільна здатність;

ЗНМ – згорткова нейронна мережа;

МП – мова програмування;

НР – низька роздільна здатність;

ПЗ – програмне забезпечення;

ТЗ – технічне завдання.

ВСТУП

Останніми роками в галузі обробки зображень спостерігається значний прогрес, особливо в області надвисокої роздільної здатності зображень. Надвисока роздільна здатність зображень має на меті підвищити якість зображень з низькою роздільною здатністю, реконструюючи їхні аналоги з високою роздільною здатністю з покращеною візуальною точністю. Ця технологія знаходить застосування в різних сферах, включаючи медичну візуалізацію, відеоспостереження, супутникові знімки та покращення мультимедійного контенту. Зростаючий попит на високоякісні зображення в цих галузях зумовив потребу в ефективних методах надвисокої роздільної здатності зображень.

Традиційно методи надвисокої роздільної здатності зображень ґрунтувалися на інтерполяційних методах, таких як бікубічна інтерполяція, що часто призводило до розмитих і насичених артефактами зображень. Однак з появою глибинного навчання та згорткових нейронних мереж (CNN) ситуація з надвисокою роздільною здатністю зображень зазнала парадигмальних змін. Підходи, засновані на глибокому навчанні, продемонстрували вищу продуктивність як за кількісними показниками, так і за якістю сприйняття порівняно з традиційними методами.

Незважаючи на значний прогрес, досягнутий в області надроздільності зображень за допомогою глибокого навчання, залишається потреба у всебічному порівняльному аналізі цих моделей для оцінки їх ефективності та придатності для різних застосувань. Крім того, практична реалізація та розгортання цих моделей вимагають ретельного розгляду обчислювальної ефективності, розміру моделі та швидкості виведення.

Метою цього дослідження є вивчення та порівняння чотирьох відомих моделей надвисокої роздільної здатності зображень, що базуються на глибокому навчанні: SRCNN, VDSR, SRGAN та EDSR. Дослідження має на меті оцінити їхню продуктивність з точки зору метрик якості зображення, таких як пікове

відношення сигнал/шум (PSNR) та індекс структурної подібності (SSIM), а також їхню обчислювальну ефективність. Провівши ретельний аналіз, це дослідження має на меті надати уявлення про сильні та слабкі сторони кожної моделі та запропонувати рекомендації щодо їх практичного застосування в різних галузях.

Основні завдання цього дослідження полягають у наступному:

- провести всебічний огляд літератури про сучасні методи надроздільної здатності зображень, зосередившись на підходах, що базуються на глибокому навчанні;
- реалізувати та навчити моделі SRCNN, VDSR, SRGAN та EDSR за допомогою мови програмування Python та фреймворку глибокого навчання PyTorch;
- оцінити продуктивність кожної моделі за допомогою стандартних метрик якості зображень, таких як PSNR та SSIM, на різноманітних наборах даних зображень низької роздільної здатності;
- проаналізувати обчислювальну ефективність кожної моделі, враховуючи такі фактори, як розмір моделі, час навчання та швидкість виведення;
- порівняти результати, отримані за допомогою реалізованих моделей, і надати уявлення про їхні сильні та слабкі сторони для практичного застосування.

Передбачається, що результати цього дослідження зроблять внесок у сферу надвисокої роздільної здатності зображень, надаючи комплексну оцінку сучасним моделям глибокого навчання. Інформація, отримана з цього дослідження, може допомогти дослідникам і практикам у виборі найбільш підходящої моделі для їхніх конкретних потреб. Крім того, деталі практичної реалізації та порівняльний аналіз, представлені в цій роботі, можуть слугувати цінним ресурсом для майбутніх досліджень у галузі надроздільної здатності зображень.

1 АНАЛІЗ ПРОБЛЕМНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

У першому розділі буде проведено детальний аналіз предметної області поліпшення якості зображень в умовах низької роздільної здатності. Цей аналіз охоплює огляд теоретичних основ, класифікацію методів, а також опис ключових алгоритмів і моделей.

1.1 Поняття суперроздільності. Сучасний стан області

Для ефективної роботи сучасних цифрових застосунків зазвичай потрібні зображення або відео високої роздільної здатності, що зумовлено двома ключовими факторами. По-перше, висока роздільна здатність забезпечує більш деталізоване представлення інформації в зображенні; по-друге, це допомагає алгоритмам машинного навчання і штучного інтелекту краще інтерпретувати і розпізнавати візуальні дані [1].

Тип камери, що використовується для отримання зображення, має значний вплив на його якість, особливо при низькій роздільній здатності, що може призвести до розмиття або пікселізації, втрати або неправильної інтерпретації важливих деталей. Це створює проблеми на подальших етапах обробки зображень, таких як розпізнавання об'єктів, анотування і генерація текстових описів. Це вимагає розробки систем, здатних перетворювати зображення низької роздільної здатності на зображення високої якості [2].

Для вирішення цього питання зображення з низькою роздільною здатністю можна змодельовати із зображення з високою роздільною здатністю [3], використовуючи формулу (1.1):

$$I_x = D(I_y; \sigma), \quad (1.1)$$

де I_x – зображення з низькою роздільною здатністю;

D – функція деградації;

I_y – зображення з високою роздільною здатністю;

σ – параметри процесу деградації (наприклад, коефіцієнт масштабування або шум).

При роботі з парами зображень високої та низької роздільної здатності, точні значення параметрів деградації, таких як D і σ , часто невідомі. У багатьох випадках ми маємо лише саме зображення з високою роздільною здатністю і відповідне йому зображення з низькою роздільною здатністю [4]. Таким чином, тепер можна визначити поняття надроздільної здатності.

Надалі під завданням створення надроздільної здатності (SR) буде матися на увазі процес оброблення низки вхідних цифрових зображень низької роздільної здатності, результатом якого є одержання одного або декількох зображень високої роздільної здатності. Основна мета цього процесу - створення зображень із вищою деталізацією і якістю, що перевершує початкові зображення [3], [5].

Технологія надвисокої роздільної здатності знаходить застосування в широкому спектрі галузей. Одним з найважливіших напрямків є медична візуалізація [6]-[8], де SR допомагає покращувати якість медичних зображень, таких як рентгенівські знімки, МРТ (магнітно-резонансна томографія) або КТ (комп'ютерна томографія). Це дозволяє лікарям отримувати більш детальну інформацію і приймати більш точні діагностичні рішення.

Іншою важливою сферою застосування SR є відеоспостереження та гарантування безпеки [9]-[10]. Також окрім безпосереднього поліпшення візуальної якості, технологія SR також сприяє вирішенню інших завдань комп'ютерного зору [11]-[14]. Наприклад, вона може бути використана як попередній етап обробки для покращення результатів сегментації зображень, розпізнавання або трекінгу об'єктів.

Сучасні методи SR, особливо ті, що базуються на моделях глибокого навчання, демонструють вражаючі результати. Зокрема, деякі з них, такі як [15]-[21], досягли точності розпізнавання облич більше ніж 99% на публічних

наборах даних. Це свідчить про високу ефективність і потенціал застосування SR для задач, пов'язаних з біометричною ідентифікацією та верифікацією.

На сьогоднішній день розроблено широкий спектр алгоритмів SR, які базуються на різноманітних методах і моделях. Ці алгоритми охоплюють як класичні підходи, що беруть своє коріння в теорії обробки сигналів, так і більш сучасні техніки, засновані на використанні нейронних мереж. У наступних розділах ці алгоритми, описані у відомих наукових джерелах, будуть розглянуті детальніше.

Важливо підкреслити, що ця робота не зосереджується на вдосконаленні апаратних засобів, таких як камери, сенсори або системи передачі даних, хоча вони також відіграють важливу роль у підвищенні якості зображень та зменшенні рівня шуму. Натомість, основна увага приділяється методам та алгоритмам суперроздільної здатності, які працюють з уже існуючими технічними системами.

1.2 Задача суперроздільності

Ключовим завданням у сфері надроздільної здатності є ефективне використання додаткової інформації, що міститься в отриманих зображеннях, під час повторного опрацювання. Теоретичні основи для розв'язання цієї проблеми було закладено ще 1964 року Гаррісом [22], який запропонував теореми для розв'язання задачі дифракції в оптичних системах.

Однак лише через два десятиліття, у 1984 році, Цай і Хуанг [23] вперше застосували концепцію SR на практиці для підвищення просторової роздільної здатності супутникових зображень Landsat TM. Їх робота стала відправною точкою для багатьох дослідників, які почали активно вивчати і розвивати методи SR як у теоретичному, так і в прикладному аспектах [24]-[28]. З того часу минуло сорок років, і дослідження в галузі SR значно просунулися вперед. Цей прогрес можна охарактеризувати наступним чином.

На початку розвитку методів підвищення роздільної здатності зображень, більшість з них базувалися на обробці у частотній області [23],[29]. Ці алгоритми використовують зв'язок між зображенням високої роздільної здатності та спостереженнями низької роздільної здатності, спираючись на просту теоретичну основу. Це забезпечує їх високу обчислювальну ефективність. Проте, ці методи мають певні недоліки, такі як чутливість до помилок моделювання та труднощі при роботі зі складними моделями руху. Ці обмеження стримують їх подальший розвиток.

У зв'язку з недоліками частотних методів, фокус досліджень змістився на алгоритми, що працюють у просторовій області [30]. Серед відомих просторових методів можна відзначити нерівномірну інтерполяцію, ітеративну зворотну проекцію (IBP), проекцію на опуклі множини (POCS), регуляризовані методи, а також деякі гібридні алгоритми, що поєднують різні підходи.

Утім, завдяки стрімкому прогресу в галузі глибокого навчання, моделі суперроздільної здатності, засновані на цьому підході, стали предметом активних досліджень. Ці моделі часто демонструють найкращі результати в різних еталонних задачах суперроздільної здатності. Для вирішення таких задач використовуються різні методи глибокого навчання, від ранніх підходів, що базуються на згорткових нейронних мережах (CNN), таких як SRCNN [31], до сучасних методів, які використовують моделі з механізмами уваги, такі як SAN [32] та HAN [33].

Крім того, нещодавно були запропоновані моделі, засновані на архітектурі трансформера, такі як IPT [34] та SwinIR [35], які адаптують архітектуру трансформера для задач обробки зображень, в тому числі для суперроздільної здатності, і показують відмінні результати.

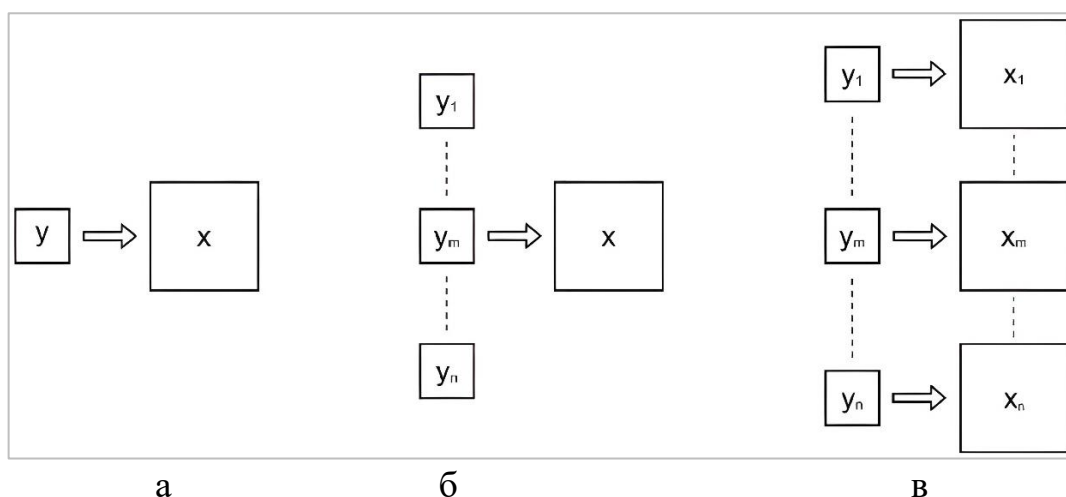
Алгоритми суперроздільної здатності, засновані на глибокому навчанні, відрізняються за кількома ключовими аспектами. Вони можуть мати різні архітектури мереж, використовувати різні функції втрат, застосовувати різні принципи та стратегії навчання тощо. Ці відмінності дозволяють адаптувати

алгоритми до специфічних вимог конкретних задач і досягати високих результатів у підвищенні роздільної здатності зображень.

1.3 Огляд відомих методів підвищення роздільної здатності

Алгоритми надроздільної здатності можуть працювати з різною кількістю вхідних і вихідних зображень, але на практиці виділяють три основні сценарії [5] (рис. 1.1):

- однокадрова суперроздільна здатність: на вхід алгоритму надходить одне зображення низької роздільної здатності, яке обробляється для збільшення роздільної здатності [5];
- багатокadroва суперроздільна здатність: на вхід подаються кілька кадрів, що представляють одну сцену з невеликими зміщеннями між ними. Алгоритм комбінує інформацію з цих кадрів для створення одного зображення високої роздільної здатності з покращеною деталізацією [5];
- суперроздільна здатність відеоданих: алгоритм приймає серію кадрів низької якості та генерує відповідні зображення високої роздільної здатності для кожного з них [5].



а – однокадрова суперроздільність; б – багатокadroва суперроздільність; в – суперроздільність відеоданих.

Рисунок 1.1 - Схеми побудови алгоритмів суперроздільної здатності [4]

Важливо відзначити, що методи покращення роздільної здатності, які використовують серію зображень або відеопослідовність, спираються на надлишковість інформації, присутню в кількох кадрах однієї сцени. Ця додаткова інформація дозволяє алгоритмам досягати кращих результатів у порівнянні з обробкою одного зображення. Часто такі підходи базуються на методах і алгоритмах, розроблених для підвищення роздільної здатності одного зображення, але адаптовані для роботи з множиною кадрів [5].

1.3.1 Алгоритми однокадрової надроздільної здатності

Останніми роками, починаючи з 2010-х років, методи глибокого навчання стають найпоширенішими в цій області [1]. Ця робота не претендує на вичерпне висвітлення теми, тут буде розглянуто деякі з цих методів, які становлять інтерес з погляду реалізації і порівняння.

Спершу необхідно описати деякі класичні методи підвищення дискретизації, оскільки в низці методів глибокого навчання як будівельні блоки моделі використовують блоки апсемплінгу, засновані на класичних методах [4].

Класичні методи інтерполяції, такі як метод найближчого сусіда [36], білінійна [37] та бікубічна [38] інтерполяція, є одними з найпростіших способів збільшення роздільної здатності зображення. Ці методи відрізняються високою швидкістю завдяки своїй простоті, але не завжди забезпечують достатню точність результатів. Незважаючи на це, бікубічна інтерполяція часто використовується як складова частина більш складних методів суперроздільної здатності. Зокрема, її застосовують для генерації початкового наближення зображення з високою роздільною здатністю (ВР), яке потім покращується за допомогою алгоритмів багатокadroвої суперроздільної здатності [4].

Попри це, точність результатів інтерполяції можна покращити, застосовуючи додаткові методи при генерації вихідного зображення з високою роздільною здатністю. Наприклад, можна використовувати вейвлетне перетворення для обробки зображення низької роздільної здатності або виявляти

межі об'єктів на ньому [39]. Ці додаткові кроки дозволяють отримати більш деталізоване та чітке зображення [5].

Іншим популярним підходом до вирішення задачі однокадрової суперроздільної здатності є використання бази зразків, яка містить множину пар зображень низької та високої роздільної здатності [40]-[44]. У цьому випадку вихідне зображення низької роздільної здатності розбивається на велику кількість перекривних фрагментів (зазвичай, квадратних блоків невеликого розміру, таких як 3×3 або 5×5 , хоча іноді використовуються і більші блоки, наприклад, 7×7). Потім кожен фрагмент замінюється відповідним зразком високої роздільної здатності з попередньо побудованої бази. Нарешті, отримані фрагменти високої роздільної здатності об'єднуються в одне результуюче зображення [4].

У рамках підходу, заснованого на використанні бази зразків, існує два основні способи її побудови:

- зовнішня база зразків: у цьому випадку база створюється на основі певної навчальної вибірки, яка містить велику кількість зображень високої роздільної здатності [40]-[42]. Ці зображення використовуються для формування пар "зразок низької роздільної здатності - зразок високої роздільної здатності", які потім застосовуються для обробки нових зображень [4];

- внутрішня база зразків: цей підхід ґрунтується на властивості самоподібності, притаманній природним зображенням. База зразків будується на основі єдиного вхідного зображення низької роздільної здатності [43]-[44]. Іншими словами, алгоритм шукає схожі фрагменти в межах самого зображення і використовує їх для генерації відповідних фрагментів високої роздільної здатності [4].

Кожен з цих способів має свої переваги та недоліки. Зовнішня база зразків дозволяє використовувати різноманітні приклади зображень високої якості, але може бути менш адаптованою до особливостей конкретного зображення. Внутрішня база зразків, навпаки, повністю налаштована на специфіку вхідного зображення, але обмежена інформацією, доступною в ньому самому.

1.3.1.1 Методи, засновані на використанні згорткових нейронних мереж

В останні роки ЗНМ набули широкого застосування для вирішення різноманітних задач обробки даних, і однокадрова суперроздільна здатність не стала винятком. Сучасні дослідження пропонують різні архітектури нейронних мереж, здатні перетворювати зображення низької роздільної здатності у зображення високої роздільної здатності [45]-[48].

Ці роботи стали логічним продовженням і розвитком ідей, закладених у дослідженнях, заснованих на використанні баз зразків. Замість явного зберігання пар зразків НР-ВР, згорткові нейронні мережі здатні автоматично навчатися виявляти та використовувати відповідні ознаки і залежності в даних для генерації зображень високої якості.

1.3.1.1.1 SRCNN

Глибока згорткова нейронна мережа SRCNN, вперше запропонована в [46], здатна безпосередньо перетворювати вхідне зображення низької роздільної здатності на зображення високої роздільної здатності. Пізніше ця архітектура була вдосконалена в [47] для підвищення швидкодії.

Її архітектура, як правило, містить три основні шари (рис. 1.2). Перший шар, відомий як шар виділення ділянок, витягує перекривні ділянки з вхідного зображення НР і перетворює їх на високорозмірні вектори. Наступний шар, шар нелінійного відображення, трансформує ці вектори в інший набір високорозмірних векторів, які відповідають їхнім аналогам з високою роздільною здатністю. Останній шар, шар реконструкції, агрегує отримані вектори для відновлення зображення ВР [49].

Для навчання SRCNN використовуються пари зображень низької та високої роздільної здатності. Мережа навчається функції відображення, яка генерує зображення ВР на основі його версії НР. Перед обробкою за допомогою

SRCNN зображення HP масштабуються до бажаного розміру традиційними методами. Потім SRCNN уточнює ці масштабовані зображення, генеруючи чіткіші та детальніші зображення BP [49].

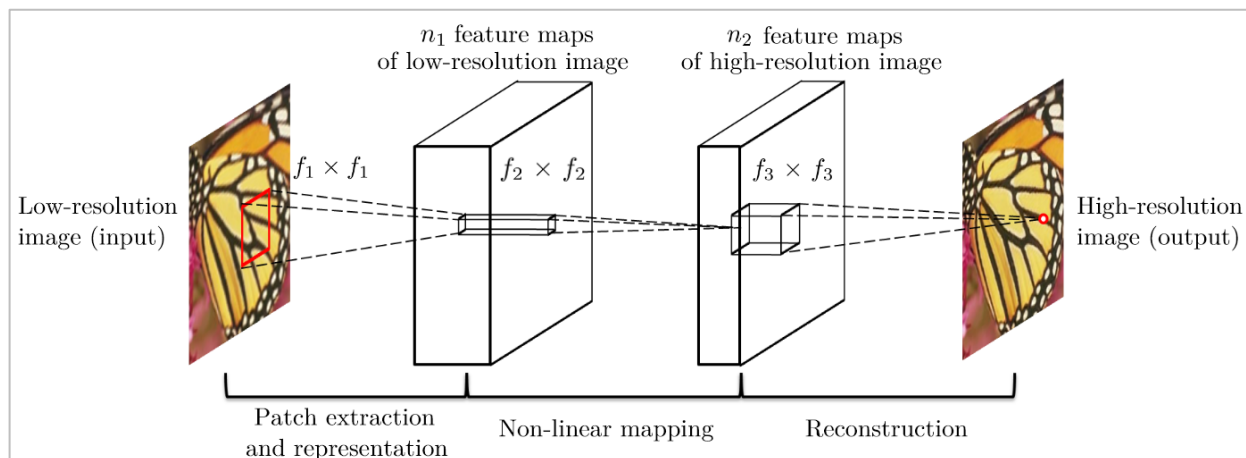


Рисунок 1.2 – Архітектура SRCNN [46]

Незважаючи на свої переваги, SRCNN не позбавлена обмежень. Обчислювальна інтенсивність SRCNN може бути перешкодою, особливо для завдань з дуже високою роздільною здатністю. Крім того, на збільшених зображеннях можуть виникати артефакти, особливо в областях, де оригінальному зображенню бракує деталей [49].

Постійна еволюція в цій галузі призвела до розробки більш досконалих моделей, які ґрунтуються на фундаменті, закладеному SRCNN. Ці новіші моделі спрямовані на усунення деяких обмежень SRCNN, пропонуючи покращену продуктивність та ефективність [49].

1.3.1.1.2 VDSR

У роботі [48] було запропоновано вдосконалення підходу до суперроздільної здатності на основі згорткових нейронних мереж. Автори значно збільшили глибину мережі, розширивши кількість шарів, що навчаються, з 3 до 20. Крім того, вони змінили цільову функцію мережі: замість безпосереднього оцінювання зображення високої роздільної здатності, мережа

тепер передбачає залишкове зображення BP. Це залишкове зображення потрібно додати до результату бікубічної інтерполяції вихідного зображення низької роздільної здатності, щоб отримати фінальне зображення BP.

Ці модифікації дозволили суттєво підвищити точність отриманих результатів у порівнянні з оригінальною архітектурою SRCNN. Збільшення глибини мережі дає змогу моделі вивчати більш складні та абстрактні ознаки, що сприяє генерації більш деталізованих та реалістичних зображень BP. Водночас, використання залишкового зображення спрощує задачу навчання мережі, оскільки їй потрібно передбачати лише різницю між бікубічною інтерполяцією та цільовим зображенням BP, а не все зображення BP з нуля.

Процес генерації зображення з високою роздільною здатністю за допомогою VDSR можна описати наступним чином [50]:

- вхідне зображення низької роздільної здатності спочатку інтерполюється до бажаного розміру за допомогою традиційних методів, таких як бікубічна інтерполяція. Це інтерпольоване зображення (ILR) потім подається на вхід мережі VDSR [50];
- зображення ILR проходить через (D-1) згорткових шарів (Conv), кожен з яких супроводжується функцією активації ReLU. Ці шари навчаються виділяти та перетворювати ознаки зображення, поступово збільшуючи їх абстрактність та складність [50];
- після (D-1) згорткових шарів слідує останній згортковий шар (Conv.D), який генерує залишкове зображення. Це залишкове зображення представляє різницю між інтерпольованим зображенням ILR та цільовим зображенням з високою роздільною здатністю [50];
- отримане залишкове зображення додається до інтерпольованого зображення ILR, щоб отримати фінальне зображення з високою роздільною здатністю [50].

Завдяки глибокій архітектурі (рис. 1.3) та залишковому навчанню, VDSR може ефективно вивчати складні нелінійні відображення між зображеннями низької та високої роздільної здатності. Це дозволяє мережі генерувати більш

деталізовані та реалістичні зображення з високою роздільною здатністю, з меншою кількістю артефактів у порівнянні з попередніми методами [50].

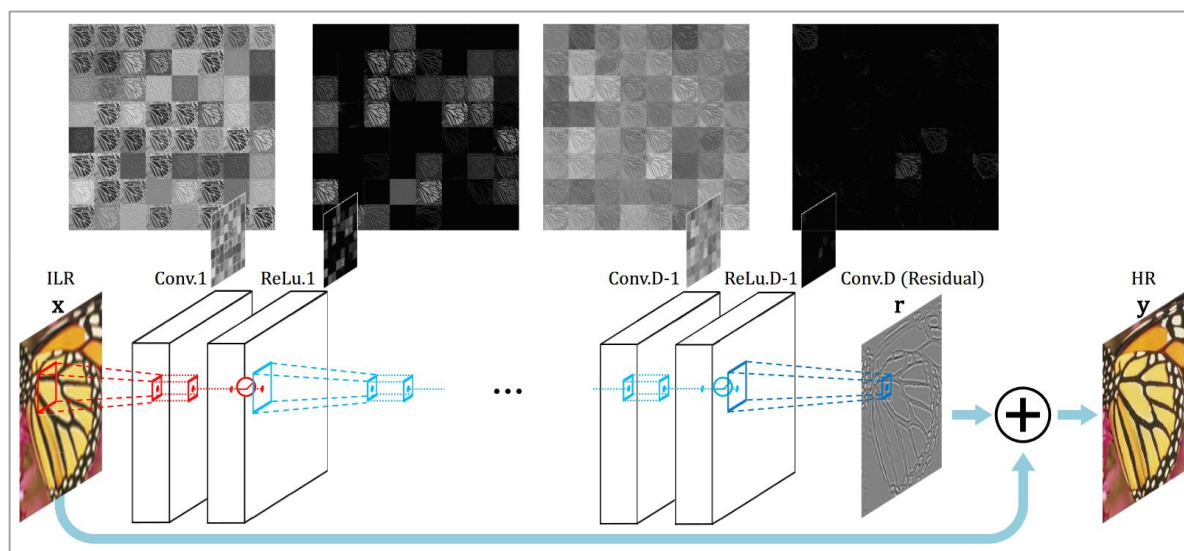


Рисунок 1.3 – Архітектура VDSR [48]

Крім того, VDSR навчається на повних зображеннях, а не на локальних патчах, що дозволяє мережі враховувати глобальний контекст при генерації зображень з високою роздільною здатністю. Також VDSR може обробляти зображення з різними коефіцієнтами масштабування за допомогою єдиної моделі, що робить її більш універсальною та практичною [50].

1.3.1.1.3 SRGAN

SRGAN, або Super-Resolution Generative Adversarial Network, - це метод глибокого навчання, який використовується для суперрозширення зображень, що є процесом підвищення роздільної здатності зображення за рахунок збільшення кількості пікселів. Він був представлений у 2017 році дослідниками з Twitter [51].

Основна ідея SRGAN [45] полягає у використанні архітектури генеративної змагальної мережі (GAN), яка складається з двох нейронних мереж: генератора та дискримінатора (рис. 1.4). Мережа-генератор приймає зображення

використовуючи можливості глибокого навчання та навчання в умовах змагання [51].

Однак важливо зазначити, що SRGAN, як і інші методи глибокого навчання, вимагає великої кількості навчальних даних і обчислювальних ресурсів, а його продуктивність може змінюватися залежно від якості та різноманітності навчальних даних [51].

1.3.1.1.4 EDSR

EDSR (Enhanced Deep Residual Networks for Single Image Super-Resolution) - це модель машинного навчання, випущена в липні 2017 року, яку можна використовувати для збільшення роздільної здатності зображення [52].

На відміну від SRGAN, яка використовує архітектуру генеративної змагальної мережі (GAN), EDSR базується на архітектурі глибокої згорткової нейронної мережі (CNN) (рис. 1.5). Мережа призначена для вивчення зв'язку між фрагментами зображення з низькою та високою роздільною здатністю, що дозволяє їй відновлювати зображення з високою роздільною здатністю за їхніми аналогами з низькою роздільною здатністю [53].

Однією з ключових особливостей EDSR є використання залишкових блоків, які натхненні архітектурою ResNet, що використовується в задачах класифікації зображень. Ці залишкові блоки допомагають вирішити проблему зникаючого градієнта - поширену проблему при навчанні глибоких нейронних мереж, дозволяючи градієнтам легше проходити через мережу під час процесу навчання [53].

EDSR також включає в себе кілька інших методів для підвищення продуктивності, таких як пакетна нормалізація, яка допомагає стабілізувати процес навчання, і мультимасштабний підхід, який дозволяє мережі більш ефективно обробляти зображення різного масштабу [53].

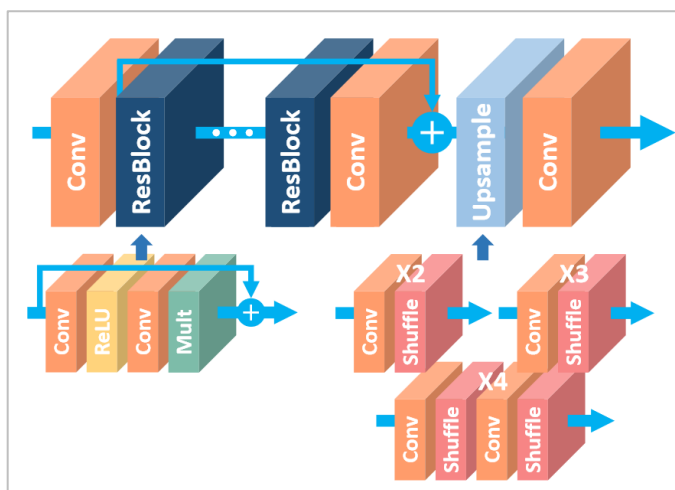


Рисунок 1.5 – Архітектура EDSR [53]

Порівняно з SRGAN, EDSR, як правило, створює більш точні та достовірні реконструкції оригінальних зображень високої роздільної здатності, особливо з точки зору збереження дрібних деталей і текстур. Однак він не завжди може генерувати візуально такі ж привабливі результати, як SRGAN, який призначений для створення більш переконливих для сприйняття зображень [52].

Крім того, звичайні моделі машинного навчання для надвисокої роздільної здатності є чутливими до навчання, і невелика зміна в архітектурі може мати великий вплив на якість зображення. Тому, навіть якщо використовується одна і та ж модель, може з'явитися велика різниця в якості зображення в залежності від початкового значення ваг і техніки, що використовується під час навчання. EDSR досягає стабільного навчання, навчаючись на x_2 , а потім навчаючись на x_3 і x_4 з вагами x_2 [52].

1.3.2 Алгоритми багатокadroвої надроздільної здатності

При створенні багатокadroвої надвисокої роздільної здатності алгоритм отримує на вході низку зображень низької роздільної здатності, які показують ту саму сцену, але з невеликими субпіксельними зсувами між кадрами. Ці зсуви надають додаткову інформацію про вихідні дані і можуть бути спричинені рухом камери з часом або рухом об'єктів у кадрі [5].

На відміну від однокадрових методів СР, алгоритми обробки послідовності зображень зазвичай включають додатковий крок аналізу вхідних кадрів НР. Цей етап передбачає оцінку зміщень між кадрами з метою їх подальшого вирівнювання один відносно одного. Після цього алгоритм генерує єдине зображення високої роздільної здатності на виході. Вирівнювання кадрів може здійснюватися різними методами, наприклад, шляхом оцінки оптичного потоку між двома кадрами або визначення матриць зсувів і поворотів [5].

Алгоритми багатокадрового підвищення роздільної здатності, які базуються на обробці зображень у частотній області [23], [29], [54], використовують властивості перетворень, таких як дискретне перетворення Фур'є або дискретне косинусне перетворення. Ці методи об'єднують кадри низької роздільної здатності у частотній області. З одного боку, ці методи характеризуються високою швидкістю, але з іншого боку, простота використовуваних моделей може негативно впливати на точність результатів [5].

Для покращення якості отриманих результатів у літературі також розглядається підхід, який використовує вейвлет-перетворення вихідних кадрів низької роздільної здатності [55]-[56]. Цей підхід має на меті підвищити точність результатів багатокадрового підвищення роздільної здатності шляхом застосування більш складних методів обробки зображень [5].

Методи багатокадрового підвищення роздільної здатності, які базуються на інтерполяції [57]-[58], зазвичай включають три етапи: вирівнювання кадрів низької роздільної здатності, інтерполяцію для отримання зображення високої роздільної здатності та покращення якості результуючого зображення високої роздільної здатності (наприклад, підвищення різкості). Ці методи обчислювально ефективні, але не забезпечують високої точності [5].

У статистичному підході [59]-[61] зображення високої роздільної здатності оцінюється як максимум апостеріорної ймовірності, враховуючи спостережувані кадри низької роздільної здатності, з використанням методів байєсівського виведення. Цей підхід дозволяє використовувати апріорну інформацію про зображення та методи регуляризації для накладання обмежень

на шукану оцінку високої роздільної здатності. У цьому підході зображення розглядаються як випадкові марковські поля [5].

Поширеним підходом до багатокадрового підвищення роздільної здатності є ітеративне уточнення початкової оцінки зображення високої роздільної здатності в міру опрацювання чергового вхідного зображення низької роздільної здатності. Цей підхід дозволяє поступово покращувати якість результуючого зображення, враховуючи додаткову інформацію з кожного нового кадру. Зокрема, цей підхід реалізовано в [62]-[63] за допомогою алгоритму ітеративної зворотної проєкції (Iterative Back-Projection), який ґрунтується на мінімізації різниці між спостережуваними кадрами низької роздільної здатності та відповідними прогнозованими кадрами, отриманими з поточної оцінки зображення високої роздільної здатності. У [64] використано метод проєкції на опуклі множини (Projection onto Convex Sets), що дає змогу використовувати апіорні відомості про зображення високої роздільної здатності, такі як обмеження на значення пікселів або просторову гладкість, для покращення якості результатів [5].

У [65]-[67] розглядаються алгоритми, засновані на оптимальній лінійній фільтрації калманівського типу. Ці алгоритми також реалізують ітеративне уточнення початкової оцінки зображення високої роздільної здатності в міру обробки нових кадрів низької роздільної здатності. Вони використовують статистичні моделі зображень та шуму для оптимального оцінювання зображення високої роздільної здатності на кожному кроці. Алгоритми підвищення роздільної здатності такого типу забезпечують результати високої точності, але характеризуються значною обчислювальною складністю через необхідність обробки великих обсягів даних та виконання складних математичних операцій на кожній ітерації [5].

Так само як і в разі однокадрового підвищення роздільної здатності, останніми роками для розв'язання задачі побудови багатокадрового підвищення роздільної здатності дедалі частіше застосовують методи, засновані на нейромережевих технологіях [68]-[71]. Ці методи використовують потужність

глибокого навчання для автоматичного виявлення та використання просторово-часових залежностей між кадрами низької роздільної здатності. При цьому можна виокремити низку відмінних підходів до організації оброблення зображень низької роздільної здатності, що надходять на вхід. Ці підходи найчастіше передбачають використання одразу декількох нейронних мереж для розв'язання різних завдань, що виникають під час оброблення, таких як реєстрація кадрів, інтерполяція, усунення шуму та підвищення різкості [5].

Підхід, описаний у [69], ґрунтується на двоетапному процесі підвищення роздільної здатності. На першому етапі кожен з вихідних кадрів низької роздільної здатності незалежно обробляється за допомогою одного з алгоритмів однокадрового підвищення роздільної здатності, що використовують штучні нейронні мережі, такі як [46] та [47]. Ці алгоритми застосовують глибокі згорткові нейронні мережі для збільшення роздільної здатності кожного кадру окремо, використовуючи лише просторову інформацію в межах цього кадру. В результаті першого етапу отримуємо набір зображень високої роздільної здатності, які відповідають вихідним кадрам низької роздільної здатності. На другому етапі отримані на першому етапі зображення високої роздільної здатності об'єднуються в одне вихідне зображення високої роздільної здатності за допомогою іншої згорткової нейронної мережі. Ця мережа призначена для злиття інформації з різних кадрів та використання часової надмірності для подальшого покращення якості результуючого зображення. Вона може враховувати такі фактори, як рух об'єктів між кадрами, зміни освітлення та наявність шуму. Завдяки цьому другому етапу досягається додаткове підвищення роздільної здатності та покращення візуальної якості вихідного зображення порівняно з результатами, отриманими на першому етапі. Таким чином, цей підхід поєднує переваги однокадрового та багатоканального підвищення роздільної здатності, використовуючи потужність нейронних мереж на кожному етапі обробки [5].

У [70] застосовується подібний підхід, який передбачає незалежну обробку всіх кадрів низької роздільної здатності за допомогою нейронної

мережі. Проте, на відміну від попереднього методу, результатом роботи цієї мережі є не безпосередньо зображення високої роздільної здатності, а набір ознак, що відповідають кожному з кадрів низької роздільної здатності. Далі всі отримані ознаки зображень низької роздільної здатності комбінуються в одне залишкове зображення високої роздільної здатності за допомогою другої нейронної мережі. Кінцеве зображення високої роздільної здатності формується шляхом додавання початкової оцінки високої роздільної здатності, отриманої усередненням вхідних зображень низької роздільної здатності, та попередньо обчисленого залишкового зображення [5].

Алгоритм, запропонований у [71], як і метод з [70], ґрунтується на обробці ознак, отриманих з кадрів низької роздільної здатності. Однак подальша обробка цих ознак виконується рекурсивно за допомогою спеціально розробленої нейронної мережі [5].

1.3.3 Алгоритми надвисокої роздільної здатності відеоданих

Алгоритми підвищення роздільної здатності відео мають багато спільних рис з алгоритмами, що використовуються для покращення роздільної здатності зображень на основі кількох кадрів. Проте, головною відмінністю є те, що при обробці відеопослідовності для отримання кожного кадру з високою роздільною здатністю враховуються не лише відповідний йому кадр з низькою роздільною здатністю, а й певна кількість (m) сусідніх кадрів з низькою роздільною здатністю, розташованих до і після нього. Таким чином, незалежно від потенційно необмеженої тривалості всього відео, обробка кадрів відбувається з використанням "ковзного вікна", яке на кожному кроці охоплює $2m+1$ зображень (рис. 1.6). Це дає можливість брати до уваги часову інформацію та взаємозв'язки між сусідніми кадрами, що в результаті дозволяє покращити якість кадрів з високою роздільною здатністю [5].

У роботі [72] запропоновано метод підвищення роздільної здатності зображень, який базується на статистичному підході. Цей метод використовує

ітеративно переважений алгоритм найменших квадратів для максимізації апостеріорної ймовірності отриманого зображення високої роздільної здатності. В процесі роботи алгоритм оцінює оптичний потік, адитивний шум та ядра розмиття, що дозволяє покращити якість реконструйованого зображення.

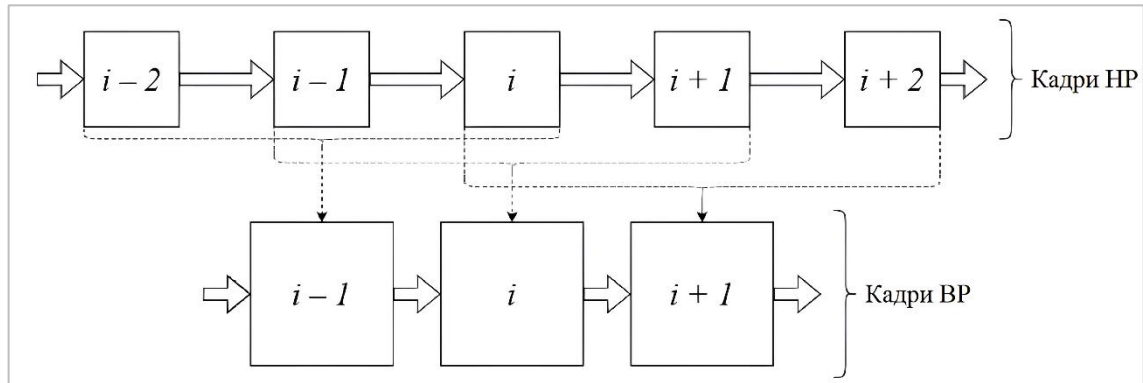


Рисунок 1.6 – Послідовне оброблення кадрів під час побудови суперроздільної здатності відеоданих у разі $m = 1$ [5]

В роботі [73] запропоновано вдосконалений підхід до підвищення роздільної здатності зображень, який базується на глибокій згортковій нейронній мережі. На відміну від попереднього методу, ця мережа використовує не тільки поточний кадр низької роздільної здатності, але й один або два сусідні кадри (до i після поточного) для отримання більш повної інформації про сцену. В результаті, на виході мережі формується зображення високої роздільної здатності, що відповідає саме поточному кадру, але з покращеною деталізацією завдяки врахуванню часової інформації.

Робота [74] розвиває ідеї, представлені в [73], пропонуючи інноваційний підхід до спільного навчання нейронних мереж для підвищення роздільної здатності та оцінки оптичного потоку. Замість роздільного вирішення цих задач, автори пропонують навчати дві мережі одночасно, дозволяючи їм обмінюватися інформацією та взаємно покращувати свої результати. Такий підхід дозволяє досягти вищої точності як в підвищенні роздільної здатності, так і в оцінці руху, оскільки мережі вчаться враховувати взаємозв'язки між цими задачами та оптимізувати свої параметри для досягнення спільної мети.

У роботі [75] представлено альтернативний підхід до підвищення роздільної здатності зображень, який також базується на згорткових нейронних мережах. Однак, на відміну від попередніх робіт, автори фокусуються на проблемі точної реєстрації (вирівнювання) вхідних кадрів низької роздільної здатності та їх подальшого злиття в одне зображення високої чіткості. Досліджуються різні методи вирівнювання та злиття, що дозволяють ефективно об'єднати інформацію з кількох кадрів та отримати високоякісний результат з покращеною деталізацією.

Робота [76] присвячена вирішенню проблеми розмиття зображення, що виникає внаслідок швидкого руху об'єктів у відео. Автори пропонують використовувати статистичну модель на основі бінарної прихованої змінної для аналізу різкості кожного пікселя в кадрі низької роздільної здатності. Ця модель визначає "корисність" кожного пікселя - наскільки чітким та інформативним він є. В процесі підвищення роздільної здатності алгоритм надає більшу вагу чітким пікселям, мінімізуючи вплив розмитих ділянок на кінцевий результат. Такий підхід дозволяє отримувати більш чіткі та деталізовані зображення навіть за умов швидкого руху в кадрі.

1.4 Висновки за розділом 1

У першому розділі було проведено всебічний аналіз концепції суперроздільності та її сучасного стану. Зокрема, визначено основні поняття суперроздільності, а також розглянуто актуальні проблеми та задачі, які стоять перед дослідниками у цій галузі. Окрім того, виконано детальний огляд відомих методів підвищення роздільної здатності.

Розглянуто як однокадрові, так і багатокадрові алгоритми надроздільної здатності. Особливу увагу було приділено однокадровим методам, включаючи класичні підходи та сучасні методи на основі глибокого навчання. У рамках цього аналізу детально описано та досліджено основні алгоритми, такі як

SRCNN, VDSR, SRGAN та EDSR. Для кожного з них були розглянуті їхні архітектурні особливості, а також області застосування.

Також було розглянуто багатокадрові алгоритми надроздільної здатності, які використовують інформацію з декількох кадрів для підвищення якості зображення. Ці методи показали свою ефективність у випадках, коли доступно кілька зображень однієї сцени.

Окремо було висвітлено алгоритми надвисокої роздільної здатності відеоданих, які спрямовані на покращення якості відео. Ці методи розглядають відео як послідовність кадрів, враховуючи тимчасову інформацію для досягнення високої якості відтворення.

Таким чином, у першому розділі виконано глибокий теоретичний аналіз методів суперроздільності, що закладає основу для подальших практичних досліджень та порівняння їхньої ефективності в умовах низької роздільної здатності.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування

При розробці програмного забезпечення для реалізації та оцінки цільових моделей надвисокої роздільної здатності було вибрано мову програмування Python.

Python - це високорівнева, універсальна, динамічно типізована та інтерпретована мова програмування. Частина визначення «інтерпретована» означає, що Python (та інші інтерпретовані мови) не компілюються безпосередньо в машинну мову або інструкції перед виконанням на комп'ютері. Замість цього Python зчитується і виконується іншою програмою (або інтерпретатором), яка потім перекладає код на «машинну мову», зрозумілу процесору комп'ютера [77].

Існує ряд переваг і недоліків мови програмування, що підпадає під категорію інтерпретованих. Найголовніше, що інтерпретовані мови програмування, такі як Python, не залежать від платформи, тобто вони можуть працювати на будь-якій операційній машині, системі або платформі. Ці типи мов кодування також зазвичай менші за розміром і мають такі особливості, як динамічна типізація [77].

Не заглиблюючись у деталі, динамічна типізація означає, що типи даних перевіряються під час виконання, а отже, тип змінної може змінюватися протягом її життя [77].

Високорівнева частина структури Python стосується того, як Python використовує елементи природної мови - або, точніше, його синтаксис легко читається і є англоподібним за своєю природою. Неважко поглянути на рядок коду на Python і точно знати, що він мав робити, на відміну від інших мов, таких як C, де намір не завжди легко зрозуміти. Мови програмування високого рівня, такі як Python, простіші для розуміння, автоматизують такі завдання, як керування пам'яттю та інші низькорівневі комп'ютерні процеси, і, як правило, мають багато абстракцій від комп'ютерної системи [77].

Абстрагування, по суті, означає рівень, на якому програміст може взаємодіяти з архітектурою комп'ютера. Наприклад, програміст на Python не може повністю оптимізувати свої програми, використовуючи конкретне комп'ютерне обладнання. Ця відсутність контролю відома як штраф за абстракцію і є ціною, яку програміст високого рівня платить за зручності, які пропонує мова (такі як багаторазове використання коду, спільні функції, налагодження коду на високому рівні та незалежність від платформи) [77].

Розробка програмного забезпечення мовою Python має безліч переваг. Нижче ми виділимо основні переваги з коротким поясненням переваг вибору Python для кожної з них [77].

Python сама по собі є потужною мовою програмування, але в ній завжди є місце для вдосконалення. Разом з тим, Python є дуже розширюваною мовою, що означає, що ви можете розширювати Python до інших мов і писати C#, C, C++, Java та інші мови програмування у вашій кодовій базі Python, щоб додати функціональності або поєднати його з іншим програмним забезпеченням [77].

Говорячи про додавання функціональності до Python, однією з головних переваг вибору Python як мови програмування (або навіть додавання її як другої чи третьої мови) є велика кількість бібліотек, які ви можете включити у свій код. Бібліотека - це, по суті, фрагмент коду, який можна використовувати для виконання загальних функцій, таких як маніпулювання зображеннями, виконання рівнянь, адміністрування баз даних, регулярні вирази тощо [77].

Бібліотеки допомагають зменшити кількість помилок у кодуванні, підвищити ефективність роботи програмістів та зменшити розмір програмного забезпечення (і кількість рядків коду). Замість того, щоб писати довгі рядки коду для виконання звичайного завдання, програмісти можуть просто звернутися до бібліотеки - часто з одним рядком коду - і виконати це завдання замість них [77].

Зауважте, що коли ми вживаємо слово «бібліотека», ми використовуємо його як синонім модулів та пакетів [77].

Без сумніву, однією з найважливіших переваг Python є його величезна спільнота розробників та інженерів програмного забезпечення. Ця спільнота

допомагає підтримувати Python, а отже і розробників Python, у багатьох напрямках. Для початку, вони допомагають мові рости та розширюватися, оскільки нові ітерації мови програмування покладаються на підтримку спільноти для додавання функціональності та пошуку недоліків у нових версіях. Вони також активно долучаються до створення навчальних ресурсів, таких як підручники, посібники, книги, дискусії на форумах та відеоконтент Python [77].

Крім того, спільнота Python створює бібліотеки та плагіни, які розширюють можливості мови. Оскільки спільнота Python така велика, вона також гарантує, що мова продовжує існувати, а компанії продовжують адаптувати Python до своїх стеків та технологій розробників. Це означає більше робочих місць для Python і більше користувачів, які купуватимуть програмне забезпечення Python, яке ви розробляєте [77].

Python неймовірно простий у вивченні, незалежно від вашого віку чи рівня досвіду. Оскільки ця мова дуже проста для читання і написання, вона також є чудовою другою або третьою мовою, яку можна додати до свого резюме розробника [77].

Навіть якщо програміст планує кодувати іншою мовою в майбутньому, Python - це гарне місце для початку. Ви можете вивчити основні філософії та концепції програмування, які перетинаються незалежно від того, яку мову ви використовуєте. Наприклад, майже кожна мова програмування використовує поняття змінних, типів даних, циклів та ітерацій, а також керуючих операторів, таких як `if` та `else` [77].

Обґрунтування вибору мови програмування для створення моделей глибокого навчання наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для розробки програмного забезпечення для розпізнавання графічних образів

МП	Переваги	Недоліки
Python	Простота і легкість вивчення. Широкий вибір бібліотек для машинного навчання і обробки зображень (TensorFlow, PyTorch, OpenCV). Велика спільнота і велика кількість документації. Хороша підтримка кросплатформенної розробки. Інтерактивні середовища розробки (Jupyter, Spyder).	Менша продуктивність порівняно з мовами, що компілюються. Іноді виникають проблеми з керуванням пам'яттю. Може бути повільнішим для великих обчислювальних завдань. Проблеми з версійною сумісністю бібліотек (Python 2 vs. Python 3). Динамічна типізація може призводити до помилок, що виявляються тільки в runtime.
R	Сильні статистичні та графічні можливості. Хороша підтримка для аналізу даних і візуалізації. Великі пакети для обробки зображень і машинного навчання (caret, mlr). Підтримка інтерактивних візуалізацій (Shiny, ggplot2). Активна спільнота статистиків і аналітиків даних. Легкість інтеграції з іншими мовами (Python, C++).	Менш популярний у сфері глибокого навчання. Повільніше порівняно з Python і C++ у деяких завданнях. Складнощі з інтеграцією у виробничі системи. Менше бібліотек для глибокого навчання порівняно з Python.
MATLAB	Простий та інтуїтивно зрозумілий синтаксис. Потужні вбудовані функції для обробки зображень і обчислень. Широкі можливості для візуалізації даних. Чудова документація та підтримка. Багатий набір інструментів для наукових та інженерних завдань.	Дороге програмне забезпечення. Обмежена інтеграція з сучасними бібліотеками глибокого навчання. Обмежена підтримка для масштабованості та розподілених обчислень. Закрита екосистема, залежить від ліцензій MathWorks. Менша продуктивність порівняно з C++ і Julia.
C++	Висока продуктивність і контроль над пам'яттю. Широке застосування в розробці високопродуктивних додатків. Бібліотеки для машинного навчання (Dlib, OpenCV). Можливість інтеграції з іншими мовами. Хороша підтримка паралельних обчислень і багатопоточності. Можливість оптимізації критично важливих ділянок коду.	Складний і багатослівний синтаксис. Більш тривалий час розробки порівняно з інтерпретованими мовами. Високий поріг входу для програмістів-початківців. Вимагає управління пам'яттю вручну, що збільшує ймовірність помилок.

Продовження таблиці 2.1

МП	Переваги	Недоліки
Julia	Висока продуктивність завдяки JIT-компіляції. Простота синтаксису, схожого з MATLAB і Python. Оптимізований для чисельних і наукових обчислень. Сумісність із Python, C і Fortran. Підтримка паралельних і розподілених обчислень. Зростаюча спільнота та активний розвиток.	Молода мова, менше бібліотек і ресурсів порівняно з Python. Менша спільнота та підтримка. Потенційна нестабільність через часті оновлення. Можливі проблеми з інтеграцією в наявні проєкти.

Отже, для реалізації ПЗ було обрано мову Python, тому що це зручна, високорівнева мова програмування, яка легко інтерпретується і чудово підходить для різноманітних завдань, у тому числі для аналізу даних та штучного інтелекту.

2.2 Вибір середовища розробки реалізації моделей надвисокої роздільної здатності

При розробці ПЗ для було використано обрано IDE під назвою Visual Studio Code.

Visual Studio Code (VS Code) - це редактор коду з сімейства Microsoft. Він безкоштовний у використанні, має відкритий вихідний код і сумісний з операційними системами Windows, Linux та macOS. VS Code належить до тієї ж категорії, що й Atom, Sublime та Text Wrangler, але має більш розширені можливості та багато регулярно оновлюваних розширень [78].

Наприклад, на відміну від Visual Studio, яка є IDE від Microsoft для програм, вебсторінок, вебдодатків, мобільних додатків і вебсервісів, VS Code пропонує такі функції, як написання, редагування та усунення несправностей коду в одному місці [78].

У той час як Visual Studio покладається на платформи Microsoft для розробки програмного забезпечення, такі як Windows API, Windows Presentation Foundation, Windows Forms, Microsoft Silverlight і Windows Store, VS Code працює незалежно на всіх платформах [78].

Незважаючи на безліч розширень, VS Code не займає багато місця на диску, тому для його завантаження вам знадобиться менше 200 МБ вільного місця. Його простий користувацький інтерфейс є додатковою перевагою, а акуратно розділений редактор значно полегшує кодування та усунення несправностей [78].

VS Code постачається з вбудованою підтримкою JavaScript, Node.js та TypeScript. Якщо ви працюєте з іншими мовами програмування, ви можете отримати необхідну підтримку, встановивши відповідні розширення для таких мов, як Python, C#, PHP, Rust та сотні інших [78].

Текстовий менеджер швидко визначить мову за розширенням файлу та його вмістом. Він виділить синтаксис, щоб представити його у вигляді коду, а для деяких мов є можливість ввести додатковий код і пропозиції щодо того, як повинен виглядати кінцевий результат [78].

Кількість доступних розширень для VS Code величезна. Є масивні і складні розширення, такі як підтримка C/C++, що дозволяє переглядати код і виділяти синтаксичні помилки. Але є й прості, наприклад, таке, що дозволить вам автоматично вставляти закриваючий HTML-тег після початкового [78].

Найважливішим є те, що користувачі можуть розробляти власні розширення для коду Visual Studio. Таким чином, розширення стають доступними для інших користувачів, які можуть спробувати їх і залишити відгуки [78].

На додаток до функцій редагування, користувачі також отримують вбудовану консоль. Немає необхідності тримати вікно терміналу окремо, тому розробники програмного забезпечення можуть використовувати унікальну герметичну консоль під час кодування [78].

Інтегрована підтримка Git'a стане в нагоді всім, хто звик до графічного інтерфейсу в системах контролю версій. Вивчення відмінностей та усунення помилок у з'єднанні відбувається з окремого вікна і допомагає розробникам програмного забезпечення вирішувати повсякденні проблеми, з якими вони стикаються у своїй роботі [78].

Крім того, фахівці з обробки даних можуть використовувати його, оскільки він підтримує Jupiter Notebook для візуалізації даних, а також популярні фреймворки для машинного навчання, такі як TensorFlow і PyTorch. Інші функції завжди можна додати за допомогою сторонніх розширень [78].

Microsoft також розробила окремий пакет розширень для системних і хмарних розробників, що дозволяє їм встановлювати SSH-з'єднання з віддаленими машинами і Docker-середовищами [78].

Нарешті, існують офіційні розширення, що дозволяють працювати з різними хмарними провайдерами, тому розробники можуть працювати з сервісами, пов'язаними з платформами AWS, Azure або Google Cloud, використовуючи VS Code і доставляти свої рішення таким чином [78].

На додаток до всього вищесказаного, VS Code має відкритий вихідний код. Він вільний у використанні, тому розробники, які працюють у великих корпораціях, можуть вільно ним користуватися. Той факт, що він безкоштовний, мотивує багатьох користувачів робити внесок у базу даних коду та розширень і брати активну участь у його розвитку. Саме ось чому VS Code стає сильнішим з кожним днем [78].

Обґрунтування вибору середовища розробки для реалізації та оцінки цільових моделей надвисокої роздільної здатності наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору середовища розробки для створення програмного забезпечення для реалізації та оцінки цільових моделей надвисокої роздільної здатності

Середовище розробки (IDE)	Переваги	Недоліки
1	2	3
Visual Studio Code	Легкий і розширюваний. Підтримка безлічі мов та інструментів через розширення. Інтеграція з Git та іншими системами контролю версій. Хороша підтримка Python (через розширення). Інтеграція з Jupyter Notebook через розширення. Безкоштовний і з відкритим вихідним кодом.	Потребує налаштування та встановлення розширень для повної функціональності. Менше специфічних функцій для наукових обчислень порівняно зі спеціалізованими IDE. Можуть виникати проблеми з продуктивністю під час роботи з великими проєктами.

Продовження таблиці 2.2

1	2	3
PyCharm	Спеціалізований для Python. Потужні інструменти для розробки та налагодження. Підтримка наукових пакетів та інтеграція з Jupyter Notebook. Вбудовані інструменти для тестування та рефакторингу. Зручна навігація по коду і автодоповнення. Інструменти для роботи з базами даних.	Платна версія для повного набору функцій (існує безкоштовна версія Community). Більш важка і вимоглива до ресурсів порівняно з VS Code. Повільний час запуску порівняно з іншими IDE.
Jupyter Notebook	Ідеальний для даних і аналізу. Підтримка живих кодових осередків з інтерактивним виведенням. Зручний для візуалізації та експериментів. Легкість у використанні та обміні результатами. Можливість інтеграції з іншими мовами (R, Julia). Чудово підходить для створення навчальних матеріалів і презентацій.	Обмежені можливості для розробки великих проєктів. Менше інструментів для налагодження та управління кодом. Іноді можуть виникати проблеми з продуктивністю на великих даних. Не підходить для складних проєктів з великою кількістю файлів.
MATLAB	Потужні вбудовані функції для обробки зображень і наукових обчислень. Інтуїтивно зрозумілий інтерфейс. Хороша інтеграція з інструментами MATLAB. Широкі можливості для візуалізації даних. Велика документація та приклади. Відмінна підтримка математичних та інженерних завдань.	Дороге програмне забезпечення. Обмежена підтримка інших мов і сучасних бібліотек глибокого навчання. Замкнута екосистема, складність інтеграції з іншими інструментами. Менше гнучкості в налаштуванні середовища порівняно з відкритими IDE.
Spyder	Спеціалізований для наукових обчислень з Python. Інтеграція з IPython та Jupyter. Потужні інструменти для налагодження та аналізу. Інтуїтивно зрозумілий інтерфейс і легкість у використанні. Панель змінних для відстеження даних у реальному часі. Підтримка плагінів і розширень.	Менше можливостей для розробки вебдодатків та інших типів проєктів порівняно з VS Code. Менше розширень і плагінів порівняно з VS Code. Іноді виникають проблеми з продуктивністю та стабільністю. Повільний розвиток порівняно з іншими IDE.

2.3 Вибір фреймворку

PyTorch - це потужна бібліотека глибокого навчання, яка пропонує ряд функцій, що роблять її привабливим вибором для дослідників та практиків. Однією з ключових особливостей PyTorch є його здатність працювати з динамічними обчислювальними графами. На відміну від статичних графових

фреймворків, де структура графа фіксована і визначається перед виконанням, PyTorch дозволяє визначати та модифікувати обчислювальні графи "на льоту" під час виконання. Це забезпечує більшу гнучкість та можливість адаптувати модель до різних сценаріїв, що особливо корисно при роботі зі складними та мінливими архітектурами нейронних мереж [79].

Іншою важливою функцією PyTorch є автоматичне диференціювання. Завдяки вбудованому пакету Autograd, PyTorch автоматично обчислює градієнти, необхідні для навчання моделі, використовуючи алгоритми оптимізації. Це позбавляє від необхідності вручну реалізовувати складні алгоритми зворотного поширення помилки та спрощує процес оптимізації моделі. Autograd забезпечує ефективне та прозоре обчислення градієнтів, що дозволяє зосередитися на розробці та експериментуванні з різними архітектурами моделей [79].

PyTorch також тісно інтегрується з екосистемою Python, що є однією з причин його популярності серед спільнот машинного навчання та науки про дані. Завдяки вбудованій підтримці Python, PyTorch дозволяє легко використовувати існуючі бібліотеки та інструменти, такі як NumPy, SciPy та Matplotlib. Це спрощує процес попередньої обробки даних, візуалізації результатів та інтеграції з іншими компонентами проєкту. Крім того, знайомий синтаксис Python робить код більш читабельним та зрозумілим, що полегшує співпрацю та обмін знаннями між дослідниками [79].

Нарешті, PyTorch пропонує виробниче середовище через TorchScript - високопродуктивне середовище для серіалізації та виконання моделей PyTorch. За допомогою TorchScript можна легко компілювати моделі PyTorch у портативний формат проміжного представлення (IR), який можна розгортати на різних платформах і пристроях без необхідності оригінального коду Python. Це дозволяє безшовно переносити моделі з дослідницького середовища в продуктивні системи, забезпечуючи ефективне виконання та оптимізацію для конкретних апаратних конфігурацій [79].

Обґрунтування вибору фреймворку для створення програмного забезпечення для реалізації та оцінки цільових моделей надвисокої роздільної здатності наведено у таблиці 2.3.

Таблиця 2.3 – Обґрунтування вибору фреймворку для створення програмного забезпечення для реалізації та оцінки цільових моделей надвисокої роздільної здатності

Фреймворк	Переваги	Недоліки
PyTorch	Інтуїтивний і «пітоницький» синтаксис. Динамічне обчислення графа (eager execution) для зручності налагодження. Широка підтримка спільнотою і велика документація. Підтримка для досліджень і прототипування. Пряма взаємодія з NumPy та інтеграція з бібліотеками Python	Менша продуктивність порівняно з TensorFlow для деяких завдань.
TensorFlow	Висока продуктивність і масштабованість. Підтримка як статичного, так і динамічного графів обчислень. Широке застосування в промисловості та підтримка Google. TensorFlow Serving для розгортання моделей. Багатий набір інструментів і бібліотек (TensorBoard для візуалізації, TensorFlow Lite для мобільних пристроїв)	Складніший синтаксис порівняно з PyTorch. Початкове навчання може бути важчим через статичний граф (до появи TensorFlow 2.0)
Keras	Простота та зручність використання. Високорівневий API для швидкого прототипування. Модульність і розширюваність. Підтримка декількох бекендів (TensorFlow, Theano, Microsoft Cognitive Toolkit). Широка спільнота і велика кількість ресурсів	Менш гнучкий порівняно з низькорівневими фреймворками (як PyTorch і TensorFlow). Продуктивність залежить від обраного бекенда. Обмежена підтримка для складних кастомних моделей
Caffe	Висока продуктивність і ефективність. Оптимізовано для розгортання і роботи у виробничому середовищі. Підтримка як CPU, так і GPU. Хороша документація та приклади для класичних завдань комп'ютерного зору	Менш інтуїтивний синтаксис порівняно з PyTorch і TensorFlow. Менше можливостей для динамічного обчислення графів. Менша спільнота та підтримка порівняно з PyTorch і TensorFlow
MXNet	Висока продуктивність і масштабованість. Підтримка як статичного, так і динамічного графів обчислень. Офіційний фреймворк для глибокого навчання від Amazon Web Services (AWS). Легкість інтеграції з іншими мовами (Scala, R, Julia). Підтримка для розгортання на хмарних платформах	Менша спільнота та ресурси порівняно з PyTorch і TensorFlow. Початкове навчання може бути важчим через менш інтуїтивний синтаксис

2.4 Підсумки за розділом 2

У цьому розділі ретельно розглянуто програмну екосистему, найбільш придатну для реалізації та оцінки цільових моделей надвисокої роздільної здатності. Враховуючи обчислювальну інтенсивність глибинного навчання, ключовою вимогою було визначення інструментів, які могли б легко впоратися з вимогами навчання і тестування складних нейронних мереж.

Після оцінки різних мов програмування очевидним вибором став Python. Низький бар'єр для входження в мову в поєднанні з багатою екосистемою бібліотек роблять її ідеальною для завдань глибокого навчання на основі зображень.

Для самого фреймворку машинного навчання було обрано PyTorch. Його широке використання в дослідницькій спільноті, гнучкий API та надійні можливості прискорення графічного процесора особливо добре підходять для експериментів та оцінки продуктивності найсучасніших моделей надвисокої роздільної здатності.

Для забезпечення спрощеного та ефективного робочого процесу розробки було обрано Visual Studio Code як основне середовище розробки. Інтегровані засоби налагодження, інтеграція з Git'ом та підтримка віртуальних середовищ стануть неоціненними помічниками в процесі розробки та аналізу. і усуває потребу в локальному налаштуванні обладнання, що дає змогу зосередитися на основних дослідницьких цілях.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Структура програмного забезпечення

В результаті аналізу предметної області, описаного в попередніх розділах було виділено ряд сутностей: «Net», «ResnetBlock», «PixelShuffleBlock», «UpsampleBlock», «ResidualBlock», «Generator», «Discriminator», «SRCNNTrainer», «VDSRTrainer», «EDSRTrainer», «SRGANTrainer», «DatasetFromFolder».

Кожен із зазначених компонентів був реалізований як самостійний клас. Такий підхід свідчить про те, що створена програма слідує принципам об'єктно-орієнтованого програмування (ООП). В рамках цієї парадигми, всі складові частини програми представлені у вигляді об'єктів, які взаємодіють між собою. Кожен об'єкт, в свою чергу, є instance (примірником) певного класу, що визначає його властивості та поведінку.

Структуру класів програми, їхні методи та атрибути, представлено на рис. 3.1.

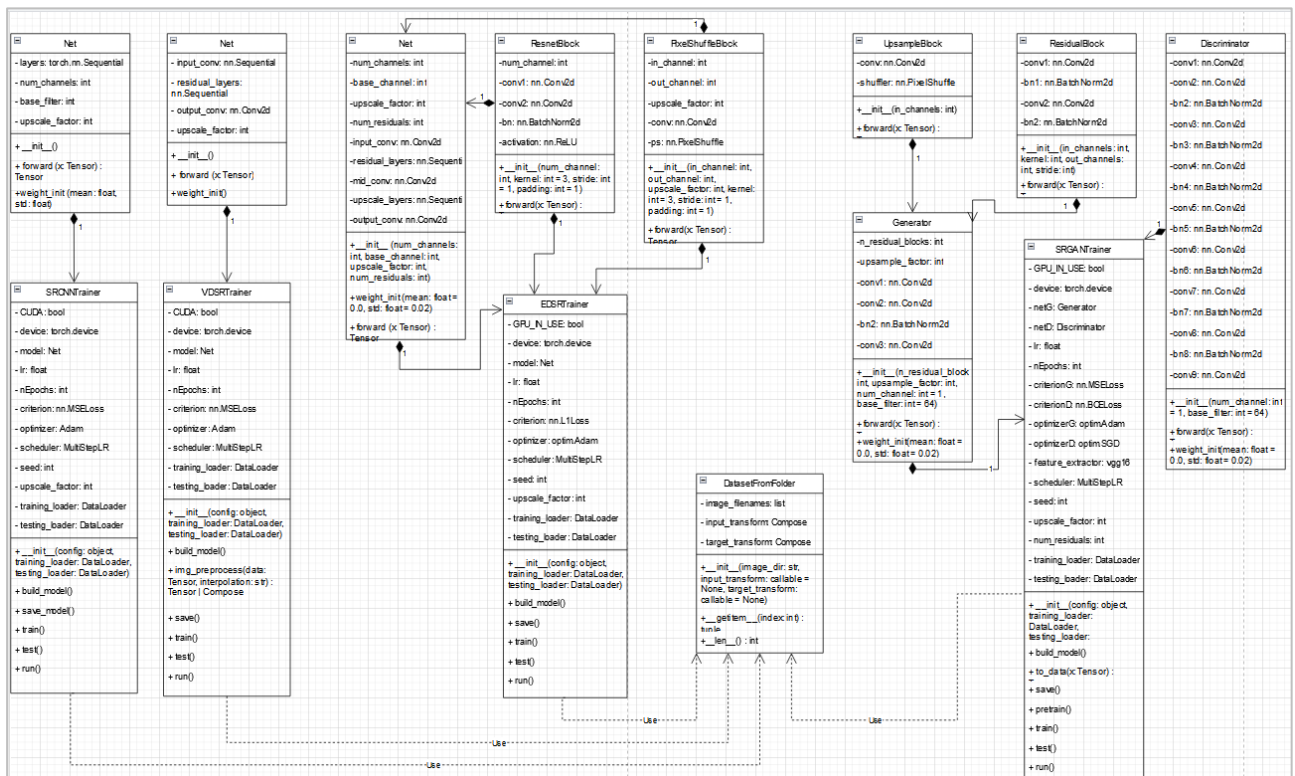


Рисунок 3.1 – Структура класів програми

За своєю суттю система поділяється на кілька ключових компонентів: обробка даних, архітектури моделей, будівельні блоки та тренувальні класи. Кожен компонент відіграє важливу роль у забезпеченні ефективного та результативного навчання цих моделей.

Процес починається із завантаження та попередньої обробки даних, яку виконує клас `DatasetFromFolder`. Цей клас зчитує зображення з вказаного каталогу, застосовуючи необхідні перетворення для підготовки даних до навчання та оцінювання. Ці перетворення мають вирішальне значення, оскільки вони забезпечують правильний формат і масштаб даних для ефективної обробки нейромережевими моделями.

Для моделей SRCNN і VDSR були створені спеціальні класи мереж (клас `Net`). Ці класи інкапсулюють специфічні архітектури цих моделей. Наприклад, клас `VDSRNet` включає шари для вхідної згортки, кілька залишкових шарів і шар вихідної згортки.

На противагу цьому, моделі EDSR та SRGAN вимагають абсолютно різних і ширших архітектур, порівняно з SRCNN та VDSR, що зумовлює необхідність створення унікальних класів, пристосованих до специфіки кожної моделі. Наприклад, модель EDSR використовує складні будівельні блоки, такі як `ResNetBlock` та `PixelShuffleBlock`, які сприяють більш складним методам масштабування. Ці блоки визначають шари та операції в моделях, такі як згортка шарів, пакетна нормалізація та перемішування пікселів для масштабування зображень. Модульна конструкція дозволяє повторно використовувати ці блоки та комбінувати їх у різних конфігураціях, підтримуючи створення складних нейромережових архітектур.

Навчання цих моделей організовано за допомогою класів-тренерів: `SRCNNTrainer`, `VDSRTrainer`, `EDSRTrainer` та `SRGANTrainer` – для кожної моделі по одному класу (тренеру). Кожен тренер відповідає за певний тип моделі і керує процесом навчання, включаючи налаштування CUDA-пристроїв для прискорення GPU, визначення функцій втрат та реалізацію алгоритмів оптимізації, таких як Adam та SGD. Тренери також займаються плануванням, що

допомагає регулювати швидкість навчання під час навчання для покращення збіжності.

3.2 Функціонування програмного забезпечення для побудови та навчання моделей для задач надвисокої роздільної здатності зображень

Функціональна схема ПЗ для побудови та навчання моделей для задач надвисокої роздільної здатності зображень наведена на рис. 3.2.

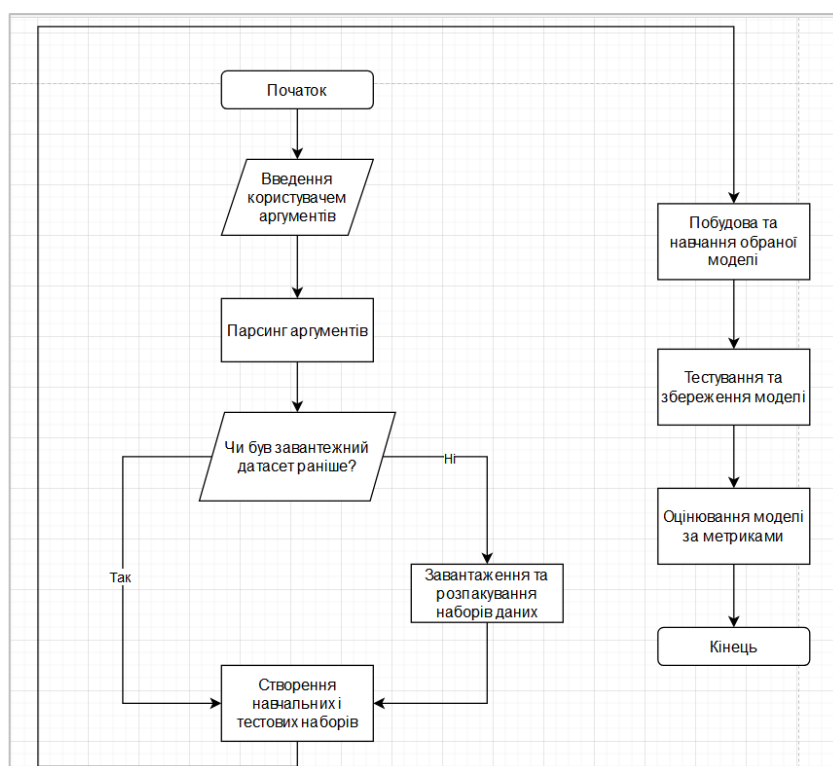


Рисунок 3.2 – Функціональна схема ПЗ

Програма починається з того, що користувач ініціює процес, надаючи необхідні вхідні аргументи. Ці вхідні дані можуть включати такі деталі, як тип моделі, яку потрібно навчити, конкретні гіперпараметри для процесу навчання, розташування наборів даних та інші відповідні налаштування конфігурації. Після того, як користувач вводить ці аргументи, програма починає їх аналізувати, переконуючись, що всі необхідні параметри правильно інтерпретовані і готові до використання на наступних кроках.

Потім програма перевіряє, чи був раніше завантажений набір даних. Це дуже важливий крок, оскільки він запобігає зайвому завантаженню та економить час. Якщо набір даних не було завантажено, програма перейде до завантаження та розпакування набору даних. Після того, як набір даних буде доступним, програма створить навчальний і тестовий набори. Це передбачає відповідне розділення набору даних, щоб забезпечити наявність у моделі як даних для навчання, так і даних для оцінювання.

Після підготовки наборів даних програма переходить до побудови та навчання обраної моделі. Залежно від вхідних аргументів, програма буде архітектуру моделі, ініціалізує її параметри і починає процес навчання. Під час навчання модель навчається на навчальному наборі даних, коригуючи свої ваги та зміщення, щоб мінімізувати функцію втрат. Цей процес є ітеративним і триває протягом певної кількості епох, протягом яких періодично оцінюється ефективність моделі на тестовому наборі даних.

Після завершення навчання програма тестує навчену модель на тестовому наборі даних, щоб оцінити її ефективність. На цьому етапі модель використовується для прогнозування тестових даних і порівняння цих прогнозів з фактичними значеннями для обчислення метрик ефективності. Ці показники дають розуміння того, наскільки добре модель узагальнює нові, небачені дані.

Після тестування програма зберігає навчену модель для подальшого використання. Це гарантує, що навчену модель можна завантажувати і використовувати без необхідності перенавчання з нуля. Нарешті, програма оцінює модель за допомогою різних метрик, щоб оцінити її ефективність та результативність у виконанні завдань надвисокої роздільної здатності.

Таким чином, програма дотримується структурованого робочого процесу, починаючи від введення даних користувачем, обробки наборів даних, побудови та навчання моделі, до оцінки та збереження навченої моделі. Такий системний підхід гарантує, що кожен крок виконується ефективно, в результаті чого ми отримуємо добре навчену модель, готову до практичного застосування в задачах надвисокої роздільної здатності зображень.

3.3 Висновки за розділом 3

У результаті було розроблено комплексне програмне рішення для побудови та навчання моделей для задач надвисокої роздільної здатності зображень. Ця програма ефективно керує всім робочим процесом, починаючи від попередньої обробки даних і побудови моделі до етапів навчання та оцінювання. Реалізація передбачає використання спеціалізованих класів і методів, які були ретельно відібрані для виконання функціональних вимог різних архітектур моделей.

Дизайн включає в себе надійну та модульну структуру, яка дозволяє легко інтегрувати та замінювати різні моделі. Ця гнучкість гарантує, що програмне забезпечення може адаптуватися до різних методів надвисокої роздільної здатності без значних модифікацій.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

Цей розділ буде присвячений опису процесу тестування розробленого програмного забезпечення. Будуть визначені метрики, що використовуються для оцінки точності класифікації. Окремо будуть проаналізовані отримані результати тестування.

4.1 Звернення до програми

Щоб ініціювати виконання програми, користувач повинен запустити основний скрипт "main.py", розташований у кореновому каталозі програми. Це можна зробити за допомогою інтерфейсу командного рядка операційної системи, що надає користувачеві гнучкість і контроль над параметрами виконання.

Під час запуску програма приймає низку аргументів командного рядка, які визначають різні аспекти її роботи. Ці аргументи дозволяють користувачеві точно налаштувати гіперпараметри, пов'язані з навчанням, такі як розмір партії, кількість епох і швидкість навчання.

Крім того, користувач може вказати бажаний коефіцієнт масштабування для надвисокої роздільної здатності і вибрати одну з низки попередньо встановлених моделей (якщо є), пристосовуючи процес покращення зображення до своїх конкретних вимог.

На додаток до основного навчального сценарію, програмне забезпечення надає окремий файл "super_resolve", призначений для застосування попередньо навчених моделей для покращення окремих зображень. Цей скрипт вимагає від користувача вказати шляхи до вхідного зображення, файлу навченої моделі та бажаного місця виводу за допомогою аргументів командного рядка (або відредагувати відповідні змінні у файлі).

Після виконання, "super_resolve" завантажує вказану попередньо навчену модель і застосовує її до вхідного зображення, генеруючи вихідне зображення (аналог), але вже з надвисокою роздільною здатністю.

4.2 Вхідні дані для тестування

У проєкті використовується набір даних BSD100 [80], який є частиною Berkeley Segmentation Dataset and Benchmark (BSDS [81]). Цей набір даних є основою для досліджень в галузі обробки зображень, особливо для таких завдань, як надвисока роздільна здатність, завдяки колекції різноманітних природних зображень. Він забезпечує надійний еталон для оцінки продуктивності різних алгоритмів надвисокої роздільної здатності.

У цьому проєкті набір даних проходить кілька етапів попередньої обробки, щоб підготувати його до використання у навчанні та оцінюванні моделей надвисокої роздільної здатності. Спочатку зображення завантажуються і конвертуються в колірний простір YCbCr. Цей колірний простір обрано тому, що канал яскравості (Y), який містить більшу частину структурної інформації зображення, є найбільш релевантним для задач надвисокої роздільної здатності. Канали кольоровості (Cb і Cr) не використовуються, фокусуючи модель на основних деталях, що передаються в каналі яскравості.

Для забезпечення сумісності з моделями та збереження цілісності зображень під час процесу надвисокої роздільної здатності, зображення обрізаються до потрібного розміру. Це обрізання гарантує, що розміри зображень діляться на коефіцієнт збільшення. Після обрізання зображення змінюють розмір до нижчої роздільної здатності, щоб створити вхідні зображення для моделей. Оригінальні зображення з високою роздільною здатністю слугують цільовими вихідними даними.

Для керування завантаженням і перетворенням зображень використовується спеціальний клас PyTorch Dataset, DatasetFromFolder. Цей клас ініціалізується з директорії, що містить зображення, і застосовує необхідні

перетворення як до вхідних, так і до цільових зображень. Метод `__getitem__` завантажує зображення, застосовує перетворення і повертає пару, що складається з вхідного зображення з низькою роздільною здатністю і цільового зображення з високою роздільною здатністю. Метод `__len__` надає загальну кількість зображень у наборі даних.

4.3 Оцінювання якості моделей

Для оцінки моделей використовують різні метрики, і, як наслідок, існують різні методи оцінки якості зображення, які дають змогу оцінити різні параметри згенерованого зображення. Усі ці метрики належать до двох категорій: суб'єктивних та об'єктивних.

Суб'єктивні метрики ґрунтуються на суб'єктивній оцінці якості зображення людиною, тоді як об'єктивні метрики використовують обчислювальні моделі для оцінки якості зображення. Хоча суб'єктивні метрики можуть бути більш "точними", вони можуть бути незручними у використанні, трудомісткими і дорогими для обчислення. Окрім того, ці дві категорії метрик не завжди відповідають одна одній.

Однією з часто використовуваних метрик є метрика PSNR. Вона є об'єктивною, і обернено пропорційна логарифму середньої квадратичної помилки, яка розраховується за формулою (4.1), між справжнім зображенням високої роздільної здатності та згенерованим. Сама метрика PSNR розраховується за формулою (4.2) [1]:

$$MSE = \frac{1}{N} \sum_{i=1}^N (I_i - \hat{I}_i)^2, \quad (4.1)$$

де N – загальна кількість спостережень;

Y_i – істинне значення (реальне значення) для i -го спостереження;

$\hat{Y}_i$ – передбачене значення для i -го спостереження.

У цьому контексті I_i і $\hat{I}_i$ можуть представляти значення, такі як цільові змінні та прогнозовані значення в задачах регресії, а N – загальне число пар $(I_i)(\hat{I}_i)$.

$$PSNR = 10 \times \log_{10} \left(\frac{L^2}{MSE} \right), \quad (4.2)$$

де L - максимальне значення пікселя в зображенні. Для 8-бітного зображення це значення зазвичай дорівнює 255;

MSE - середнє квадратичне відхилення, розраховане за наведеною вище формулою.

PSNR вимірюється в децибелах (дБ). Чим вище значення PSNR, тим краща якість реконструйованого зображення. Це тому, що високий PSNR означає, що сигнал оригіналу набагато сильніший за будь-які шуми або спотворення, що виникли під час реконструкції. Іншими словами, реконструйоване зображення з високим PSNR буде ближчим до оригіналу з точки зору візуальної якості.

SSIM (індекс структурної подібності) - це метрика, що використовується для вимірювання схожості між двома зображеннями. На відміну від таких метрик, як MSE і PSNR, які оцінюють абсолютні відмінності між пікселями, SSIM оцінює сприйняття зображень людиною, з огляду на зміни в структурній інформації, а також у яскравості та контрасті. Ця метрика є зваженим добутком яскравості, контрасту і структури зображення, обчислених незалежно один від одного, і розраховується за формулою (4.3):

$$SSIM(I, \hat{I}) = \frac{(2\mu_I \mu_{\hat{I}} + C_1)(\sigma_{I\hat{I}} + C_2)}{(\mu_I^2 + \mu_{\hat{I}}^2 + C_1)(\sigma_I^2 + \sigma_{\hat{I}}^2 + C_2)}, \quad (4.3)$$

де I і $\hat{I}$ – порівнювані зображення;

μ_I і $\mu_{\hat{I}}$ – середні значення зображення;

σ_I і $\sigma_{\hat{I}}$ – стандартні відхилення пікселів (дисперсія) для I та $\hat{I}$ відповідно;
 $\sigma_{I\hat{I}}$ – коваріація між зображеннями;
 C_1 і C_2 – константи для стабілізації.

Найчастіше під час оцінювання користуються обчисленням локальної метрики SSIM, у різних вікнах зображення, за подальшої зваженої оцінки отриманих результатів, для стабілізації процесу оцінювання. Оскільки SSIM є мірою якості реконструкції на основі зорової системи людського ока, вона краще підходить для оцінки сприйняття, ніж інші метрики.

Також для оцінювання моделей використовують такі метрики, як: середня оцінка думок, що є ручним способом визначення результатів моделі, де людей просять оцінити зображення, після чого результати агрегують, і як показник використовують середній результат оцінювання, критерій вірогідності інформації, візуальну достовірність інформації, але нехай ці метрики й існують, ми не будемо їх використовувати, а будемо користуватися об'єктивними метриками, такими як PSNR та SSIM, оскільки вони дозволяють кількісно оцінити якість зображення без суб'єктивних людських упереджень.

4.4 Результати обробки зображень

Вдосконалені версії зображень із вищою роздільною здатністю згенеровані кожною моделлю представлено на рис. 4.1.

Натомість, огляд значень метрик PSNR та SSIM для кожної моделі при використанні різних масштабів підвищення роздільності наведено в таблиці 4.1.



Рисунок 4.1 – Вихідні зображення навчених моделей

Таблиця 4.1 – Загальні результати тестування моделей

Модель	Масштаб	PSNR	SSIM
SRCNN	2x	31.04	0.8667
SRCNN	3x	28.10	0.7713
SRCNN	4x	26.34	0.7015
VDSR	2x	27.46	0.7126
VDSR	3x	25.78	0.6913
VDSR	4x	24.54	0.6604
EDSR	2x	31.89	0.8957
EDSR	3x	29.03	0.8243
EDSR	4x	27.20	0.8011
SRGAN	—	—	—

Результати експериментів показали, що модель SRCNN продемонструвала кращі результати порівняно з VDSR, що, на перший погляд, може видатися несподіваним, оскільки VDSR зазвичай вважається потужнішим методом. Однак існує кілька факторів, які могли вплинути на це. По-перше, якість результату сильно залежить від параметрів навчання та гіперпараметрів моделі. Можливо, обрані параметри для VDSR були неоптимальними, що й призвело до зниження її ефективності. По-друге, VDSR використовує значно глибшу архітектуру порівняно з SRCNN, що робить її більш вразливою до проблем перенапруги або за недостатнього часу навчання.

Що стосується моделі SRGAN, її навчання було неможливим через обмеження оперативної пам'яті. SRGAN вимагає значно більше обчислювальних ресурсів через свою архітектуру, що включає генеративну та дискримінативну мережі. На жаль, нестача оперативної пам'яті на доступному обладнанні не дозволила завершити навчання цієї моделі.

Проте результати, отримані за допомогою реалізованих моделей, показали значне поліпшення якості зображень порівняно з вихідними зображеннями з низькою роздільною здатністю. Модель EDSR також

продемонструвала високі результати, підтверджуючи свою ефективність у завданні надроздільної здатності.

Таким чином, незважаючи на деякі труднощі й обмеження, отримані результати підтвердили ефективність сучасних моделей глибокого навчання для завдання підвищення роздільної здатності зображень. У майбутньому, для поліпшення результатів і подолання поточних обмежень, рекомендується розглянути можливість використання потужнішого обладнання та оптимізації гіперпараметрів моделей.

4.5 Висновки за розділом 4

Практична реалізація та аналіз цих моделей підтвердили їхній потенціал і актуальність у сфері підвищення роздільної здатності зображень. Однак результати також вказали на необхідність ретельного налаштування гіперпараметрів і використання більш потужних обчислювальних ресурсів для досягнення оптимальних результатів. Ці висновки підкреслюють важливість подальших досліджень і розроблення більш ефективних методів і підходів для розв'язання задач надроздільної здатності.

ВИСНОВКИ

З метою реалізації та порівняльного аналізу методів надроздільної здатності зображень у роботі досліджено актуальну задачу підвищення якості зображень із низькою роздільною здатністю.

У роботі реалізовано чотири моделі надроздільної здатності зображень: SRCNN, VDSR, SRGAN і EDSR. Кожну модель було ретельно вивчено та програмно реалізовано з використанням мови програмування Python і фреймворку PyTorch.

Проведене тестування моделей на наборі зображень різного типу та роздільної здатності дозволило оцінити їх ефективність за допомогою стандартних метрик якості зображень, таких як PSNR і SSIM.

Результати показали, що моделі на основі глибокого навчання значно перевершують традиційні методи поліпшення якості зображень. Зокрема, модель EDSR продемонструвала високі показники якості зображень, значно покращуючи їхню деталізацію та чіткість.

Незважаючи на очікування, модель VDSR показала гірші результати порівняно з SRCNN. Це може бути пояснено впливом гіперпараметрів і можливими проблемами перенавчання, які виникають через глибоку архітектуру VDSR. Спроба навчання моделі SRGAN виявилася невдалою через недостатність пам'яті на доступному обладнанні, що підкреслює високі обчислювальні вимоги цієї моделі.

Також було проведено дослідження впливу різних параметрів моделей на якість одержуваних зображень. Результати експериментів дозволили оптимізувати параметри кожної моделі для досягнення найкращих результатів.

Практична значущість роботи полягає в можливості використання розроблених моделей для різноманітних додатків, які потребують поліпшення якості зображень.

Перспективи подальших досліджень охоплюють розроблення й тестування нових моделей надроздільної здатності зображень, оптимізацію

наявних алгоритмів для підвищення їхньої ефективності, а також дослідження методів багатокадрової надроздільної здатності для поліпшення якості відеозаписів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Yang J. Image Super-Resolution: Historical Overview and Future Challenges / J. Yang, T. Huang // Super-Resolution Imaging / P. Milanfar. – Boca Raton, CRC Press, 2010. – ch. 1. – 17 p.
2. Patmawati N. P., Quality Image Enhancement from Low Resolution Camera using Convolutional Neural Network [Text] / N. P. Patmawati, A. Arifianto, K. N. Ramadhani // Information and Communication Technology : seventh international conference, Kuala Lumpur, 24-26 July 2019 : proceedings. – Kuala Lumpur : IEEE, 2019. – P. 1-6.
3. Wang Z. Deep Learning for Image Super-Resolution: A Survey [Text] / Z. Wang, J. Chen, S. C. H. Hoi // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2021. – Vol. 43, Issue 10. – P. 3365-3387.
4. Алгоритм повышения пространственного разрешения изображений на основе генеративно-состязательной сети : выпускная квалификационная работа магистранта (окончат.) : 40-57 / рук. Ю.А. Иванова; исполн.: Т. В. Монгуш. – Томск, 2023. – 14-20 с.
5. Сирота, А. А. Алгоритмы обработки изображений с достижением эффекта сверхразрешения на основе методов оптимальной фильтрации и машинного обучения : дис. ... доктор техн. Наук / Сирота Александр Анатольевич. – Воронеж, 2022. – 17-25 с.
6. Greenspan H. Super-Resolution in Medical Imaging [Text] / H. Greenspan // The Computer Journal. – 2009. – Volume 52, № 1. – P. 43-63.
7. Isaac J. S. Super resolution techniques for medical image processing [Text] / J. S. Isaac, R. Kulkarni // Technologies for Sustainable Development : International Conference, Mumbai, 04-06 February 2015 : proceedings. – Mumbai: IEEE, 2015. – P. 1-6.
8. Huang Y. Simultaneous Super-Resolution and Cross-Modality Synthesis of 3D Medical Images Using Weakly-Supervised Joint Convolutional Sparse Coding [Text] / Y. Huang, L. Shao, A. Frangi // Computer Vision and Pattern Recognition :

2017 IEEE Conference, Honolulu, 21-26 July 2017 : proceedings. – Honolulu: IEEE, 2017. – P. 5787-5796.

9. A super-resolution reconstruction algorithm for surveillance images [Text] / [L. Zhang, H. Zhang, H. Shen et al.] // Signal Processing. – 2010. – Issue 3. – P. 848-859.

10. Pejman R. Convolutional Neural Network Super Resolution for Face Recognition in Surveillance Monitoring [Text] / R. Pejman, U. Tönis, E. Sergio // Articulated Motion and Deformable Objects : Ninth International Conference, Palma de Mallorca, 13-15 July 2016 : proceedings. – Palma de Mallorca: AMDO, 2016. – P. 175–184.

11. Is image super-resolution helpful for other vision tasks? [Text] / [D. Dai, Y. Wang, Y. Chen, et al.] // Applications of Computer Vision : 2016 IEEE Winter Conference, Lake Placid, 07-10 March 2016 : proceedings. – Lake Placid: IEEE, 2016. – P. 1-9.

12. Haris M. Task-driven super resolution: Object detection in low-resolution images [Text] / M. Haris, G. Shakhnarovich, N. Ukita // Neural Information Processing : International Conference, 30 March 2018 : proceedings. – arXiv, 2018. – P. 1-29.

13. Sajjadi M. S. M. EnhanceNet: Single Image Super-Resolution Through Automated Texture Synthesis [Text] / B. Schölkopf, M. Hirsch // Computer Vision : 2017 IEEE International Conference, Venice, 22-29 October 2017 : proceedings. – Venice: IEEE, 2017. – P. 4501-4510.

14. Patmawati N. P. Quality Image Enhancement from Low Resolution Camera using Convolutional Neural Network [Text] / A. Arifianto, K. N. Ramadhani // Information and Communication Technology : 2019 7th International Conference, Kuala Lumpur, 24-26 July 2019 : proceedings. – Kuala Lumpur: IEEE, 2019. – P. 1-6.

15. Deep learning face representation by joint identification-verification [Text] / Y. Sun, Y. Chen, X. Wang, et al.] // Neural Information Processing Systems : 27th International Conference, Montreal, 08 December 2014 : proceedings. – vol. 2 – P. 1988–1996.

16. Schroff F. FaceNet: A unified embedding for face recognition and clustering [Text] / D. Kalenichenko, J. Philbin // Computer Vision and Pattern Recognition : IEEE Conference, Boston, 07-12 June 2015 proceedings. – Boston: IEEE, 2015. – P. 815-823.
17. Parkhi, O.M. Deep Face Recognition [Text] / O.M. Parkhi, A. Vedaldi, A. Zisserman // British Machine Vision Conference : Swansea, 7-10 September 2015 : proceedings. – Swansea: BMVA, 2015: – P. 1-10.
18. Sun Y. Sparsifying Neural Network Connections for Face Recognition [Text] / X. Wang, X. Tang // Computer Vision and Pattern Recognition : IEEE Conference, Las Vegas, 27-30 June 2016 : proceedings. – Las Vegas: IEEE, 2016 – P. 4856-4864.
19. A Discriminative Feature Learning Approach for Deep Face Recognition [Text] / [Y. Wen, K. Zhang, Z. Li et al.] // Computer Vision : ECCV 2016, 16 September 2016 : Springer: ECCV, 2016. – P. 499–515.
20. SphereFace: Deep Hypersphere Embedding for Face Recognition [Text] / W. Liu, Y. Wen, Z. Yu, et al.] // Computer Vision and Pattern Recognition : 2017 IEEE Conference, Honolulu, 21-26 July 2017 : proceedings. – Honolulu: IEEE, 2017. – P. 6738-6746.
21. Herrmann C. Low-resolution Convolutional Neural Networks for video face recognition [Text] / D. Willersinn, J. Beyerer // Advanced Video and Signal Based Surveillance : 13th IEEE International Conference, Colorado Springs, 23-26 August 2016 : proceedings. – Colorado Springs: IEEE, 2016. – P. 221-227.
22. Harris, J. L. Diffraction and Resolving Power [Text] / J. L. Harris // Journal of the Optical Society of America. – 1964. – Vol. 54, № 7. – P. 931-936.
23. Tsai, R. Y. Multiframe Image Restoration and Registration [Text] / R. Y. Tsai, T. S. Huang // Advances in Computer Vision and Image Processing. – Greenwich : JAI Press, 1984. – ch. 3. – 317-339.
24. Super-Resolution Without Explicit Subpixel Motion Estimation [Text] / [H. Takeda, P. Milanfar, M. Protter et al.] // IEEE Transactions on Image Processing. – 2009. – Vol. 18, Issue 9. – P. 1958-1975.

25. Greenspan H. Super-Resolution in Medical Imaging [Text] / H. Greenspan // The Computer Journal. – 2009. – Vol. 52, Issue 1. – P. 43-63.
26. Elad M. A fast super-resolution reconstruction algorithm for pure translational motion and common space-invariant blur [Text] / M. Elad, Y. Hel-Or // Electrical and Electronic Engineers : 21st IEEE Convention, Tel-Aviv, 11-12 April 2000 : proceedings. – Tel-Aviv: IEEE, 2002. – P. 402-405.
27. Super-Resolution Reconstruction Algorithm To MODIS Remote Sensing Images [Text] / [H. Shen, M. K. Ng, Pingxiang Li et al.] // The Computer Journal. – 2009. – Vol. 52, № 1. – P. 90–100.
28. Keren D. Image sequence enhancement using sub-pixel displacements [Text] / D. Keren, S. Peleg, R. Brada // Computer Vision and Pattern Recognition : '88 The Computer Society Conference, Ann Arbor, 05-09 June 1988 : proceedings. – Ann Arbor: IEEE, 1988. – P. 742-746.
29. Kim S. P. Recursive reconstruction of high resolution image from noisy undersampled multiframe [Text] / S. P. Kim, N. K. Bose, H. M. Valenzuela // IEEE Transactions on Acoustics, Speech, and Signal Processing. – 1990. – vol. 38, № 6. – P. 1013-1027.
30. Park S. C. Super-resolution image reconstruction: a technical overview [Text] / S. C. Park, M. K. Park, M. G. Kang // IEEE Signal Processing Magazine. – 2003. – vol. 20, № 3. – P. 21-36.
31. Learning a Deep Convolutional Network for Image Super-Resolution [Text] / [C. Dong, C. C. Loy, K. He et al.] // Computer Vision : 13th European Conference, Zurich, 6 September 2014 : proceedings. – Zurich: ECCV, 2014. – P. 184–199.
32. Second-Order Attention Network for Single Image Super-Resolution [Text] / [T. Dai, J. Cai, Y. Zhang, S.-T. Xia et al.] // Computer Vision and Pattern Recognition : IEEE/CVF Conference, Long Beach, 15-20 June 2019 : : proceedings. – Long Beach: IEEE, 2019. – P. 11065-11074.
33. Single Image Super-Resolution via a Holistic Attention Network [Text] / [B. Niu, W. Wen, W. Ren et al.] // International Conference on Computer Vision : 16th

European Conference, Glasgow, August 23-28, 2020 : proceedings. – Glasgow: IEEE, 2020. – P. 191-207.

34. Pre-Trained Image Processing Transformer [Text] / [H. Chen, Y. Wang, T. Guo et al.] // Computer Vision and Pattern Recognition : 2021 IEEE/CVF Conference, Nashville, June 20-25 2021: proceedings. – Nashville: CVPR, 2020. – P. 12294-12305.

35. SwinIR: Image restoration using swin transformer [Text] / [J. Liang, J. Cao, G. Sun et al.] // Computer Vision Workshops : 2021 IEEE/CVF International Conference, Montreal, 11-17 October 2021: proceedings. – Montreal: IEEE, 2021. – P. 1-12.

36. Shultz, R. R. A Bayesian approach to image expansion for improved definition [Text] / R. R. Schultz, R. L. Stevenson // IEEE Transactions on Image Processing. – 1994. – Vol. 3, Issue 3. – P. 233-242.

37. Hou, H. S. Cubic splines for image interpolation and digital filtering [Text] / H. S. Hou, H. C. Andrews // IEEE Transactions on Acoustics, Speech and Signal Processing. – 1978. – Vol. 26, Issue 6. – P. 508-517.

38. Li, X. New edge-directed interpolation [Text] / X. Li, M. T. Orchard // IEEE Transactions on Image Processing. – 2001. – Vol. 10, Issue 10. – P. 1521-1527.

39. Cha, Y. Edge forming methods for color image zooming [Text] / Y. Cha, S. Kim // IEEE Transactions on Image Processing. – 2006. – Vol. 15, Issue 8. – P. 2315-2323.

40. Timofte, R. A+: Adjusted anchored neighborhood regression for fast super-resolution [Text] / R. Timofte, V. De Smet, L. Van Gool // Computer Vision : 12th Asian Conference, Singapore, November 1-5, 2014 : proceedings. – Singapore: Springer, 2015. – P. 111–126.

41. Schuler, S. Fast and accurate image upscaling with super-resolution forests [Text] / S. Schuler, C. Leistner, H. Bischof // Computer Vision and Pattern Recognition : IEEE Conference, Boston, June 7-12, 2015 : proceedings. – Boston: IEEE, 2015. – P. 1-9.

42. Huang, J.-B. Single image super-resolution using transformed self-exemplars [Text] / J.-B. Huang, A. Singh, N. Ahuja // Computer Vision and Pattern Recognition : IEEE Conference, Boston, June 7-12, 2015 : proceedings. – Boston: IEEE, 2015. – P. 1-9.
43. Freedman, G. Image and video upscaling from local self-examples [Text] / G. Freedman, R. Fattal // ACM Transactions on Graphics. – 2011. – Vol. 30, №2. – P. 1-11.
44. Yang, J. Fast image super-resolution based on in-place example regression [Text] / J. Yang, Z. Lin, S. Cohen // Computer Vision and Pattern Recognition : IEEE Conference, Portland, June 23-28, 2013: proceedings. – Portland: IEEE, 2013. – P. 1059-1066.
45. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network [Text] / [C. Ledig, L. Theis, F. Huszar et al.] // Computer Vision and Pattern Recognition : 2017 IEEE Conference, Honolulu, 21-26 July 2017 : proceedings. – Honolulu: IEEE, 2017. – P. 105-114.
46. Image super-resolution using deep convolutional networks [Text] / C. Dong, C.C. Loy, K. He, et al.] // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2016. – Vol. 38, Issue 2). – P. 295-307.
47. Dong, C. Accelerating the super-resolution convolutional neural network [Text] / C. Dong, C. C. Loy, X. Tang // Computer Vision : 14th European Conference, Amsterdam, October 11–14, 2016 : proceedings. – Amsterdam: arXiv, 2016. – P. 391-407.
48. Kim, J. Accurate Image Super-Resolution Using Very Deep Convolutional Networks [Text] / J. Kim, J. K. Lee, K. M. Lee // Computer Vision and Pattern Recognition : IEEE Conference, Las Vegas, 27-30 June 2016 : proceedings. – Las Vegas: IEEE, 2016. – P. 1646-1654.
49. Gomede E. Exploring the Super-Resolution Convolutional Neural Network (SRCNN): A Milestone in Image Processing [Electronic resource]. – Access mode: <https://medium.com/the-modern-scientist/exploring-the-super-resolution-convolutional-neural-network-srcnn-a-milestone-in-image-f969bd3cd056>.

50. Tsang S-H. Review: VDSR (Super Resolution) [Electronic resource]. – Access mode: <https://towardsdatascience.com/review-vdsr-super-resolution-f8050d49362f>.
51. Gowda R.: SRGAN Paper Explained [Electronic resource]. – Access mode: <https://medium.com/@ramyahrgowda/srgan-paper-explained-3d2d575d09ff>.
52. Cochard D.: EDSR: A Machine Learning Model for Super-resolution Image Processing [Electronic resource]. – Access mode: <https://medium.com/axinc-ai/edsr-a-machine-learning-model-for-super-resolution-image-processing-9deaf36b24ed>.
53. Enhanced Deep Residual Networks for Single Image Super-Resolution [Text] / [B. Lim, S. Son, H. Kim et al.] // Computer Vision and Pattern Recognition Workshops : 2017 IEEE Conference, Honolulu, 21-26 July 2017 : proceedings. – Honolulu: IEEE, 2017. – P. 1-9.
54. Kim, S.P. Recursive high-resolution reconstruction of blurred multiframe images [Text] / S.P. Kim, W.Y. Su // Acoustics, Speech, and Signal Processing : International Conference, Toronto, 14-17 April 1991 : proceedings. – Toronto: IEEE, 2002. – P. 534–539.
55. Nguen, N. An efficient wavelet-based algorithm for image superresolution [Text] / N. Nguyen, P. Milanfar // Image Processing : International Conference, Vancouver, September 10-13, 2000 : proceedings. – Los Alamitos: IEEE, 2000. – P. 351-354.
56. Ji, H. Robust Wavelet-Based Super-Resolution Reconstruction: Theory and Algorithm [Text] / H. Ji, C. Fermüller // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2009. – Vol. 31, Issue 4. – P. 649-660.
57. Ur., H. Improved Resolution from Subpixel Shifted Pictures [Text] / H. Ur, D. Gross // CVGIP: Graphical Models and Image Processing. – 1992. – Vol. 54, Issue 2. – P. 181-186.
58. Bose, N. K. Superresolution and Noise Filtering Using Moving Least Squares [Text] / N. K. Bose, N. A. Ahuja // IEEE Transactions on Image Processing. – 2006. – Vol. 15, Issue 8. – P. 2239-2248.

59. Chantas, G. K. Super-resolution based on fast registration and maximum a posteriori reconstruction [Text] / G. K. Chantas, G. P. Galatsanos, N.A. Woods // IEEE Transactions on Image Processing. – 2007. – Vol. 16, Issue 7. – P. 1821-1830.
60. Tian, J. Stochastic super-resolution image reconstruction [Text] / J. Tian, K.-K. Ma // Journal of Visual Communication and Image Representation. – 2010. – Vol. 21, Issue 3. – P. 232-244.
61. Belekos, S. P. Maximum a Posteriori Video Super-Resolution Using a New Multichannel Image Prior [Text] / S. P. Belekos, N. P. Galatsanos, A. K. Katsaggelos // IEEE Transactions on Image Processing. – 2010. – Vol. 19, Issue 6. – P. 1451-1464.
62. Irani, M. Super resolution from image sequences [Text] / M. Irani, S. Peleg // Pattern Recognition : 10th International Conference, Atlantic City, 16-21 June 1990 : proceedings. – Atlantic City: IEEE, 1990.– P. 115-120.
63. Ji, H. Wavelet-Based Super-Resolution Reconstruction: Theory and Algorithm [Text] / H. Ji, C. Fermüller // Computer Vision : 9th European Conference, Graz, May 7-13, 2006 : proceedings. – Berlin: Springer, 2006. – P. 295-307.
64. Patti, A. J. Superresolution Video Reconstruction with Arbitrary Sampling Lattices and Nonzero Aperture Time [Text] / A. J. Patti, M. I. Sezan, A. M. Tekalp // IEEE Transactions on Image Processing. – 1997. – Vol. 6, Issue 8. – P. 1064-1076.
65. Elad, M. Super-resolution reconstruction of image sequences [Text] / M. Elad, A. Feuer // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 1999. – Vol. 21, Issue 9. – P. 817-834.
66. Newland, C. Time invariant steady-state Kalman filter for image super-resolution [Text] / C. Newland, D. Gray // Image and Vision Computing New Zealand 2005 (IVCNZ05). – 2006. – P. 381-387.
67. Newland, C. Modified Kalman filtering for image super-resolution [Text] / C. Newland, D. Gray, D. Gibbins // Image and Vision Computing New Zealand. – 2006. – P. 79-84.

68. Sajjadi, M. S. M. Frame-Recurrent Video Super-Resolution [Text] / M. S. M. Sajjadi, R. Vemulapalli, M. Brown // Computer Vision and Pattern Recognition : IEEE Conference, Salt Lake City, 18-23 June 2018 : proceedings. – Salt Lake City: IEEE, 2018. – P. 6626-6634.
69. Deep Learning for Multiple-Image Super-Resolution [Text] / [M. Kawulok, P. Benecki, S. Piechaczek, et al.] // IEEE Geoscience and Remote Sensing Letters. – 2019. – Vol. 17, Issue 6. – P. 1062-1066.
70. DeepSUM: Deep Neural Network for Super-Resolution of Unregistered Multitemporal Images [Text] / [A.B. Molini, D. Valsesia, G. Fracastoro, et al.] // IEEE Transactions on Geoscience and Remote Sensing. – 2019. – Vol. 58, Issue 5. – P. 3644-3656.
71. Kim, J. Deeply-Recursive Convolutional Network for Image Super-Resolution [Text] / J. Kim, J. K. Lee, K. M. Lee // Computer Vision and Pattern Recognition : IEEE Conference, Las Vegas, 27-30 June 2016 : proceedings. – Las Vegas: IEEE, 2016. – P. 1637-1645.
72. Liu, C. On Bayesian Adaptive Video Super Resolution / C. Liu, D. Sun // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2014. – Vol. 36, Issue 2. – P. 346-360.
73. Kappeler, A. Video Super-Resolution with Convolutional Neural Networks [Text] / A. Kappeler, S. Yoo, Q. Dai, A.K. Katsaggelos // IEEE Transactions on Computational Imaging. – 2016. – Vol. 2, Issue 2. – P. 109-122.
74. Makansi, O. End-to-End Learning of Video Super-Resolution with Motion Compensation [Text] / O. Makansi, E. Ilg, T. Brox. // Pattern Recognition : 39th German Conference, Basel, September 12-15, 2017 : proceedings. – Springer: Lecture Notes in Computer Science, 2017. – P. 203-214.
75. Real-Time Video Super-Resolution with Spatio-Temporal Networks and Motion Compensation [Text] / [J. Caballero, C. Ledig, A. Aitken et al.] // Computer Vision and Pattern Recognition : 2017 IEEE Conference, Honolulu, July 21-26, 2017 : proceedings. – Honolulu: IEEE, 2017. – P. 2848-2857.

76. Handling motion blur in multi-frame super-resolution [Text] / [Z. Ma, R. Liao, X. Tao et al.] // Computer Vision and Pattern Recognition : 2015 IEEE Conference, Boston, June 7-12, 2015 : proceedings. – Boston: IEEE, 2017. P. 5224-5232.
77. Payne J.: Benefits of Python Programming Language [Electronic resource]. – Access mode: <https://www.developer.com/languages/python/python-benefits/>.
78. Uspenski A.: Why VS Code remains a developer favorite, year after year [Electronic resource]. – Access mode: <https://shiftmag.dev/vs-code-171/>.
79. Start learning PyTorch for Beginners [Electronic resource]. – Access mode: <https://www.geeksforgeeks.org/start-learning-pytorch-for-beginners/#what-is-pytorch>.
80. BSD100 [Electronic resource]. – Access mode: <https://paperswithcode.com/dataset/bsd100>.
81. The Berkeley Segmentation Dataset and Benchmark [Electronic resource]. – Access mode: <https://www2.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/>.

ДОДАТОК А
Технічне завдання

Вступ

Цей документ являє собою технічне завдання на розробку ПЗ для дослідження і порівняння методів поліпшення якості зображень в умовах низької роздільної здатності. Основна увага приділяється аналізу наявних алгоритмів і моделей, таких як SRCNN, VDSR, EDSR і SRGAN, з метою виявлення їхніх переваг і недоліків. Програма призначена для виконання однокадрової надроздільної здатності, а також для проведення порівняльного аналізу отриманих результатів.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Дослідження та програмна реалізація методів поліпшення якості зображень в умовах низької роздільної здатності», затверджене наказом Національного університету «Запорізька політехніка» № 168 від 24 квітня 2024 р.

A.2 Призначення розробки

Програмне забезпечення призначено для виконання однокадрової надроздільної здатності зображень з використанням різних моделей, а також для проведення порівняльного аналізу їхньої ефективності на основі метрик якості, таких як PSNR і SSIM.

A.3 Основні вимоги до програми

A.3.1 Вимоги до функціональних характеристик

Програмне забезпечення забезпечує виконання наступних функцій:

- реалізація та застосування моделей SRCNN, VDSR, EDSR і SRGAN;

- розрахунок і виведення значень метрик PSNR і SSIM для кожної моделі;
- підтримка різних форматів зображень (JPEG, PNG);
- введення зображень з низькою роздільною здатністю і генерація зображень з високою роздільною здатністю;
- перевірка коректності вхідних даних;
- обробка помилок, що виникають під час виконання програми.

А.3.2 Вимоги до надійності

Програма має коректно обробляти помилки введення даних і гарантувати стабільну роботу під час виконання алгоритмів надроздільної здатності. У разі некоректних вхідних даних або інших помилок, програма повинна виводити інформативні повідомлення про помилки.

А.3.4 Умови експлуатації

Для успішного запуску та експлуатації програми необхідно:

- встановлений інтерпретатор Python версії 3.6 або вище;
- доступ до інтернету для завантаження датасету (за необхідності);
- встановлені бібліотеки: PyTorch, NumPy, Scikit-image;
- операційна система: Windows (тестувалося на Windows 10).

А.3.5 Вимоги до складу та параметрів технічних засобів

Мінімальні характеристики ПК для запуску програми:

- процесор: Intel Core i5 або аналогічний;
- оперативна пам'ять: 16 ГБ;
- графічний процесор: NVIDIA з підтримкою CUDA (рекомендується для прискорення роботи нейронних мереж);

- вільний дисковий простір: 500 МБ.

A.4 Вхідні дані до роботи

В якості вхідних даних алгоритм приймає зображення з низькою роздільною здатністю, збережене у форматі JPEG або PNG. За необхідності, користувач може також вказати додаткові параметри для алгоритмів обробки зображень.

На виході алгоритм надає зображення з високою роздільною здатністю, збережене у форматі JPEG або PNG. Крім того, для оцінки якості отриманих зображень алгоритм розраховує значення метрик PSNR (пікове відношення сигнал/шум) та SSIM (структурна подібність) для кожної використаної моделі.

ДОДАТОК Б
Фрагмент тексту програми

Б.1 Текст файла CNNmodel.py

```

import torch
import torch.nn as nn

class Net(torch.nn.Module):
    def __init__(self, num_channels, base_filter, scaling_factor=2):
        super(NeuralNetwork, self).__init__()

        self.layers = torch.nn.Sequential(
            nn.Conv2d(in_channels=num_channels,
out_channels=base_filter,
                    kernel_size=9, stride=1, padding=4, bias=True),
            nn.ReLU(inplace=True),
            nn.Conv2d(in_channels=base_filter,
out_channels=base_filter // 2, kernel_size=1,
bias=True),
            nn.ReLU(inplace=True),
            nn.Conv2d(in_channels=base_filter // 2,
out_channels=num_channels *
                    (scaling_factor ** 2), kernel_size=5, stride=1,
padding=2, bias=True),
            nn.PixelShuffle(scaling_factor)
        )

    def forward_pass(self, input_data):
        output = self.layers(input_data)
        return output

    def initialize_weights(self, mean, std_dev):
        for module in self._modules:
            normal_initialize(self._modules[module], mean, std_dev)

    def normal_initialize(module, mean, std_dev):
        if isinstance(module, nn.ConvTranspose2d) or isinstance(module,
nn.Conv2d):
            module.weight.data.normal_(mean, std_dev)
            module.bias.data.zero_()

```

Б.2 Текст файла CNNtrain.py

```

import torch
import torch.backends.cudnn as cudnn
from math import log10
from SRCNN.model import Net
from __future__ import print_function

class SRCNNTrainer:
    def __init__(self, config, training_loader, testing_loader):
        super(SuperResolutionTrainer, self).__init__()

```

```

self.is_cuda_available = torch.cuda.is_available()
self.device = torch.device('cuda' if self.is_cuda_available else
'cpu')

self.model = None
self.learning_rate = config.lr
self.num_epochs = config.nEpochs
self.criterion = None
self.optimizer = None
self.scheduler = None
self.seed = config.seed
self.upscale_factor = config.upscale_factor
self.training_loader = training_loader
self.testing_loader = testing_loader

def setup_model(self):
    self.model = Net(num_channels=1, base_filter=64,
upscale_factor=self.upscale_factor).to(self.device)
    self.model.weight_init(mean=0.0, std=0.01)
    self.criterion = torch.nn.MSELoss()
    torch.manual_seed(self.seed)

    if self.is_cuda_available:
        torch.cuda.manual_seed(self.seed)
        cudnn.benchmark = True
        self.criterion.cuda()

    self.optimizer = torch.optim.Adam(self.model.parameters(),
lr=self.learning_rate)
    self.scheduler = torch.optim.lr_scheduler.MultiStepLR(self.optimizer, milestones=[50, 75, 100],
gamma=0.5)

def save_trained_model(self):
    model_out_path = "CNN_model.pth"
    torch.save(self.model, model_out_path)
    print("Model saved at: {}".format(model_out_path))

def train_model(self):
    self.model.train()
    total_loss = 0
    for batch_num, (data, target) in enumerate(self.training_loader):
        data, target = data.to(self.device), target.to(self.device)
        self.optimizer.zero_grad()
        loss = self.criterion(self.model(data), target)
        total_loss += loss.item()
        loss.backward()
        self.optimizer.step()
        progress_bar(batch_num, len(self.training_loader), 'Loss:
%.4f' % (total_loss / (batch_num + 1)))

    print("Average Loss: {:.4f}".format(total_loss /
len(self.training_loader)))

def test_model(self):
    self.model.eval()
    avg_psnr = 0

```

```

        with torch.no_grad():
            for batch_num, (data, target) in
enumerate(self.testing_loader):
                data, target = data.to(self.device),
target.to(self.device)
                prediction = self.model(data)
                mse = self.criterion(prediction, target)
                psnr = 10 * log10(1 / mse.item())
                avg_psnr += psnr
                progress_bar(batch_num, len(self.testing_loader), 'PSNR:
%.4f' % (avg_psnr / (batch_num + 1)))

    print("Average PSNR: {:.4f} dB".format(avg_psnr /
len(self.testing_loader)))
    def run(self):
        self.build_model()
        for epoch in range(1, self.nEpochs + 1):
            print("\n■■■ Epoch {} starts:".format(epoch))
            self.train()
            self.test()
            self.scheduler.step(epoch)
            if epoch == self.nEpochs:
                self.save_model()

```

ДОДАТОК В
Слайди презентації

Національний університет «Запорізька Політехніка»

Кафедра програмних засобів

Дипломна кваліфікаційна робота бакалавра

**ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ
ПОЛІПШЕННЯ ЯКОСТІ ЗОБРАЖЕНЬ В УМОВАХ
НИЗЬКОЇ РОЗДІЛЬНОЇ ЗДАТНОСТІ**

Виконав
ст. гр. КНТ-120

Ілля АЗМАНОВ

Керівник
д.т.н., професор

Сергій СУББОТІН

1



Рисунок Б.1 – Слайд вступу

Об'єкт, предмет та мета дослідження

- Об'єкт дослідження – процес опрацювання зображень для підвищення їхньої роздільної здатності.
- Предмет дослідження – алгоритми та методи поліпшення якості зображень із низькою роздільною здатністю.
- Мета роботи – дослідити, проаналізувати та програмно реалізувати моделі підвищення роздільної здатності зображень.

2



Рисунок В.2 – Слайд з об'єктом, предметом та метою дослідження

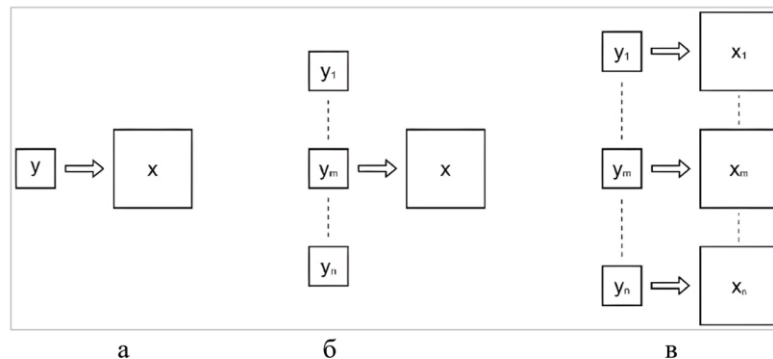
Задачі

- аналіз методів поліпшення якості зображень;
- реалізація обраних моделей;
- тестування та оцінювання моделей;
- оптимізація параметрів моделей;
- підготовка та представлення результатів.

3

Рисунок В.3 – Слайд задач

Схеми побудови алгоритмів суперроздільної здатності



а – однокадрова суперроздільність; б – багатокadroва суперроздільність; в – суперроздільність відеоданих.

4

Рисунок В.4 – Слайд зі схемою побудови алгоритмів суперроздільної здатності

Порівняльний аналіз методів глибокого навчання і традиційних методів для задач суперроздільної здатності зображень

Критерій	Методи глибокого навчання	Традиційні методи
Точність	Дуже висока	Середня
Швидкість навчання	Тривале, іноді дуже довге	Швидке
Швидкість виконання	Помірна	Висока
Складність реалізації	Висока	Низька
Обсяг навчальних даних	Великий	Значно менший
Ресурси обладнання	Високі	Більш доступні
Розмір моделей	Великий	Менший
Інтерпретованість	Складна	Проста
Застосовність до різних типів даних	Висока універсальність	Обмежена гнучкість
Якість вилучення ознак	Автоматична	Потребує ручного налаштування

5

Рисунок В.5 – Слайд з порівнянням методів глибокого навчання і традиційних методів для задач суперроздільної здатності зображень

Обґрунтування вибору мови програмування

Критерій порівняння	Мови програмування				
	Python	R	MATLAB	C++	Julia
Простота і швидкість розробки	Висока	Середня	Висока	Низька	Середня
Підтримка бібліотек глибокого навчання	Відмінна	Хороша	Обмежена	Хороша	Хороша
Продуктивність	Середня	Середня	Середня	Висока	Висока
Спільнота та документація	Велика й активна	Гарне	Гарне	Середнє	Зростаюче
Інтеграція та кросплатформеність	Хороша	Середня	Обмежена	Хороша	Хороша
Підтримка паралельних і розподілених обчислень	Хороша	Обмежена	Обмежена	Хороша	Хороша
Гнучкість та адаптованість	Висока	Середня	Середня	Висока	Висока
Підтримка візуалізації	Відмінна	Відмінна	Хороша	Середня	Хороша

6

Рисунок В.6 – Слайд порівняння мов програмування

Обґрунтування вибору середовища розробки

Критерій порівняння	Середовища розробки				
	Visual Studio Code	PyCharm	Jupyter Notebook	MATLAB	Spyder
Простота та зручність використання	Висока	Висока	Висока	Середня	Висока
Продуктивність	Середня	Середня	Середня	Висока	Середня
Підтримка для наукових обчислень	Середня	Висока	Висока	Висока	Висока
Інструменти для налагодження та аналізу	Хороша	Відмінна	Обмежена	Відмінна	Відмінна
Підтримка плагінів і розширень	Відмінна	Середня	Обмежена	Обмежена	Середня
Інтеграція з іншими інструментами	Хороша	Хороша	Хороша	Середня	Середня
Спільнота та документація	Велика й активна	Велика й активна	Велика й активна	Середня	Середня
Підтримка роботи з версіями (Git)	Відмінна	Відмінна	Середня	Обмежена	Хороша
Візуалізація даних	Середня	Хороша	Відмінна	Відмінна	Хороша

7

Рисунок В.7 – Слайд порівняння середовищ розробки

Обґрунтування вибору фреймворку

Критерій порівняння	Фреймворки				
	PyTorch	TensorFlow	Keras	Caffe	MXNet
Простота та зручність використання	Висока	Середня	Висока	Середня	Середня
Продуктивність	Середня	Висока	Залежить від бекенда	Висока	Висока
Підтримка динамічного та статичного графів	Динамічний	Обидва	Залежить від бекенда	Статичний	Обидва
Спільнота та документація	Велика й активна	Велика й активна	Велика й активна	Середнє	Середнє
Інтеграція з іншими бібліотеками та мовами	Хороша	Хороша	Хороша	Обмежена	Відмінна
Масштабованість і підтримка розгортання	Хороша	Відмінна	Середня	Висока	Відмінна
Зручність відладки	Високе	Середня	Середня	Низька	Середня

8

Рисунок В.8 – Слайд порівняння фреймворків

Рисунок В.9 – Слайд зі схемою алгоритму роботи додатка



Результати роботи моделей



11

Рисунок В.11 – Слайд з результатами роботи моделей

Порівняння моделей за метриками якості

Модель	Масштаб	PSNR	SSIM
SRCNN	2x	31.04	0.8667
SRCNN	3x	28.10	0.7713
SRCNN	4x	26.34	0.7015
VDSR	2x	27.46	0.7126
VDSR	3x	25.78	0.6913
VDSR	4x	24.54	0.6604
EDSR	2x	31.89	0.8957
EDSR	3x	29.03	0.8243
EDSR	4x	27.20	0.8011
SRGAN	–	–	–

12

Рисунок В.12 – Слайд з результатами порівняння моделей за метриками якості

Висновки

- проведено дослідження предметної області, виявлено основні аспекти, що впливають на розробку методів суперроздільної здатності зображень;
- визначено основну суть дослідження, його провідну ідею та цілі;
- обрано та обґрунтовано компоненти розробки;
- спроектовано структуру системи на основі виявлених сутностей та функціональних потреб;
- реалізовано методи суперроздільності з використанням обраних компонентів;
- проведено тестування та аналіз результатів.

13

Рисунок В.13 – Слайд з висновками