НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра системного програмування і спеціалізованих комп'ютерних систем

«На правах рукопису»
УДК 004.93

До захисту допущено:
Завідувач кафедри
_____ Віталій РОМАНКЕВИЧ
«___»_____2021 р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-професійною програмою

«Системне програмування і спеціалізовані комп'ютерні системи»

зі спеціальності 123 «Комп'ютерна інженерія»

на тему: «Вебсервіс підвищення роздільної здатності зображень з використанням SR-алгоритмів»

Виконав:
студент ІІ курсу, групи КВ-01мп
Рекеда Володимир Валерійович _____

Науковий керівник:
Доцент кафедри СПСКС, к.т.н., доцент,
Павловський Володимир Ілліч _____
_____

Рецензент:
_____
_____ _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2021 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра системного програмування і спеціалізованих комп‘ютерних систем

Рівень вищої освіти – другий (магістерський)

Спеціальність – 123 «Комп‘ютерна інженерія»

Освітньо-професійна програма «Системне програмування і спеціалізовані комп‘ютерні системи»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Віталій РОМАНКЕВИЧ

«___»_____2020 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Рекеді Володимиру Валерійовичу

1. Тема дисертації «Вебсервіс підвищення роздільної здатності зображень з використанням SR-алгоритмів», науковий керівник дисертації Павловський Володимир Ілліч, к.т.н., доцент, затверджені наказом по університету від «5» 11. 2021 р. №3682-С

2. Термін подання студентом дисертації 7.12.2021

3. Об’єкт дослідження: SR-алгоритми для покращення роздільної здатності, їх модифікації та застосування у програмних засобах.

4. Вихідні дані: вебсервіс для підвищення роздільної здатності зображень з використанням SR-алгоритмів.

5. Перелік завдань, які потрібно розробити:

1) Проаналізувати принципи роботи SR-алгоритмів та обрати модифікацію SR-алгоритму для використання у вебсервісі.

2) Розробити вебсервіс підвищення роздільної здатності зображення з використанням SR-алгоритмів.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу: презентація.

7. Перелік публікацій:

1) Прикладна математика та комп'ютинг. «XIV науково-практична конференція магістрантів та аспірантів ПМК-2021 факультету прикладної математики» (Київ, 17-19 листопада 2021 р.).

2) VIII Міжнародна науково-технічна Internet-конференція «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами» (Київ, 26 листопада 2021 р.).

9. Дата видачі завдання 7.10.2020

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Вивчення літератури за тематикою МД	02.10.2020	
2.	Розроблення та узгодження технічного завдання	05.10.2020	
3.	Аналіз існуючих рішень	15.12.2020	
4.	Підготовка матеріалів першого розділу МД	21.05.2021	
5.	Розроблення програмного забезпечення	07.09.2021	
6.	Тестування розробленого програмного забезпечення	15.10.2021	
7.	Підготовка матеріалів другого розділу МД	19.10.2021	
8.	Розроблення API програмного забезпечення	26.10.2021	
9.	Оформлення документації дипломного МД	27.11.2021	
10.	Попередній розгляд магістерської дисертації на кафедрі	01.12.2021	

Студент Володимир РЕКЕДА

Науковий керівник Володимир ПАВЛОВСЬКИЙ

РЕФЕРАТ

Актуальність теми. Завдання покращення роздільної здатності зображень є важливим для різних практичних застосувань. Серед таких застосувань можна виділити обробку та аналіз медичних зображень, супутникових фотознімків, обробку даних з пристроїв відеоспостереження і т.д. Ці застосування вимагають високої якості зображень тому за останні два десятиліття було запропоновано багато методів підвищення роздільної здатності зображень.

На сьогоднішній день існує два основних підходи, які застосовуються при вирішенні цієї задачі. Перший спосіб - апаратний. Для цього необхідно використовувати фотоапарати з високоякісною матрицею. Мінусом даного підходу є необхідність використання дорогого спеціалізованого обладнання. Інший спосіб – використання програмних засобів, які реалізують різні алгоритми підвищення якості зображень. Такі алгоритми отримали назву SR алгоритми, які дозволяють якісно збільшити роздільну здатність вихідного зображення, що виходить за межі фізичної роздільної здатності цифрового сенсора, який записав зображення. Перевагою даного підходу є зниження вимог до апаратної складової.

Об’єктом дослідження є проблема підвищення роздільної здатності зображень з початковою низькою роздільною здатністю.

Предметом дослідження є алгоритми та засоби підвищення роздільної здатності зображень.

Мета роботи: модифікація SR-алгоритму та розробка на його основі вебсервісу покращення роздільної здатності зображень.

Наукова новизна полягає в наступному: запропоновано модифікацію Super resolution-алгоритму. Особливістю цієї модифікації є те, що дана модифікація поєднує в собі принципи роботи класичних SR-алгоритмів та SR-алгоритмів, які базуються на бібліотеці навчальних прикладів, завдяки

чому зображення з низькою роздільною здатністю конвертується в зображення з високою роздільною здатністю без будь-якої іншої додаткової вхідної інформації.

Практична цінність отриманих в роботі результатів полягає в тому, що розроблений вебсервіс дозволяє підвищити роздільну здатність зображення без використання додаткових вхідних даних. Такий функціонал може бути корисним у багатьох галузях, які потребують роботи із зображеннями високої роздільної здатності, наприклад, у медичній, у відеоспостереженні і т.д. Також розроблений вебсервіс має відкритий API, що дозволяє використовувати його і в іншому програмному забезпеченні.

Апробація роботи. Основні положення і результати роботи були представлені та обговорювались на XIV науковій конференції магістрантів та аспірантів «Прикладна математика та комп'ютинг» ПМК-2021 (Київ, 17-19 листопада 2021 р.). Також опублікована стаття на конференції «VIII Міжнародна науково-технічна Internet-конференція «Сучасні методи, інформаційне, програмне та технічне забезпечення систем керування організаційно-технічними та технологічними комплексами» (Київ, 26 листопада 2021 р.).

Структура та обсяг роботи. Магістерська дисертація складається з вступу, чотирьох розділів та висновків.

У вступі подано загальну характеристику роботи, зроблено оцінку сучасного стану проблеми, обґрунтовано актуальність напрямку досліджень, сформульовано мету і задачі досліджень, показано наукову новизну отриманих результатів і практичну цінність роботи, наведено відомості про апробацію результатів і їхнє впровадження.

У першому розділі розглянуто існуючі програмні засоби, які дозволяють підвищити роздільну здатність зображення.

У другому розділі описано принципи роботи SR-алгоритмів та наведено порівняння різноманітних модифікації SR алгоритмів.

У третьому розділі та четвертому розділах наведено структуру та опис роботи розробленого програмного забезпечення, а також проведено тестування та проаналізовано результати виконаного дослідження.

У висновках представлені результати проведеної роботи.

Робота представлена на 81 аркуші, містить посилання на список використаних літературних джерел.

Ключові слова: SR-алгоритм, вебсервіс, роздільна здатність, API.

ABSTRACT

Actuality of theme. The task of improving image resolution is important for various practical applications. Such applications include the processing and analysis of medical images, satellite photographs, data processing from video surveillance devices, etc. These applications require high image quality, so over the last two decades, many methods have been proposed to improve image resolution.

To date, there are two main approaches used in solving this problem. The first way is hardware. To do this, you must use cameras with a high quality matrix. The disadvantage of this approach is the need to use expensive specialized equipment. Another way is to use software that implements various algorithms to improve image quality. Such algorithms are called SR algorithms that allow you to qualitatively increase the resolution of the original image, which goes beyond the physical resolution of the digital sensor that recorded the image. The advantage of this approach is to reduce the requirements for the hardware component.

The object of research is the problem of increasing the resolution of the image with the initial low resolution.

The subject of research are algorithms and means of increasing the resolution of images.

The goal of the work: modification of the SR-algorithm and development on its basis of a web service to improve image resolution.

The scientific novelty is as follows: a web service is proposed to increase the resolution of the image using SR-algorithms. This service has an open API. Modification "Super resolution from a single image" is used. A feature of this modification is that the low-resolution image is converted to a high-resolution image without any other additional input information.

Practical value of the results obtained in this work is that the developed web service allows you to increase the resolution of the image without the use of additional input data. This functionality can be useful in many areas that require working with high-resolution images, such as medical, video surveillance, etc. Also developed web service has an open API that allows you to use it in other software.

Approbation of work. The main provisions and results of the work were presented and discussed at the XIV scientific conference of undergraduates and graduate students "Applied Mathematics and Computing" PMK-2021 (Kyiv, November 17-19, 2021). An article was published at the conference "VIII International Scientific and Technical Internet Conference" Modern Methods, Information, Software and Technical Support of Management Systems of Organizational, Technical and Technological Complexes" (Kyiv, November 26, 2021).

Structure and scope of work. The master's dissertation consists of an introduction, four chapters and conclusions.

The introduction presents a general description of the work, assesses the current state of the problem, substantiates the relevance of research, formulates the purpose and objectives of research, shows the scientific novelty of the results and the practical value of the work, provides information on testing the results and their implementation.

The first section discusses the existing software tools that increase the image resolution.

The second section describes the principles of SR-algorithms and compares various modifications of SR-algorithms.

The third section and the fourth sections provide the structure and description of the developed software, as well as testing and analysis of the results of the study.

The conclusions present the results of the work.

The work is presented on 81 sheets, contains references to the list of used literature sources.

Keywords: SR-algorithm, web service, resolution, API.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

ООП - об'єктоорієнтоване програмування. Це парадигма програмування, заснована на представленні програми у вигляді сукупності об’єктів, кожен із яких є примірником певного класу, а класи утворюють ієрархію спадкування.

ОС – операційна система.

ПЗ – програмне забезпечення.

ЦП – центральний процесор.

API (англ. Application Programming Interface) - прикладний програмний інтерфейс. Це набір визначених способів та методів взаємодії різних програмних компонентів.

BN (англ. Batch normalization) – пакетна нормалізація. Метод, що застосовується у нейронних мережах для підвищення їх продуктивності.

BTSRN (англ. Balanced Two-Stage Residual Network) – збалансована двоступенева залишкова нейронна мережа.

CARN (англ. Cascading Residual Network) – каскадна залишкова нейронна мережа.

CLI (англ. Common Language Infrastructure) - відкрита специфікація та технічний стандарт, розроблена Microsoft і стандартизований ISO (ISO/IEC 23271) і Ecma International (ECMA 335), яка описує виконуваний код і середовище виконання, яке дозволяє використовувати кілька мов високого рівня на різних комп’ютерних платформах без переписування для певних архітектур. Це означає, що він не залежить від платформи. .NET Framework, .NET і Mono є реалізаціями CLI.

CNN (англ. Convolutional Neural Network) – згорткова нейронна мережа, різновид штучних нейронних мереж, застосовується ля аналізу візуальних даних.

DRCN (англ. Deep Recursive Convolutional Network) - глибинна рекурсивна згорткова нйеронна мережа.

EDSR (англ. Enhanced Deep Super-Resolution Network) – вид залишкової нейронної мережі, яка застосовується для підвищення роздільної здатності зображення.

ESPCN (англ. Efficient Sub-Pixel Convolutional Neural Network) - вид згорткової нейронної мережі, яка застосовується для підвищення роздільної здатності зображення.

FSRCNN (англ. Fast Super-Resolution Convolutional Neural Network) – вид згорткової нейронної мережі, яка застосовується для підвищення роздільної здатності зображення.

HR-зображення (High Resolution зображення) зображення високої роздільної здатності.

HTTP (англ. Hypertext Transfer Protocol) – протокол передачі даних прикладного рівня, який застосовується в комп’ютерних мережах.

IDE (англ. Integrated Development Environment) - інтегроване середовище розробки. Це програма, яка надає програмістам комплексні засоби для розробки програмного забезпечення.

IL (англ. Intermediate language) - об’єктно-орієнтована мова програмування, розроблена для використання компіляторами для .NET Framework перед статичною або динамічною компіляцією в машинний код.

JIT-компіляція (англ. Just-In-Time (JIT) compilation) - спосіб виконання комп’ютерного коду, який включає компіляцію під час виконання програми, а не перед виконанням.

JSON (англ. JavaScript Object Notation) - відкритий стандартний формат файлу та формат обміну даними.

LapSRN (англ. Laplacian Pyramid Super-Resolution Network) – нейронна мережа з використання піраміди Лапласа.

LR-зображення (Low Resolution зображення) зображення низької роздільної здатності.

MDSR (англ. Multi-Scale Super Resolution Network) – багатомасштабна глибинна нейронна мережа для збільшення роздільної здатності зображення.

MSE (англ. Mean Squared Error) – функція вимірювання середньоквадратичної похибки.

PSNR (англ. Peak Signal-to-Noise Ratio) – критерій для оцінки якості зображення, який відображає співвідношення між максимально можливим значенням (потужністю) сигналу та потужністю спотворення шуму, вимірюється в dB.

RCAB (англ. Residual Channel Attention Block) - блок уваги залишкового каналу.

REST (англ. Representational State Transfer) - це архітектурний стиль програмного забезпечення, який визначає набір методів для побудови інтерфейсу програмування веб-додатків (API).

RG (англ. Residual Group) – залишкова група.

RGB (англ. Red, Green, Blue) - адитивна колірна модель, в якій використовуються червоне, зелене та синє світло для синтезу певного кольору.

RPC (англ. Remote Procedure Call) – протокол, який надає можливість виконання процедури в іншому адресному просторі.

SNR (англ. Signal-to-noise ratio) міра що вказує на співвідношення сигналу до шуму.

SOAP (англ. Simple Object Access Protocol) – специфікація протоколу обміну структурованими даними у розподілених системах. Для передачі інформації використовується XML формат, як протокол передачі найчастіше використовується HTTP.

SRCNN (англ. Super-Resolution Convolutional Neural Network)

SR-алгоритм (Super Resolution алгоритм) – сімейство алгоритмів, призначених для збільшення роздільної здатності зображення.

SSIM (англ. Structural Similarity Index Measure) - метод виміру схожості між двома зображеннями. SSIM-індекс — це метод повного співставлення, іншими словами, він проводить вимірювання якості на основі вихідного не стиснутого зображення.

VDSR (англ. Very Deep Super Resolution) – підхід глибинного навчання для збільшення роздільної здатності зображення.

XML (англ. Extensible Markup Language) – стандарт мов розмітки, який базується на стандартній узагальненій мові розмітки (SGML).

YCbCr - колірна модель, яка використовується у цифрових зображеннях та відео, де Y (luminance) – компонента, яка відповідає за яскравість, Cb (chrominance-blue) – компонента синього кольору, Cr (chrominance-red) – компонента червоного кольору.

ВСТУП

На сьогоднішній день накопичилась і продовжує накопичуватись стрімкими темпами дуже велика кількість мультимедійної інформації. Ця інформація включає в себе зображення, відеоматеріали, аудіофайли та інші типи контенту. Саме зображення є одним з найбільш поширених типів мультимедіа, тому нині стоїть багато завдань по роботі саме з цим типом даних. Головною ціллю таких завдань є отримання зображення з якнайкращою якістю. Серед таких задач можна виділити редагування (видалення або додавання певних деталей, зміна фрагментів композиції), усунення дефектів зображення (зміна яскравості, контрасту, виправлення оптичних дефектів, покращення роздільної здатності). Задля вирішення частини цих зображень потрібна присутність людини, наприклад, при редагуванні фото саме людина вирішує, що потрібно додати до нього або ж прибрати, для вирішення таких завдань потрібно мати навички роботи зі спеціалізованим програмним забезпеченням. Проте вирішення деяких завдань можна автоматизувати та виконувати їх без участі людини, серед таких можна виділити завдання підвищення роздільної здатності зображення.

Завдання покращення роздільної здатності зображень є важливим для різних практичних застосувань. Серед таких застосувань можна виділити обробку та аналіз медичних зображень, супутникових фотознімків, обробку даних з пристроїв відеоспостереження і т.д. Ці застосування вимагають високої якості зображень. Крім того, була накопичена досить велика кількість зображень, які було зроблено на цифрові фотокамери з матрицями перших поколінь, які дозволяли отримати лише зображення низької роздільної здатності. Саме через поширеність такої проблеми за останні два десятиліття було запропоновано багато методів для підвищення роздільної здатності зображень.

На сьогоднішній день існує два основних підходи, які застосовуються при вирішенні цієї задачі. Перший спосіб - апаратний. Для цього необхідно використовувати фотоапарати з високоякісною матрицею. Мінусом даного підходу є необхідність використання дорогого спеціалізованого обладнання. Інший спосіб – використання програмних засобів, які реалізують різні алгоритми

підвищення якості зображень. Такі алгоритми отримали назву SR алгоритми, які дозволяють якісно збільшити роздільну здатність вихідного зображення, що виходить за межі фізичної роздільної здатності цифрового сенсора, який записав зображення. Перевагою даного підходу є зниження вимог до апаратної складової.

1. АНАЛІЗ ІСНУЮЧИХ СИСТЕМ ПОКРАЩЕННЯ РОЗДІЛЬНОЇ ЗДАТНОСТІ ЗОБРАЖЕННЬ

1.1. Загальний опис проблеми

Завдання покращення роздільної здатності зображень на сьогоднішній день є актуальним, адже існує багато галузей, яким для роботи потрібні зображення з високою роздільною здатністю. Для прикладу можна виділити обробку та аналіз медичних зображень, супутникових фотознімків, обробку даних з пристроїв відеоспостереження і т.д. Крім того, була накопичена досить велика кількість архівних зображень, які було зроблено на цифрові фотокамери з низькоякісними матрицями перших поколінь. Саме через поширеність такої проблеми за останні два десятиліття було запропоновано багато методів для підвищення роздільної здатності зображень.

Такі методи поділяють на дві групи: апаратні та програмні. Мінусом апаратного підходу є необхідність використання дорогого спеціалізованого обладнання. В програмному способі використовують спеціальні алгоритми.

Алгоритми для підвищення роздільної здатності отримали назву Super Resolution. Нині існує багато модифікацій даного сімейства алгоритмів, які можна поділити на дві групи. Алгоритмам першої групи для роботи необхідно подавати на вхід додаткові зображення предмета, зображеного на вхідному зображенні низької здатності. Алгоритми другої групи для підвищення роздільної здатності використовують власну бібліотеку пар зображень низької та високої роздільної здатності, серед фрагментів цих зображень виконується пошук збігів зі вхідним зображенням низької роздільної здатності.

Для того, щоб була можливість користуватись таким програмним забезпеченням з будь-якого пристрою доцільно виконати його у формі веб-сервісу. Крім того, у створеному веб-сервісі має бути присутній API для того, щоб була можливість користуватись розробленим програмним забезпеченням і у інших програмах, наприклад, веб-додатках, класичних додатках чи у чат-ботах.

1.2. Опис існуючих рішень

Основне призначення веб-сервісу це збільшення роздільної здатності зображення для отримання зображення високої роздільної здатності із зображення низької роздільної без будь-якої додаткової інформації.

На сьогоднішній день існує кілька видів програмного забезпечення, які дозволяють підвищити роздільну якість зображення. Такі програмні засоби існують у різних формах, наприклад у формах класичних програм або у формі веб-додатків.

Одним з найпопулярніших прикладів програмного забезпечення, які містять функціонал для підвищення роздільної здатності, можна назвати Adobe Photoshop (рисунок 1.1). Adobe Photoshop — це растровий графічний редактор, розроблений і виданий Adobe Inc. у вигляді класичного додатку, доступний для операційних систем Windows і macOS [1].

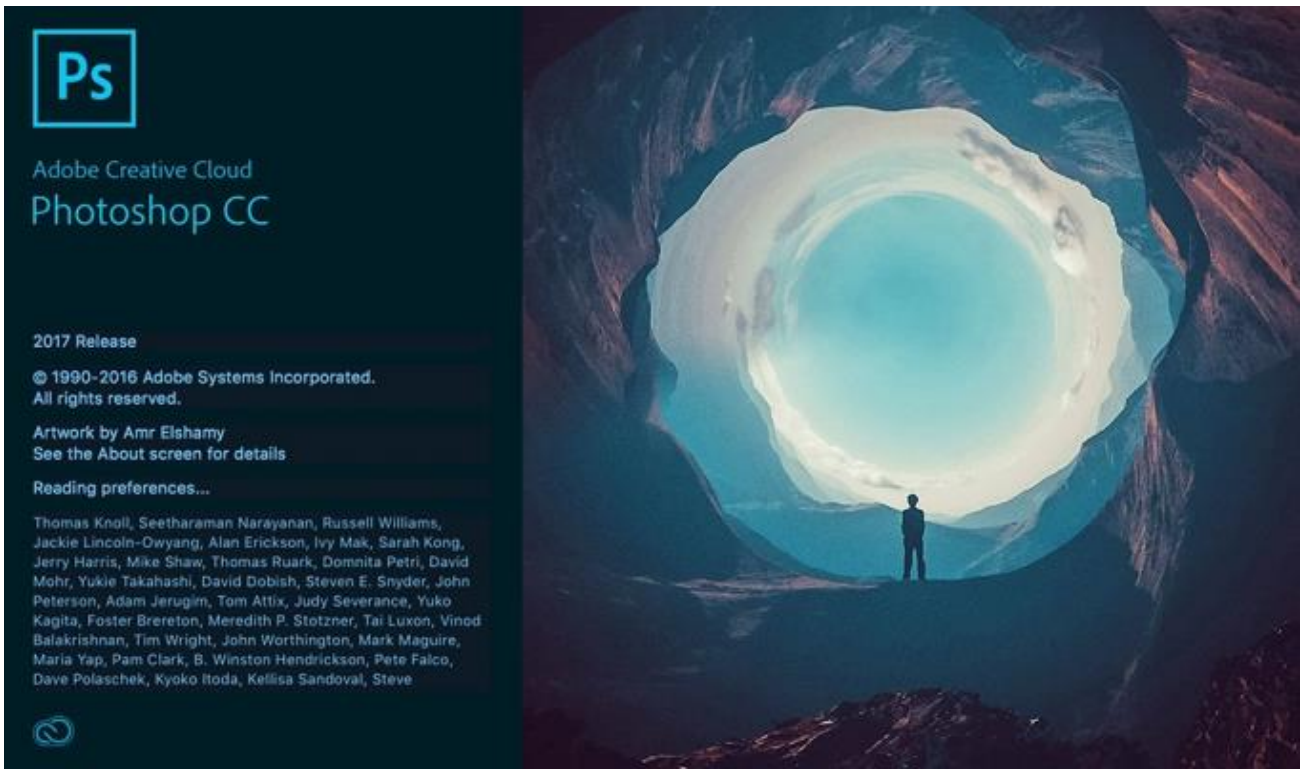

Рисунок 1.1 – Графічний редактор «Adobe Photoshop»

Спочатку він був створений у 1988 році Томасом і Джоном Ноллами. З тих пір програмне забезпечення стало галузевим стандартом не тільки для редагування растрової графіки, а й у цифровому мистецтві в цілому. Photoshop має обмежені можливості редагування або візуалізації тексту та векторної графіки (особливо за допомогою відсічного контуру для останньої), а також 3D-графіки та відео. Його функціонал також включає можливість покращити роздільну здатність зображення, в цьому інструменті використано саме один за SR-алгоритмів. Крім того, набір функцій можна розширити за допомогою плагінів. Плагіни - програми, розроблені та розповсюджені незалежно від Photoshop, які працюють у ньому та пропонують нові або розширені функції. До недоліків даного програмного забезпечення можна віднести складність при навчанні, яка зумовлена великою кількістю різноманітних функцій, тип ліцензії Adobe Photoshop – це умовна безкоштовна програма з безкоштовним тимчасовим періодом для ознайомлення, а також відсутність відкритого API.

Отже, Adobe Photoshop це потужний інструмент для роботи з графікою, який має широкий набір інструменті, але для вирішення поставленої задачі не підходить, адже він не має відкритого API, що не дозволяє використовувати його в інших програмних засобах.

Серед інших класичних додатків також можна виділити SmartDeblur. Основним призначенням цього ПЗ є відновлення розмитих зображень, присутня можливість налаштування рівня згладжування, також присутні інструменти для підвищення роздільної здатності зображення. Програма доступна лише для операційних систем сімейства Windows. До недоліків також віднести те, що дане ПЗ є умовно безкоштовним, а саме в базовій версії на відредаговані зображення додається водяний знак, також відсутній API, тому для вирішення поставленої задачі таке ПЗ не підходить.

Також існує досить багато різноманітних вебдодатків, які дозволяють певним чином покращувати якість зображень. Зазвичай подібні додатки мають менш різноманітний функціонал, але при їх використання є більш зручним, адже вони не потребують встановення.

Серед веб-додатків, які дозволяють покращити роздільну якість зображення можна виділити letsenhance.io.

Даний веб-додаток цікавий тим, що дозволяє обробляти фото в повністю автоматизованому режимі, функціонал включає в себе можливість покращити роздільну здатність, змінити яскравість, контраст, а також видалити фото з фото (рисунок 1.2). Крім того, даний веб-додаток має власний API, що дозволяє використовувати його і в інших програмних засобах [2]. Головним недоліком є те, що використання API повністю платне.

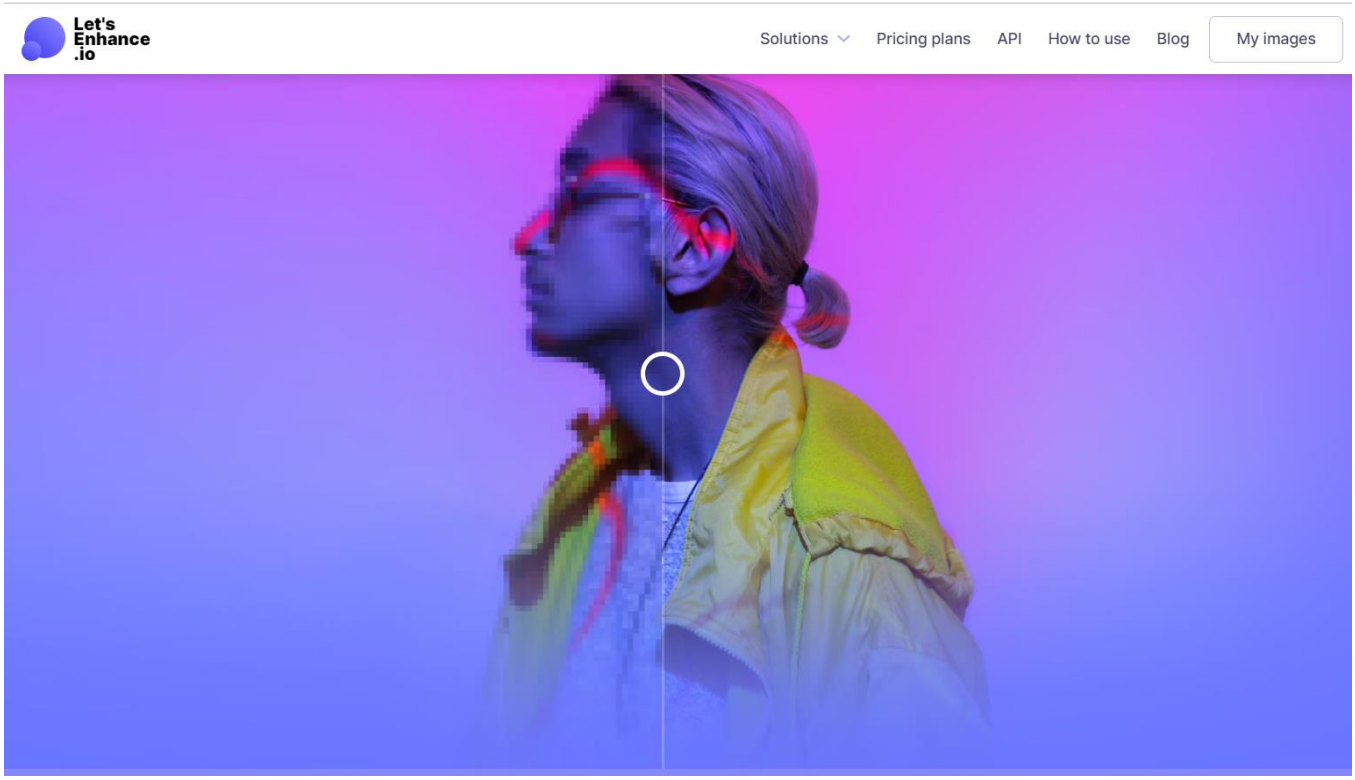

Рисунок 1.2 – Веб-додаток «letsenhance.io»

Даний веб-додаток цікавий тим, що дозволяє обробляти фото в повністю автоматизованому режимі, функціонал включає в себе можливість покращити роздільну здатність, змінити яскравість, контраст, а також видалити фото з фото. Крім того, даний веб-додаток має власний API, що дозволяє використовувати його і в інших програмних засобах. Головним недоліком є те, що використання API повністю платне.

В таблиці 1.1 наведено порівняння кількох ПЗ, які дозволяють підвищити роздільну здатність зображення.

Таблиця 1.1 - Порівняльна таблиця програмних засобів покращення роздільної здатності зображень

Назва	Тип	Платформа	Тип ліцензії	Наявність API
Adobe Photoshop	Класичний додаток	Windows, macOS	Умовно безкоштовне ПЗ	Ні
letsenhance.io	Веб-додаток	Будь-яка	Умовно безкоштовне ПЗ	Так
SmartDeblur	Класичний додаток	Windows	Умовно безкоштовне ПЗ	Ні

Основним недоліком всіх описаних вище сервісів є їх умовна безкоштовність. Такий тип ліцензії накладає певні обмеження на використання ПЗ, серед яких пробний період, обмежена кількість зображень, які можливо опрацювати, а також нанесення водяних знаків на оброблені зображення. Крім того, не всі ці сервіси мають власні API, що унеможливлює їх використання в інших програмних засобах. З цього можна зробити висновок, що у вільному доступі практично відсутні ПЗ підвищення роздільної здатності зображення, які б не мали ліцензійних обмежень та надавали власні API для їх використання в якості веб-сервісу.

Таким чином задача створення веб-сервісу підвищення роздільної здатності зображень з відкритим API є актуальною.

1.3. Постановка задачі магістерської дисертації

На сьогоднішній день завдання покращення роздільної здатності зображень на сьогоднішній день є актуальним, тому що існує багато галузей, де потрібні саме зображення з високою роздільною здатністю. Для прикладу можна виділити обробку та аналіз медичних зображень, супутникових фотознімків, обробку даних з пристроїв відеоспостереження і т.д. Існує два основних підходи вирішення цієї задачі: апаратний та програмний. Так як апаратний підхід потребує більших

витрат, доцільно використовувати саме програмний підхід. Алгоритми, які дозволяють підвищити роздільну якість зображення отримали назву Super Resolution.

Головною задачею магістерської дисертації є розробка вебсервісу підвищення роздільної здатності з відкритим API. Даний вебсервіс має дозволяти підвищити роздільну здатність з лише одного вхідного зображення. Також необхідно створити модифікацію SR-алгоритму, яка буде задовольняти цю вимогу.

ВИСНОВКИ ДО ПЕРШОГО РОЗДІЛУ МД

У даному розділі дисертації проаналізовано актуальність проблеми, а саме необхідність створення вебсервісу з відкритим API для підвищення роздільної здатності зображення. Аналіз показав, що проблема є нині актуальною, адже завдання покращення роздільної здатності зображень є важливим для різних практичних застосувань. Серед таких застосувань можна виділити обробку та аналіз медичних зображень, супутникових фотознімків, обробку даних з пристроїв відеоспостереження і т.д. Проведено порівняння існуючих програмних засобів, які можуть виконувати поставлену задачу, але всі вони не повністю відповідають поставленим вимогам.

2. ОПИС SR-АЛГОРИТМІВ ТА АНАЛІЗ ТЕХНОЛОГІЙ РОЗРОБКИ ВЕБСЕРВІСІВ

2.1. Загальні принципи роботи SR-алгоритмів

Super-resolution (SR) алгоритми — це клас методів, які дозволяють покращити (збільшити) роздільну здатність зображення. Такі алгоритми знайшли широке застосування в таких сферах:

1. спостереження: для виявлення, ідентифікації та виконання розпізнавання обличчя на зображеннях з низькою роздільною здатністю, отриманих з камер безпеки;

2. медицина: отримання зображень МРТ з високою роздільною здатністю може бути складним, коли справа доходить до часу сканування, просторового покриття та співвідношення сигнал/шум (SNR). Застосування SR-алгоритмів допомагає вирішити цю проблему, генеруючи МРТ високої роздільної здатності з зображень МРТ з низькою роздільною здатністю;

3. медіа: SR алгоритми можна використовувати для зниження витрат на сервер, оскільки медіафайли можна надсилати з нижчою роздільною здатністю та збільшувати на льоту.

SR-алгоритми ґрунтуються на ідеї, що комбінація послідовності зображень однієї сцени з низькою роздільною здатністю (зашумлення) може бути використана для створення зображення з високою роздільною здатністю або кількох таких зображень. Таким чином, завдання SR алгоритмів полягає у відновленні вихідного зображення сцени з високою роздільною здатністю, використовуючи набір зображень тієї ж сцени, але з нижчою роздільною здатністю.

Super Resolution алгоритми можна розділити на два основні класи: класичні SR-алгоритми та SR-алгоритми, які базуються на послідовності навчальних прикладів. В класичних алгоритмах використовується кілька вхідних зображень з низькою роздільною здатністю. Такі SR-алгоритми базуються на припущенні, що

кілька зображень об’єкту, які використовуються для побудови зображення з високою роздільною здатністю, є геометричними перетвореннями та несумісними версіями один одного, і якщо їх об’єднати належним чином, то можливо отримати зображення з більшою детальністю, ніж міститься в кожному окремому вхідному зображення (рисунок 2.1). Такий підхід не дуже практичний, оскільки кілька зображень не завжди доступні.

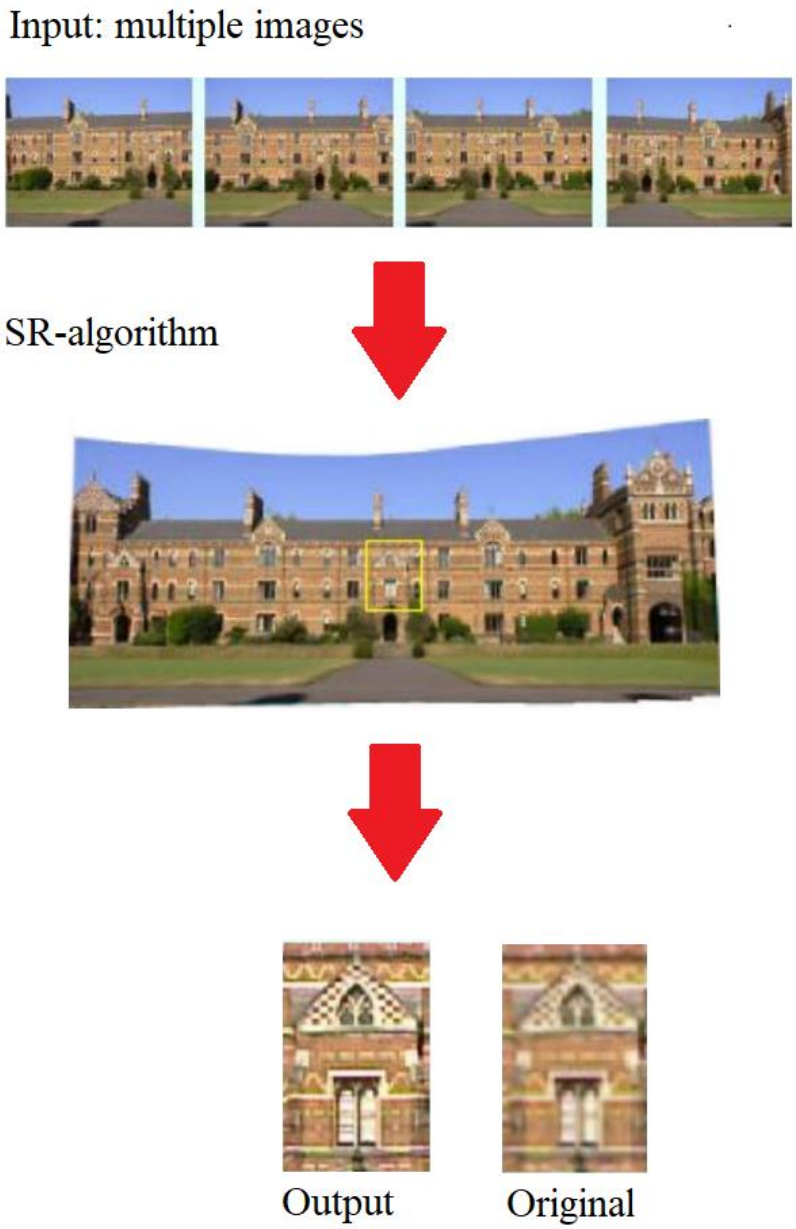

Рисунок 2.1 – Принцип роботи класичного SR-алгоритму

Однією з проблем використання класичних SR-алгоритмів є те, що кілька зображень з низькою роздільною здатністю можуть представляти різні точки огляду однієї і тієї ж сцени, а реєстрація зображення стосується відображення відповідних точок на цих зображеннях з реальними точками вихідної сцени та перетворення даних в одну систему координат. Для реєстрації зображень може знадобитися кілька типів перетворень, таких як афінні перетворення, біквадратичні перетворення або планарні гомографічні перетворення. Це вирівнювання включає геометричний компонент, а також фотометричний компонент.

На рисунку 2.2 проілюстрований приклад планарного графічного перетворення. Точки x і x’ відповідають фактичній точці X в оригінальній сцені.

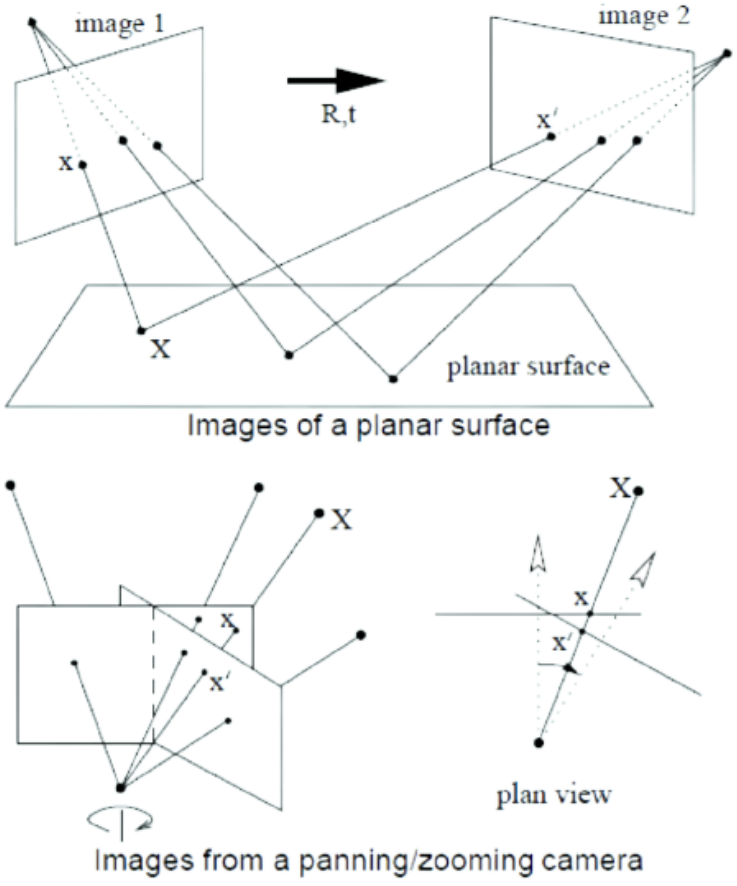

Рисунок 2.2 – Планарне графічне перетворення

При використанні SR-алгоритмів, які базуються на послідовності навчальних прикладів, існує база даних, в якій попарно зберігаються зображення з

високою та низькою роздільною здатністю (рисунок 2.3). Ідея полягає в розбитті вхідного зображення на окремі фрагменти та пошук схожих фрагментів серед наявних в базі даних [3]. Серед недоліків цього підходу – велика ймовірність отримання неточного зображення.

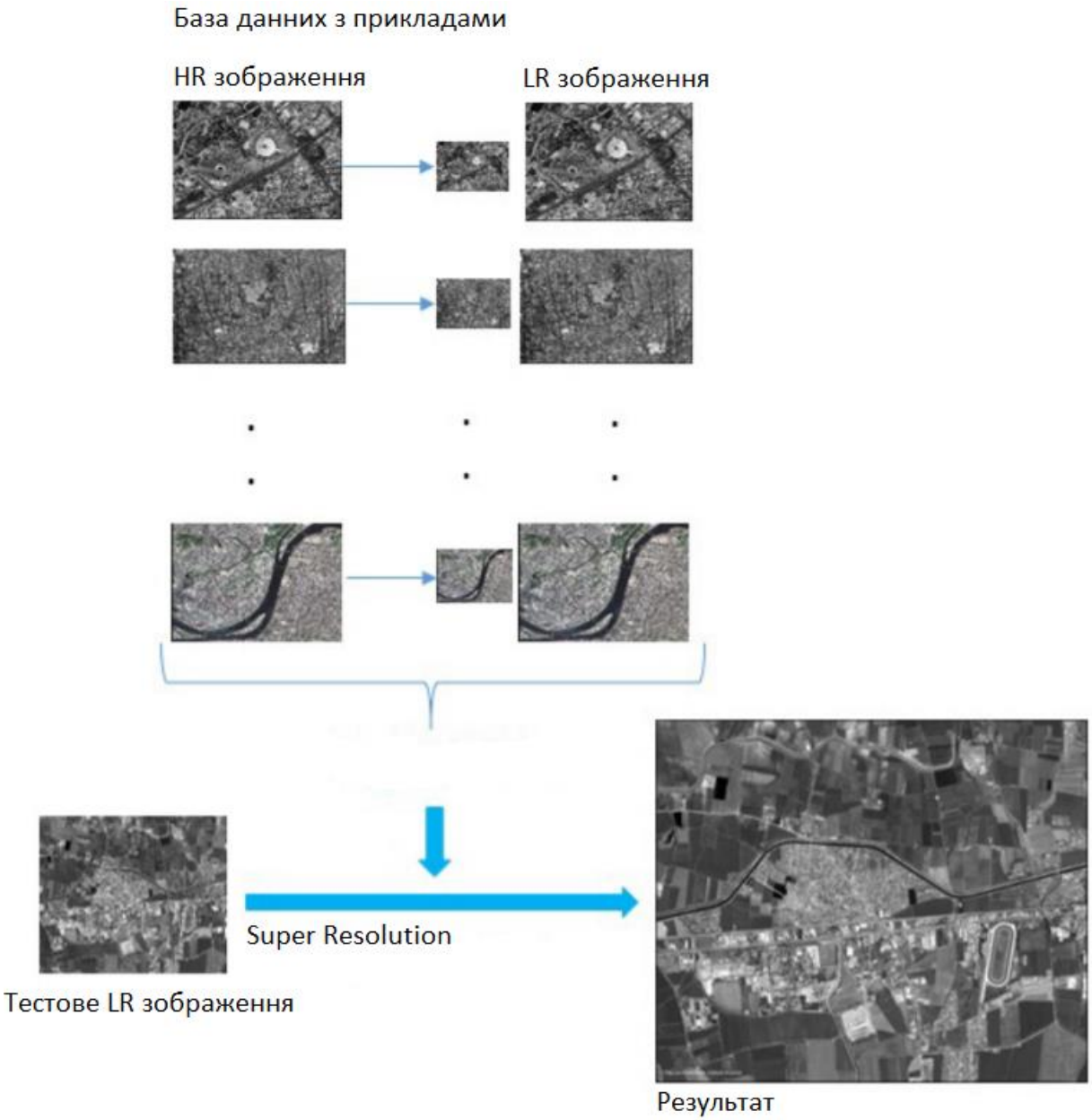

Рисунок 2.3 – Принцип роботи SR-алгоритмів, які базуються на послідовності навчальних прикладів

2.2. Порівняння модифікацій SR-алгоритмів

До класичних SR-алгоритмів відносять наступні види алгоритмів: алгоритми з використанням регуляризації, алгоритми, що базуються на спектральному поданні, інтерполяційні алгоритми, баєсівські алгоритми [4].

Підходи, засновані на регуляризації, позбавляються неоднозначності у рішеннях, використовуючи апріорні знання про природу зображень. Найбільш відомим підходом для регуляризації є Compressive Sensing. Compressive Sensing – це парадигма, що використовується в теорії сигналів, заснована на тому, що певний сигнал (в даному випадку зображення) – це невипадковий шум, а щось, що має структуру і тому для зображення, представлення його у вигляді набору пікселів є надлишковим. Тобто зображення з реального світу, яке може бачити людина, це лише одне з можливих представлень. Тому в деякому базисі воно повинно мати багато ненульових коефіцієнтів. Зазвичай застосовують спектральні уявлення у вигляді коефіцієнтів низки Фур'є, синусів, косинусів або у вигляді коефіцієнтів деякого вейвлетного розкладання.

Крім того, це сімейство алгоритмів також можна поділити на стохастичний, детермінований та гібридний підходи. Стохастичні підходи використовують випадкові величини у формі розподілів ймовірностей, щоб ефективно забезпечити стабільні оцінки та розрізняти можливі рішення за допомогою моделі попереднього зображення. Хоча детерміновані підходи не використовують жодних випадкових величин, але їх можна сформулювати, вибравши змінну для мінімізації Лагранжа та розв’язати обернену задачу, використовуючи попередню інформацію про рішення, яка може бути використана для правильної постановки проблеми. Гібридні підходи використовують комбінацію стохастичного та детермінованого підходів.

Для алгоритмів, що базуються на спектральному поданні, потрібно здійснити перетворення вхідного зображення в спектральне подання. Для цього потрібно розкладання зображення в інший базис. Прикладом такого базису може

бути двовимірне дискретне перетворення Фур'є або дискретне косинусне перетворення. Коли зображення представлене у вигляді спектра, коефіцієнти з різних зображень об'єднуються, потім застосовується зворотне перетворення Фур'є, і в результаті виходить зображення вищої роздільної здатності. Переваги цього підходу в тому, що дрібні деталі в зображенні, які потрібно відновити, якраз відповідають високочастотним компонентам спектру - їх вдається екстраполювати за рахунок інформації з декількох LR-зображень. Однак, цей підхід передбачає, що геометричне спотворення між вихідними зображеннями низької роздільної здатності – лише глобальне зрушення, що в реальних даних зовсім не обов'язково.

Для вирішення задачі за допомогою інтерполяційного алгоритму загальному вигляді потрібно виконати ряд кроків:

1. потрібно визначити крок-відстань між зображеннями;
2. спроектувати вихідне зображення на сітку HR;
3. регуляризувати отримане зображення за допомогою різних методів видалення змащування та шуму.

Наприклад, найвідоміший алгоритм інтерполяції Nearest Neighbor надає невідомому пікселю значення, що дорівнює значенню найближчого відомого.

При використанні байєсівських алгоритмів на основі вхідних LR-зображень моделюється ймовірнісний розподіл можливих HR-зображень. Для вихідного зображення обчислюються функції, що визначають, з якою ймовірністю воно може належати до кожного з класів, за ними обчислюються апостеріорні ймовірності класів. Зображення відноситься до того класу, для якого апостеріорна ймовірність максимальна.

Крім того, існують гібридні методи, а також методи, які базуються на використанні нейронних мереж.

2.2.1. Методи Pre-Upsampling Super Resolution

В методах Pre-Upsampling Super Resolution використовуються традиційні методи, такі як бікубічна інтерполяція та глибинне навчання, щоб збільшити роздільну здатність зображення. Найпопулярнішим методом цього класу є метод SRCNN, також в цьому методі було вперше застосував глибинне навчання.

Super-Resolution Convolutional Neural Network (SRCNN) — в цьому алгоритмі використовується архітектура CNN, яка складається з трьох шарів: шар для екстракції латок, нелінійного відображення і реконструкції (рисунок 2.4). Шар екстракції латок використовується для вилучення щільних ділянок із входу та їх представлення за допомогою згорткових фільтрів. Шар нелінійного відображення складається із згорткових фільтрів 1×1, які використовуються для зміни кількості каналів і додавання нелінійності. Останній шар реконструкції реконструює зображення з високою роздільною здатністю [5]. Функція втрат MSE використовується для навчання мережі, а PSNR використовується для оцінки результатів.

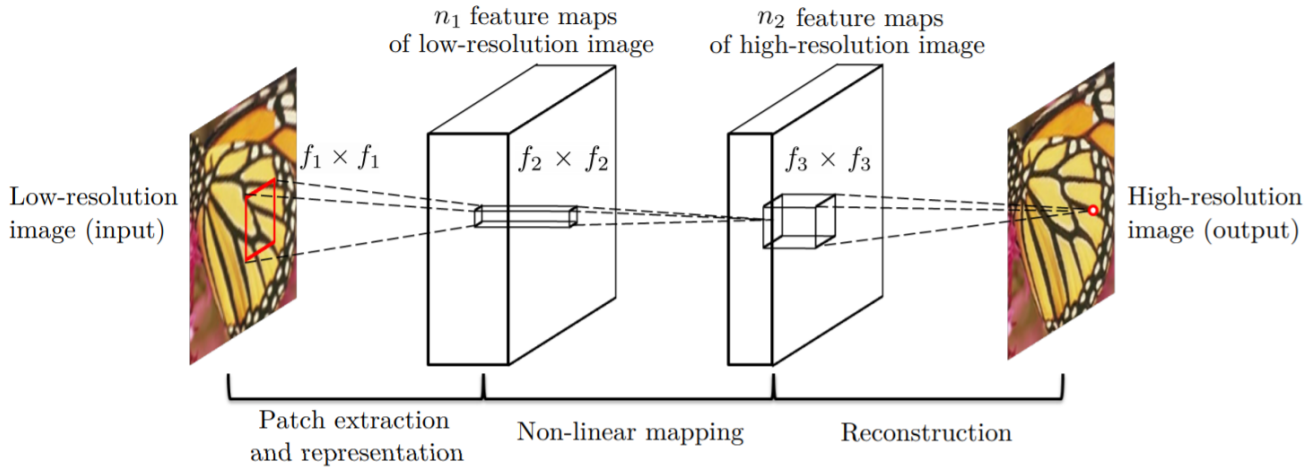

Рисунок 2.4 – CNN архітектура для Super-Resolution Convolutional Neural Network (SRCNN)

Також існує модифікація SRCNN, яка отримала назву Very Deep Super Resolution (VDSR) (рисунок 2.5).

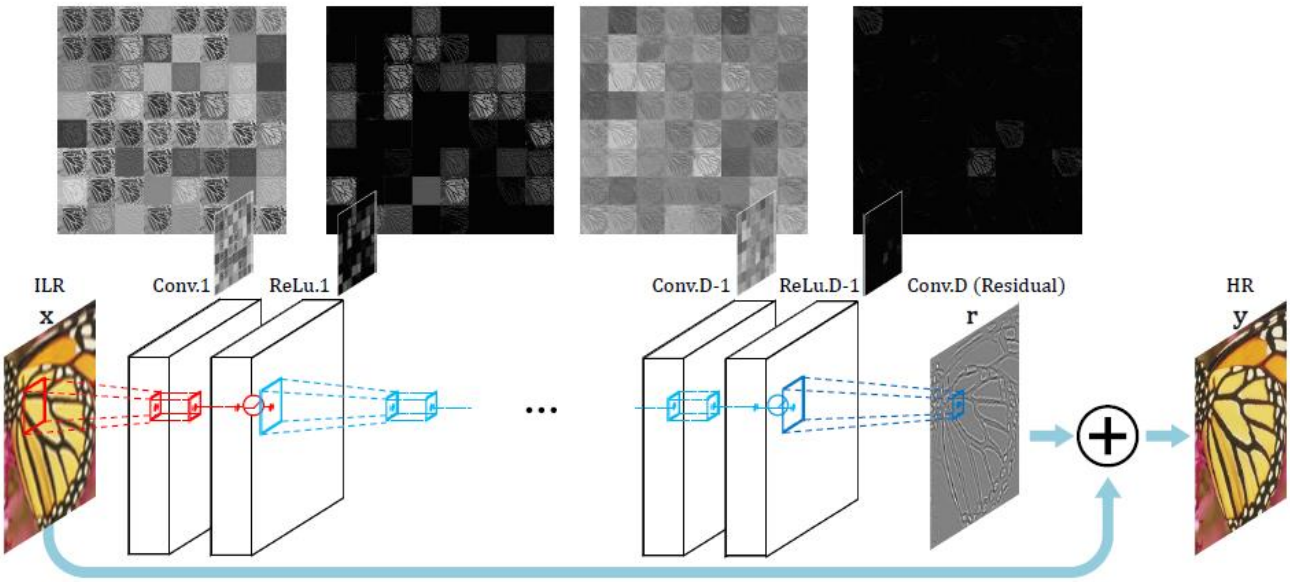

Рисунок 2.5 – Very Deep Super Resolution (VDSR)

Відмінності від SRCNN полягають у додаванні наступних функцій:

1. використовується глибинна мережа з невеликими згортковими фільтрами 3×3 замість меншої мережі з великими згортковими фільтрами. Це засновано на архітектурі VGG [6];

2. мережа намагається дізнатися залишок вихідного зображення та інтерпольованого входу. Це спрощує завдання. Початкове зображення з низькою роздільною здатністю додається до мережевого виходу, щоб отримати остаточний результат HR;

3. відрізання градієнта використовується для навчання глибинної мережі з вищими темпами навчання;

2.2.2. Методи Post-Upsampling Super-Resolution

Оскільки процес вилучення ознак в методах Pre-Upsampling Super Resolution відбувається в просторі високої роздільної здатності, такі методи потребують вищу обчислювальна потужність. Що вирішити цю проблему було

запропоновано методи Post-Upsampling Super-Resolution. В даних методах процес вилучення ознак відбувається в просторі низької роздільної здатності і лише після цього відбувається збільшення роздільної здатності, такі зміни значно зменшують кількість необхідних обчислень. Крім того, замість використання простої бікубічної інтерполяції для підвищення дискретизації, використовується засвоєне підвищення дискретизації у формі деконволюції/підпіксельної згортки, що робить мережу доступною для наскрізного навчання.

Одним з прикладів таких методів є Fast Super-Resolution Convolutional Neural Networks (FSRCNN) (рисунок 2.6).

FSRCNN в кінцевому підсумку досягає кращих результатів, ніж SRCNN, і при цьому швидше.

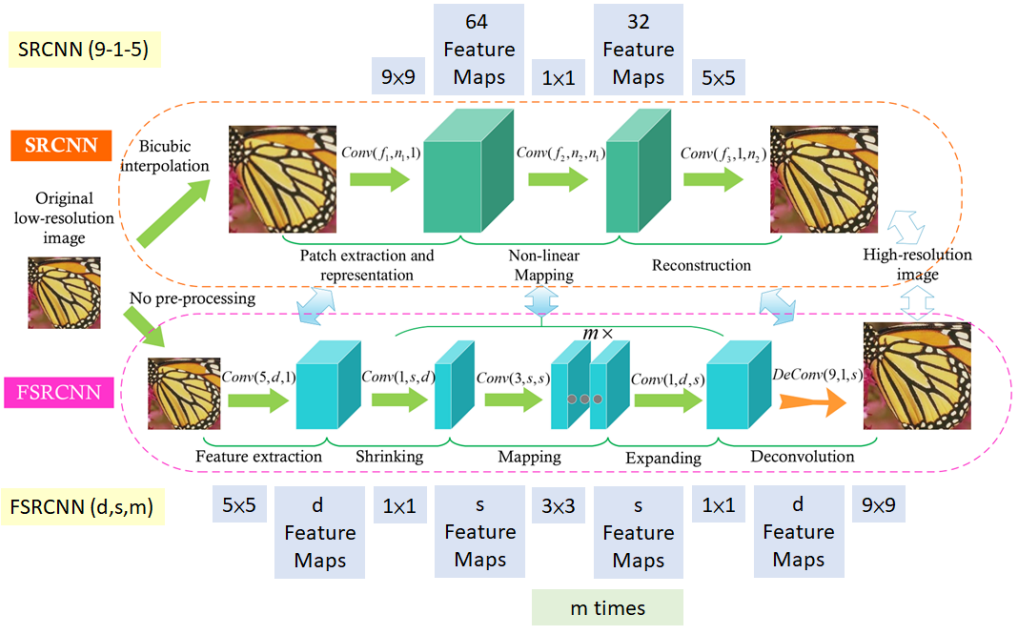

Рисунок 2.6 – Fast Super-Resolution Convolutional Neural Networks (FSRCNN)

Основні відмінності FSRCNN від SRCNN:

1. відсутня попередня обробка. Вилучення ознак відбувалося в просторі з низькою роздільною здатністю;

2. згортка 1×1 використовується після початкової згортки 5×5, щоб зменшити кількість каналів і, отже, зменшити кількість обчислень і використання пам’яті [7];

3. використовується кілька згорток 3×3 замість великого згорткового фільтра, подібно до того, як працює мережа VGG, спрощуючи архітектуру для зменшення кількості параметрів;

4. підвищення роздільної здатності здійснюється за допомогою деконволюційного фільтра, що покращує модель.

FSRCNN в кінцевому підсумку досягає кращих результатів, ніж SRCNN, і при цьому має кращу швидкодію.

Ще одним прикладом підходу Pre-Upsampling Super Resolution є Efficient Sub-Pixel Convolutional Neural Network (ESPCN). ESPCN вводить концепцію субпіксельної згортки для заміни деконволюційного шару [8]. Це вирішує дві пов’язані з ним проблеми:

1. деконволюція виконується в просторі з високою роздільною здатністю, і, таким чином, є більш дорогою з точки зору обчислень;

2. це вирішує проблему шахової дошки під час деконволюції, яка виникає через операцію перекриття згортки.

Субпіксельна згортка працює шляхом перетворення глибини в простір (рисунок 2.7). Пікселі з кількох каналів у зображенні з низькою роздільною здатністю перебудовуються на один канал у зображенні з високою роздільною здатністю. Для прикладу вхідне зображення розміром 5×5×4 може переставити пікселі в останніх чотирьох каналах в один канал, в результаті чого буде зображення розміром 10×10 HR.

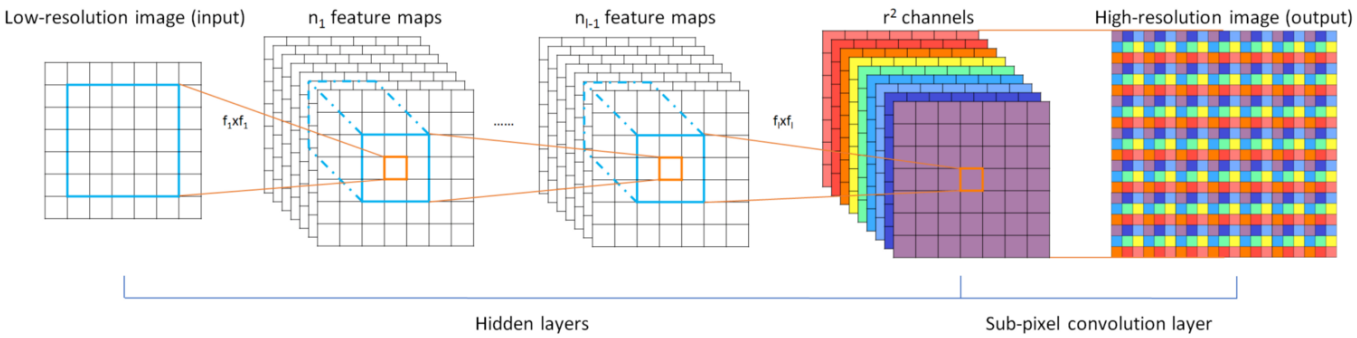

Рисунок 2.7 – ESPCN з двома шарами згортки

2.2.3. Методи з використанням залишкових нейронних мереж

Також існують методи з використанням залишкових нейронних мереж (Residual Networks). Прикладом такого методу є Enhanced Deep Super-Resolution Network (EDSR).

Архітектура EDSR заснована на архітектурі SRResNet, що складається з кількох залишкових блоків. Залишковий блок в EDSR показаний вище. Основна відмінність від SRResNet полягає в тому, що шари пакетної нормалізації (Batch normalization) видаляються (рисунок 2.8). Шари BN виконують операцію нормалізації вхідних даних, тим самим обмежуючи діапазон мережі; видалення BN призводить до підвищення точності. Шари BN також споживають пам’ять, і їх видалення призводить до зменшення витрат пам’яті до 40%, що робить навчання мережі більш ефективним [9].

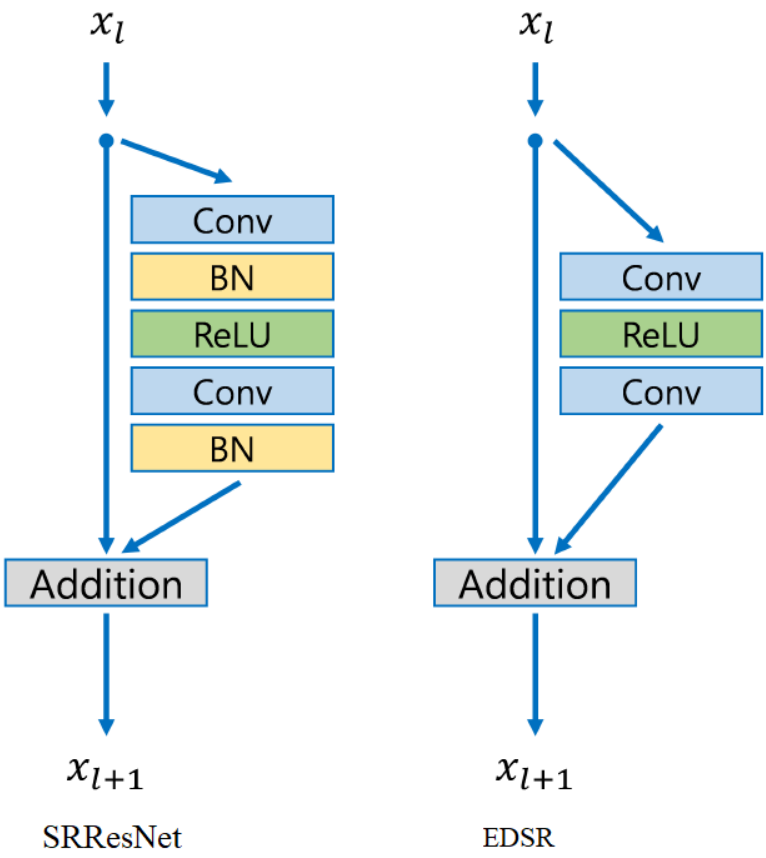

Рисунок 2.8 – Відмінність EDSR від SRResNet

Для EDSR також існує розширенням, яке отримало назву Multi-Scale SR Network (MDSR). Це розширення має кілька модулів введення та виводу, які дозволяють збільшити роздільну здатність у 2, 3 та 4 рази. На початку присутні модулі попередньої обробки для масштабного введення, які складаються з двох залишкових блоків з ядрами 5×5. Велике ядро використовується для шарів попередньої обробки, щоб зберегти мережу неглибокою, але при цьому досягти високого сприйнятливого поля. Наприкінці специфічних для масштабу модулів попередньої обробки знаходяться спільні залишкові блоки, які є загальним блоком для даних усіх роздільних можливостей [9]. Нарешті, після спільних залишкових блоків йдуть специфічні для кожного масштабу модулі підвищення роздільної здатності. Хоча загальна глибина MDSR становить 5 порівняно з одномасштабним EDSR, кількість параметрів лише 2,5, а не 5 завдяки використанню спільних параметрів. MDSR дозволяє досягти результатів які є схожі з результатами використання EDSR.

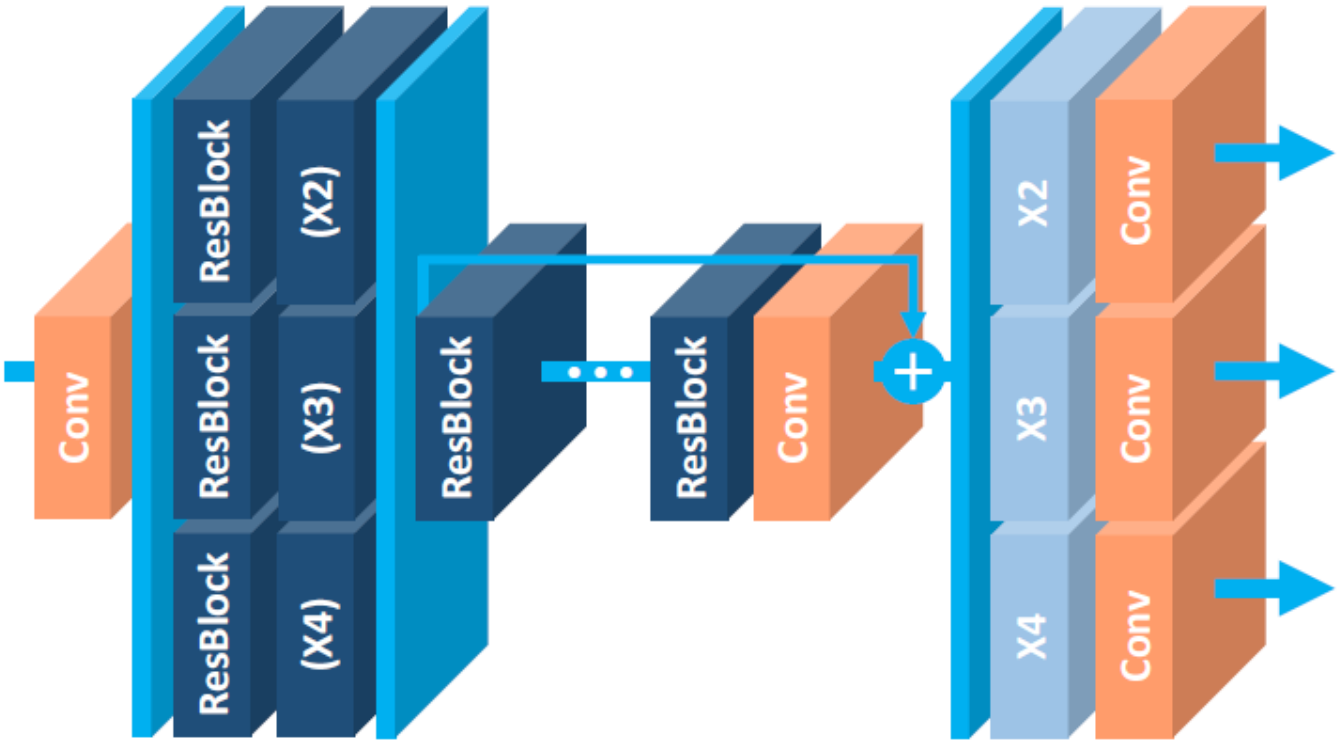

Рисунок 2.8 – Multi-Scale SR Network (MDSR)

Також можливе використання каскадних залишкових нейронних мереж (Cascading Residual Network). Використання таких нейронних мереж має наступні переваги порівняно з використанням звичайних залишкових мереж:

1. каскадний механізм як на локальному, так і на глобальному рівнях надає мережі можливість отримувати більше інформації;
2. крім нейронної мережі CARN, також можна використовувати CARN-M, яка має простішу архітектуру, але при цьому дозволяє отримати результат без значного погіршення. Це досягається за рахунок використання рекурсивної архітектури мережі.

Виходи проміжних шарів каскадуються до вищих шарів і, нарешті, сходяться в одному шарі згортки 1x1 (рисунок 2.9). Зауважимо, що проміжні шари реалізовані як каскадні блоки, які самі розміщують локальні каскадні з'єднання. Місцевий каскад майже ідентичний глобальному, за винятком того, що одиничні блоки є простими залишковими блоками [10].

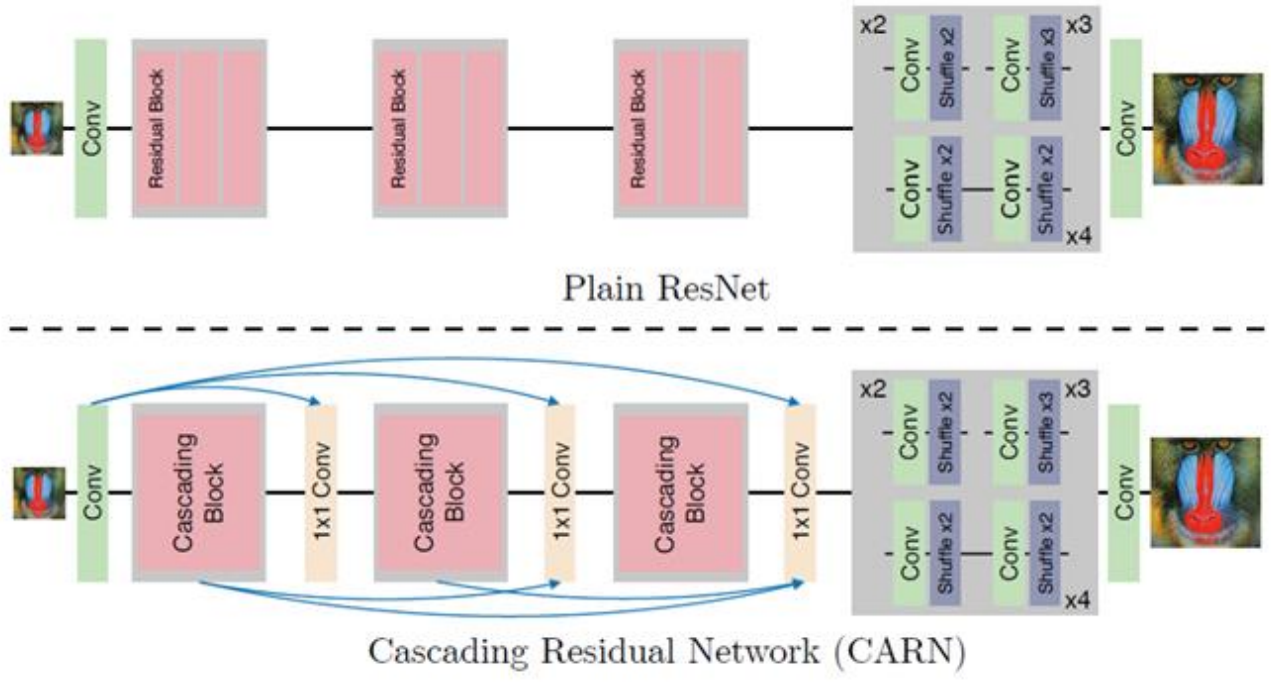

Рисунок 2.9 – Відмінність при використанні звичайної залишкової мережі (Plain ResNet) від каскадної залишкової мережі (CARN)

Кожен залишковий блок у каскадному блоці закінчується згорткою 1x1, яка має з'єднання з усіма попередніми залишковими блоками разом з основним входом, подібно до того, як працює глобальний каскад.

Залишковий блок у ResNet замінено новим дизайном Residual-E блоком, натхненним глибинними згортками в MobileNet. Замість глибинних згорток використовуються групові згортки, і результати показують зменшення кількості використаних обчислень у 1,8-14 разів, залежно від розміру групи.

Для подальшого зменшення кількості параметрів використовується загальний залишковий блок (рекурсивний блок), що призводить до зменшення кількості параметрів у три рази більше, ніж початкове. Як видно з рисунку 2.10, рекурсивний загальний блок допомагає зменшити загальну кількість параметрів.

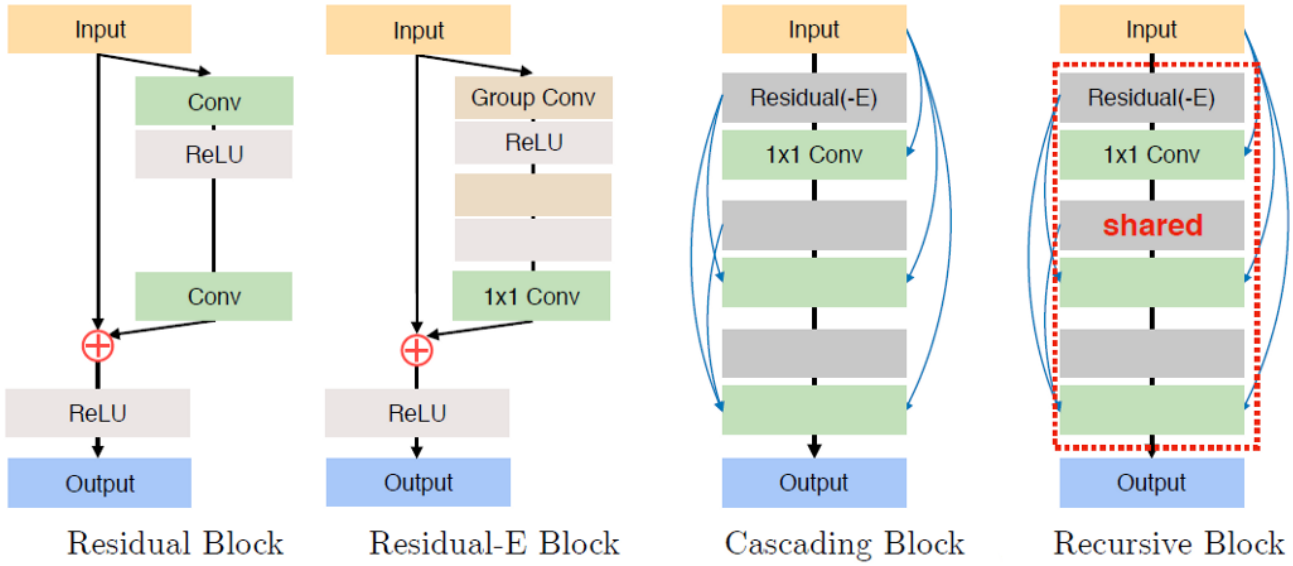

Рисунок 2.10 – Відмінність при використанні звичайної залишкової мережі (Plain ResNet) від каскадної залишкової мережі (CARN)

Також існують підходи, які використовують багатоступеневі залишкові нейронні мережі (Multi-Stage Residual Networks). Такі підходи були розроблені для вирішення задачі виділення ознак окремо в просторі з низькою і високою роздільною здатністю. На першому етапі виокремлюються грубі ознаки, тоді як на наступному рівні відбувається їх покращення.

Прикладом такого підходу є Balanced Two-Stage Residual Network (BTSRN). BTSRN складається з двох етапів: стадії низької роздільної здатності (LR) і стадії високої роздільної здатності (HR) (рисунок 2.11). Стадія LR складається з 6 залишкових блоків, тоді як стадія HR містить 4 залишкових блоку. Згортка на етапі HR вимагає більше обчислень, ніж на етапі LR, оскільки розмір вхідних даних більший. Кількість блоків на обох етапах визначається таким чином, щоб досягти компромісу між точністю та продуктивністю [11].

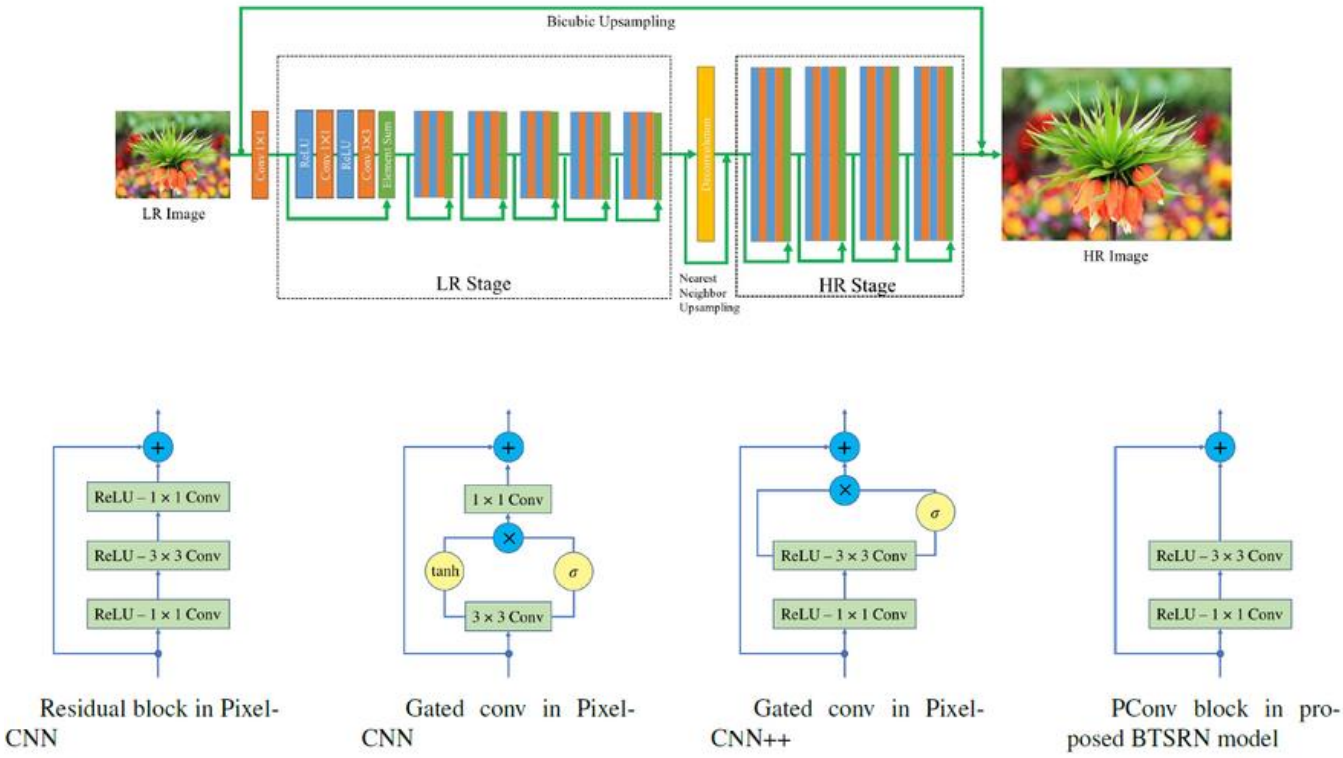

Рисунок 2.11 – Balanced Two-Stage Residual Network (BTSRN)

На виході етапу LR підвищується роздільна здатність перед відправкою на етап HR. Це робиться шляхом додавання вихідних даних шару деконволюції та вибірки найближчого сусіда.

Додається новий залишковий блок під назвою PConv. Запропонований блок допомагає досягти хорошого компромісу між точністю та продуктивністю на основі результатів.

Подібно до EDSR, пакетної нормалізації уникають, щоб запобігти повторного центрування та повторного масштабування, оскільки виявляється, що це шкідливо. Це пов'язано з тим, що Super resolution є завданням регресії, і, таким чином, цільові результати сильно корелюють зі статистикою першого порядку вхідних даних.

2.2.4. Методи з використанням рекурсивних мереж

Рекурсивні мережі використовують спільні параметри мережі на згорткових шарах, щоб зменшити обсяг пам’яті.

Прикладом методу з використанням рекурсивних мереж є Deep Recursive Convolutional Network (DRCN). Глибока рекурсивна згортка (DRCN) передбачає застосування одного і того ж шару згортки кілька разів. Згорткові шари в залишковому блоці є спільними (рисунок 2.12).

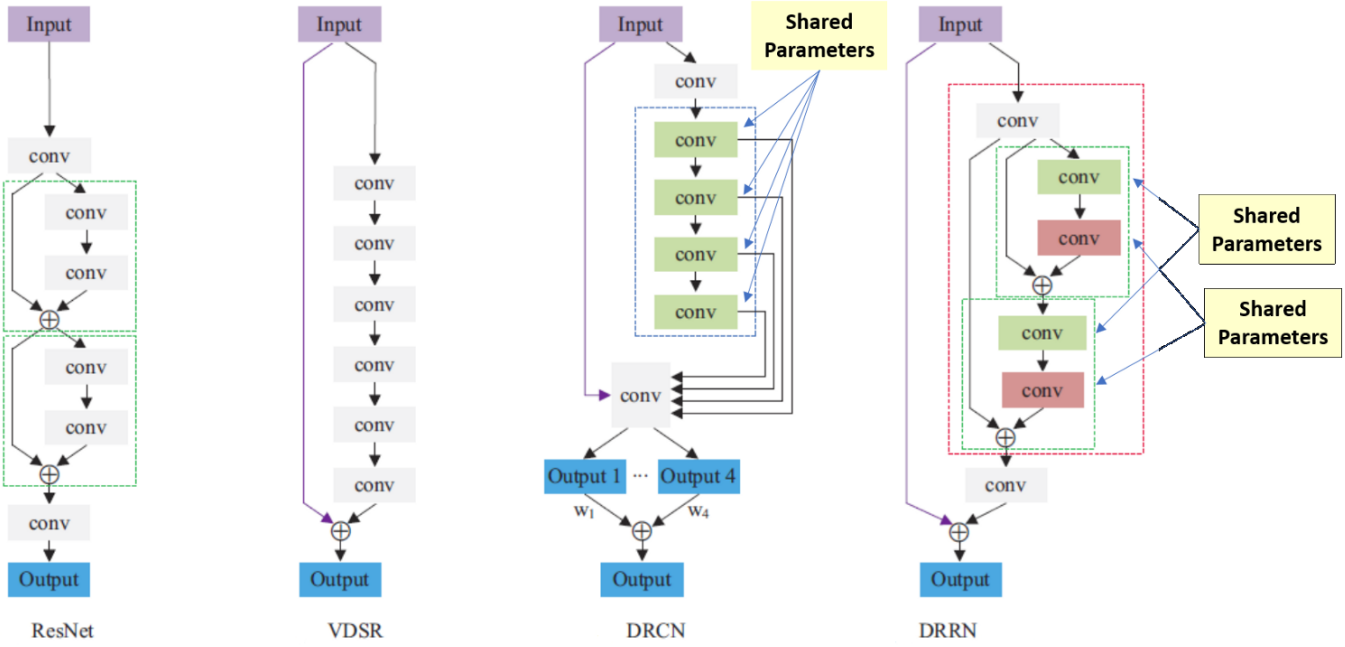

Рисунок 2.12 – Порівняння DRCN з іншими методами

Вихідні дані з усіх проміжних спільних згорткових блоків разом із вхідним сигналом надсилаються на рівень реконструкції, який генерує зображення з

високою роздільною здатністю, використовуючи всі входи [12]. Оскільки для генерування вихідних даних використовується кілька входів, цю архітектуру можна розглядати як ансамбль мереж(рисунок 2.13).

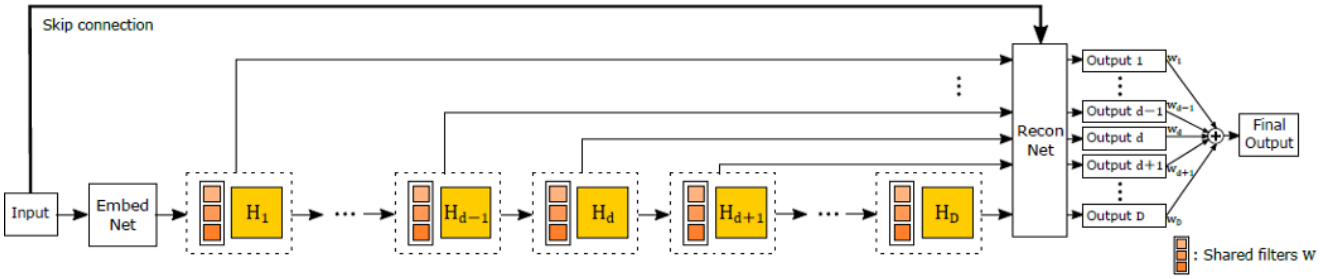

Рисунок 2.13 – Обробка вхідних даних в DRCN

2.2.5. Методи з використанням Progressive Reconstruction Networks

Згорткові нейронні мережі (CNN) зазвичай виконують обробку зображення за один прохід, але отримання зображення з високою роздільною здатністю з великим коефіцієнтом масштабування (наприклад, 8-кратним) є складним завданням для нейронної мережі. Щоб вирішити цю проблему, деякі архітектури мережі збільшують роздільну здатність зображень поетапно.

Прикладом такого методу є метод з використанням Laplacian Pyramid Super-Resolution Network (LapSRN). LAPSRN, або MS-LAPSRN, складається з піраміди Лапласа, за допомогою якої можливо збільшити зображення у 2, 4 і 8 разів за допомогою покрокового підходу.

LAPSRN складається з кількох етапів. Мережа складається з двох гілок: гілки вилучення ознак (Feature Extraction Branch) та гілки реконструкції зображення (Reconstruction Branch) (рисунок 2.14). Кожен ітераційний етап складається з блоку вбудовування функцій і блоку підвищення дискретизації функцій, як показано на малюнку нижче. Вхідне зображення пропускається через шар вбудовування об’єктів для вилучення об’єктів у просторі з низькою роздільною здатністю, яке потім підвищується за допомогою згортки транспонування. Вивчений вихід є залишковим зображенням, яке додається до

інтерпольованого входу для отримання зображення з високою роздільною здатністю. Вихід блоку підвищення дискретизації функцій також передається до наступного етапу, який використовується для уточнення виводу високої роздільної здатності цього етапу та його масштабування до наступного рівня. Оскільки вихідні дані з нижчою роздільною здатністю використовуються для уточнення подальших етапів, існує спільне навчання, яке допомагає мережі працювати краще [13].

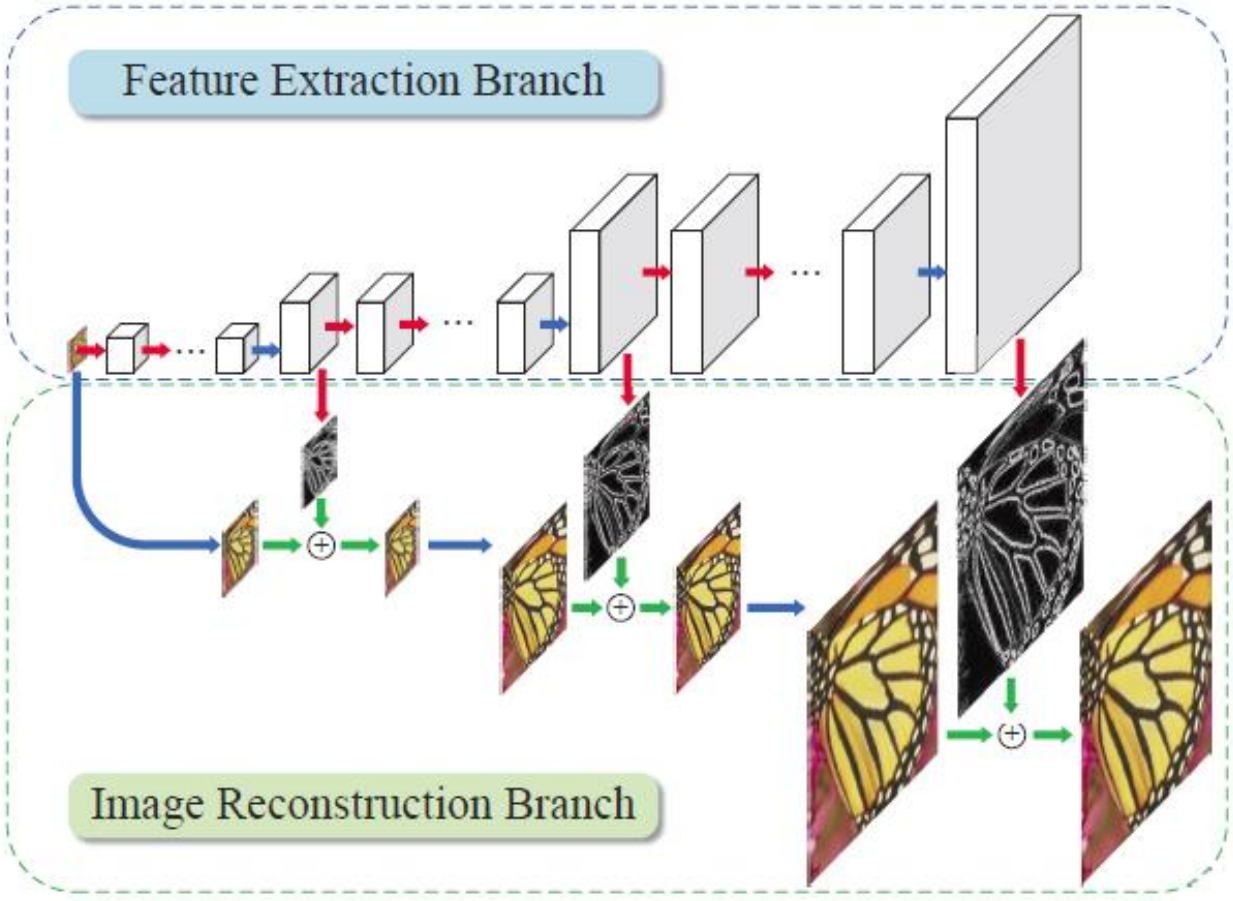

Рисунок 2.14 – Структура Laplacian Pyramid Super-Resolution Network (LapSRN)

Щоб зменшити обсяг пам’яті мережі, параметри вбудовування функцій, підвищення дискретизації функцій тощо розподіляються між етапами рекурсивно (рисунок 2.15).

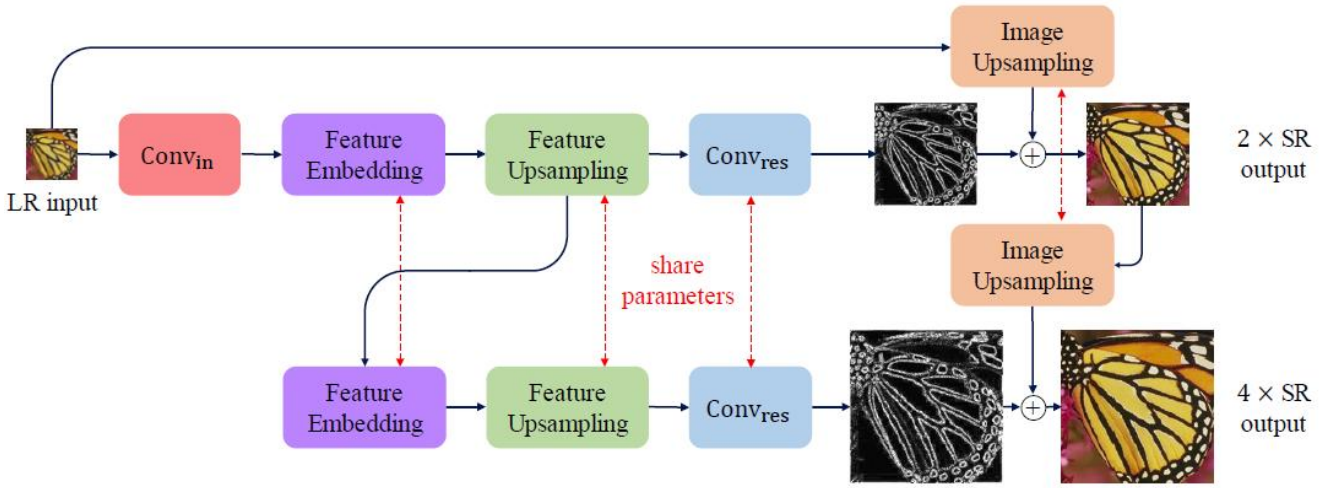

Рисунок 2.15 –Розподіл параметрів в LapSRN

Усередині блоку вбудовування ознак індивідуальний залишковий блок складається із спільних параметрів згортки (рисунок 2.16), щоб ще зменшити кількість параметрів [13].

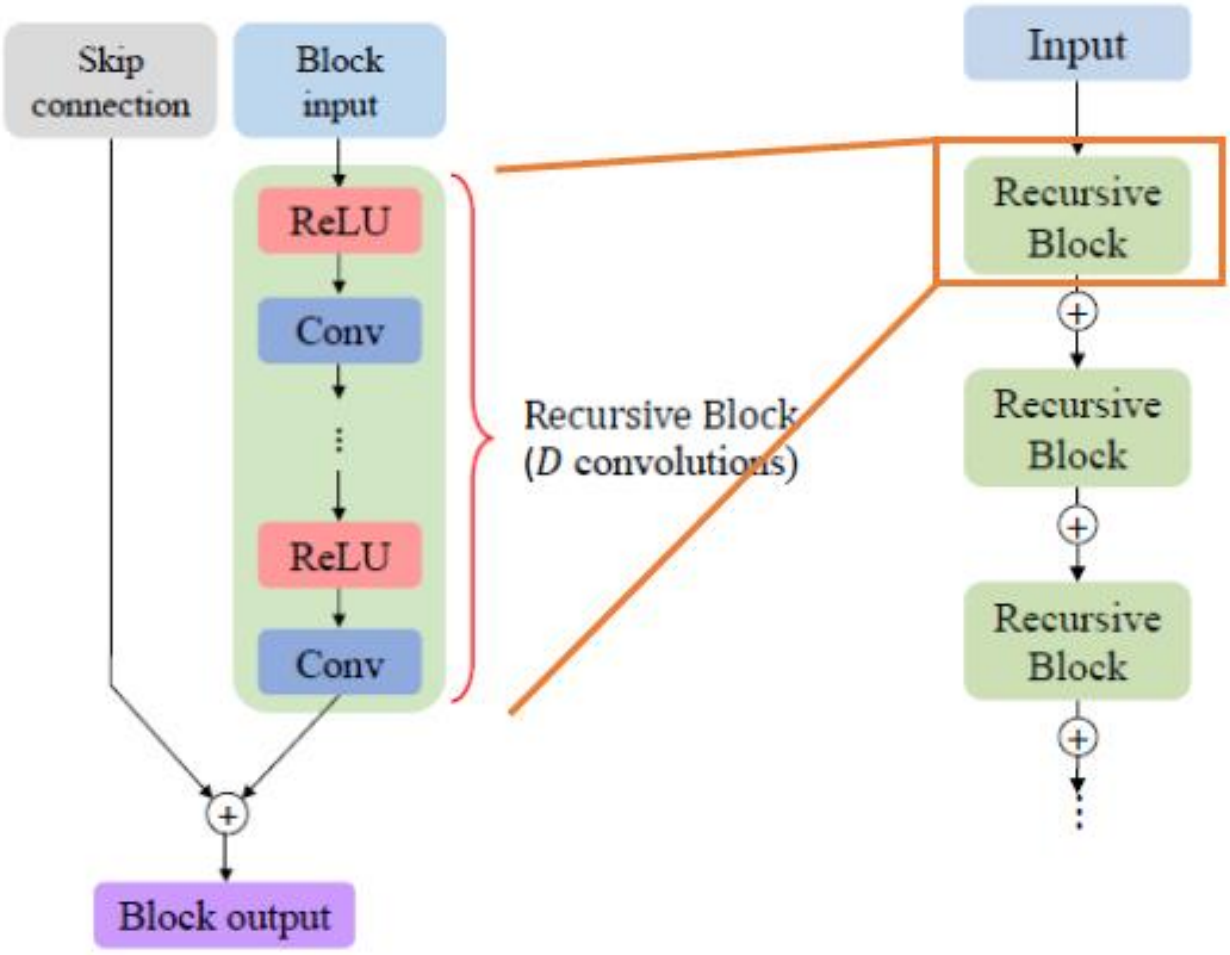

Рисунок 2.16 – Розподіл параметрів в LapSRN

Оскільки кожен вхід LR може мати кілька представлень HR, функція втрат L2 виробляє згладжений вихід для всіх уявлень, таким чином, зображення не виглядають різкими. Для вирішення цього використовується функція втрат Шарбоньє, яка краще справляється з викидами.

2.2.6. Методи з використанням багатогілкових нейронних мереж

З вищеописаних методів помітно тенденцію: більш глибокі мережі дають кращі результати. Але навчання глибших мереж є важким через проблему потоку інформації. Залишкові мережі вирішують це певною мірою за допомогою різноманітних оптимізацій. Багатогілкові мережі працюють над покращенням інформаційного потоку, маючи кілька гілок, через які інформація може проходити, що призводить до об’єднання інформації з кількох сприйнятливих полів і, отже, кращого навчання.

Прикладом такої мережі є Cascaded Multi-Scale Cross-Network (CMSC). Подібно до інших, каскадна багатомасштабна перехресна мережа (CMSC) має шар вилучення ознак, каскадні підмережі та рівень реконструкції (рисунок 2.17).

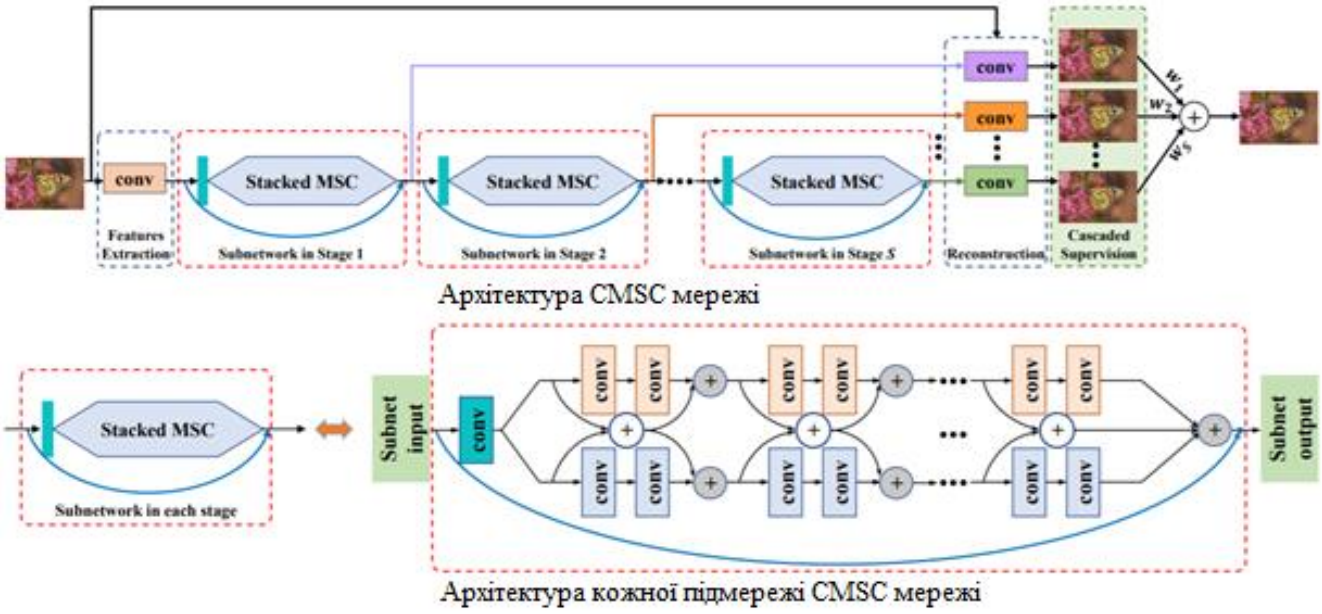

Рисунок 2.17 – Архітектура мережі CMSC

Каскадна підмережа складається з двох гілок. Кожна гілка має різні розміри фільтрів, і, отже, призводить до різного сприйнятливого поля. Злиття інформації з різних сприйнятливих полів по всьому модулю призводить до кращого потоку інформації. Кілька блоків MSC укладаються один за одним, щоб поступово зменшувати різницю між вихідним і кадровим зображенням ітераційно. Вихідні дані з усіх блоків разом передаються в блок реконструкції, щоб отримати остаточний результат у виглядрі HR-зображення [14].

Іншим прикладом є Information Distillation Network (IDN). IDN пропонується для досягнення швидких і точних результатів для завдання підвищення роздільної здатності. Як і інші багатогілкові мережі, IDN використовує можливість кількох гілок для покращення інформаційного потоку в глибокій мережі.

Архітектура IDN складається з FBlock для вилучення функцій, кількох DBlocks і RBlock для виконання транспонованої згортки для досягнення засвоєного підвищення масштабу. Структура блоку покращення зображена на рисунку 2.18.

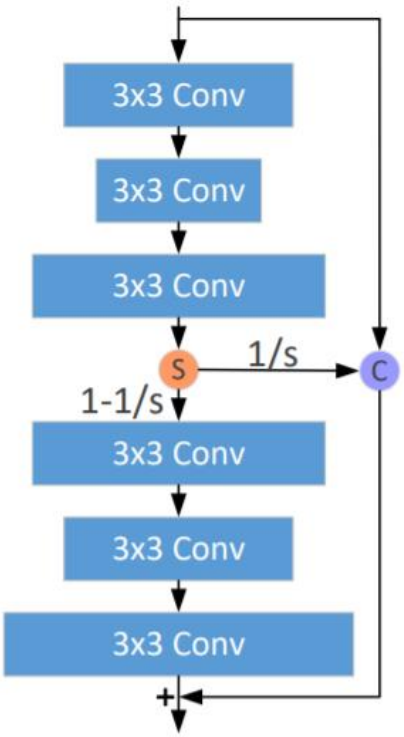

Рисунок 2.18 – Структура блоку покращення мережі IDN

Блок покращеня працює наступним чином. Вхідні дані пропускаються через три згорткові фільтри розміром 3×3, а потім розрізаються. Одна частина фрагмента з'єднується з початковим введенням, щоб перейти через ярлик до кінцевого шару. Зріз, що залишився, пропускають через інший набір згорткових фільтрів розміром 3×3. Остаточний вихід генерується шляхом підсумовування вхідних даних і кінцевого шару. Наявність такої структури допомагає одночасно отримувати як короткочасну, так і далекобічну інформацію.

2.2.7. Методи з використанням мереж Attention-Based

Мережі, про які йшлося вище, надають однакове значення всім просторовим локаціям і каналам. Якщо приділяти вибіркову увагу різним регіонам зображення, можна отримати набагато кращі результати.

Прикладом такої мережі є SelNet. SelNet пропонує новий блок вибору в кінці згорткових блоків, який допомагає вирішувати, яку інформацію передавати, вибірково. Модуль вибору складається з активації ReLu з наступною згорткою 1×1 і сигмовидним стробуванням (рисунок 2.19). Одиниця вибору – це множення модуля вибору та ідентифікаційного з'єднання.

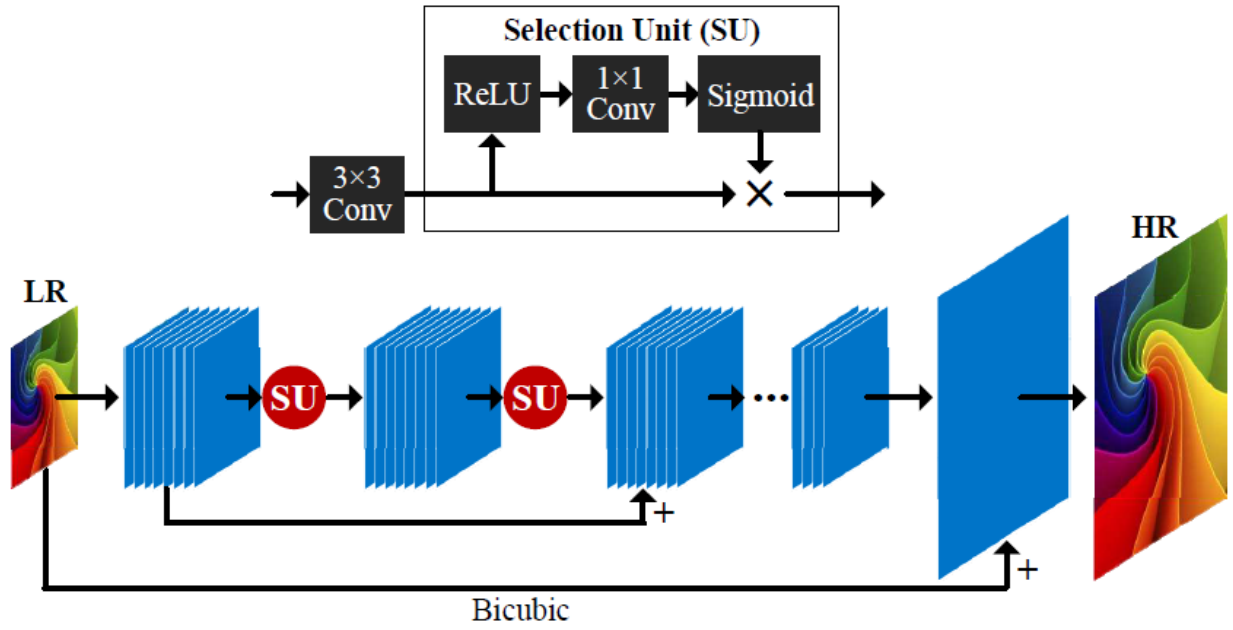

Рисунок 2.19 – Струкутра мережі SelNet

Субпіксельний шар (подібний до ESPCN) зберігається до кінця мережі для досягнення засвоєного збільшення масштабу. Мережа вивчає залишкове кадрове зображення, яке потім додається до інтерпольованого входу для отримання остаточного кадрового зображення [15].

Іншим прикладом є Residual Channel Attention Networks (RCAN). Дана мережа містить модулі RIR, які застосовуються для оброблки різних каналів.

Вхід дані у RCAN пропускаються через один згортковий фільтр для вилучення ознак, який потім обходиться до кінцевого шару за допомогою довгого з'єднання з пропуском. Довге з'єднання з пропуском додається для передачі низькочастотних сигналів з LR зображення, тоді як основна мережа (тобто RIR) зосереджується на захопленні високочастотної інформації.

RIR складається з кількох блоків RG, кожен з яких має структуру, показану на рисунку 2.20. Кожен блок RG має кілька модулів RCAB разом із з'єднанням пропуску, яке називається коротким з'єднанням пропуску. Такі модулі призначеі для передачі низькочастотного сигналу [16].

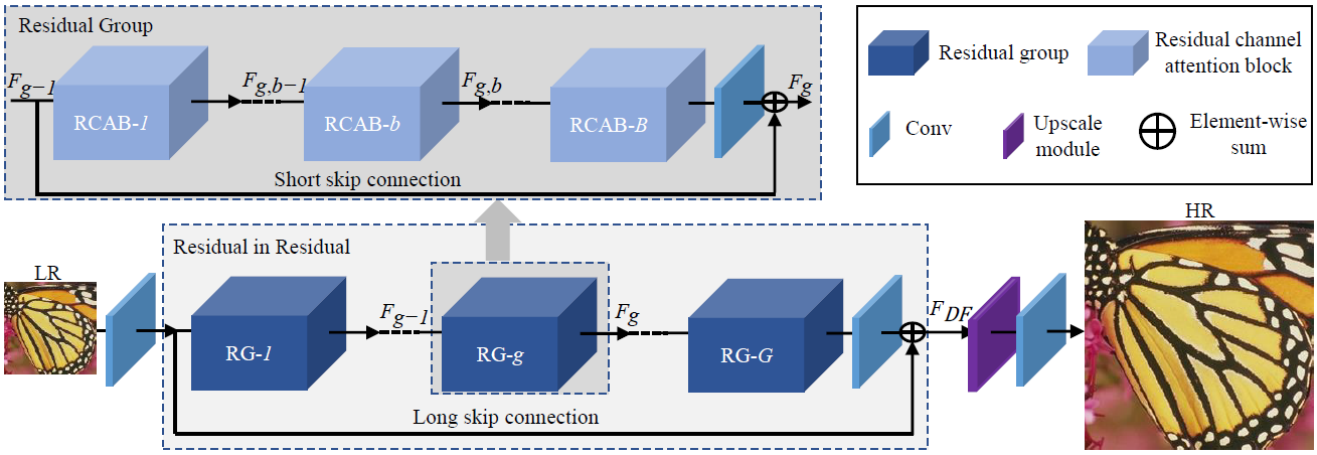

Рисунок 2.20 – Структура блоку RIR

RCAB має структуру, що складається з модуля GAP для досягнення уваги каналу, подібну до блоків Squeeze та Excite у SqueezeNet. Канальна увага помножується на вихід із сигмовидної функції стробування згорткового блоку. Ці

вихідні дані потім додаються до вхідних даних, щоб отримати кінцевий результат обробки модулем RCAB (рисунок 2.21).

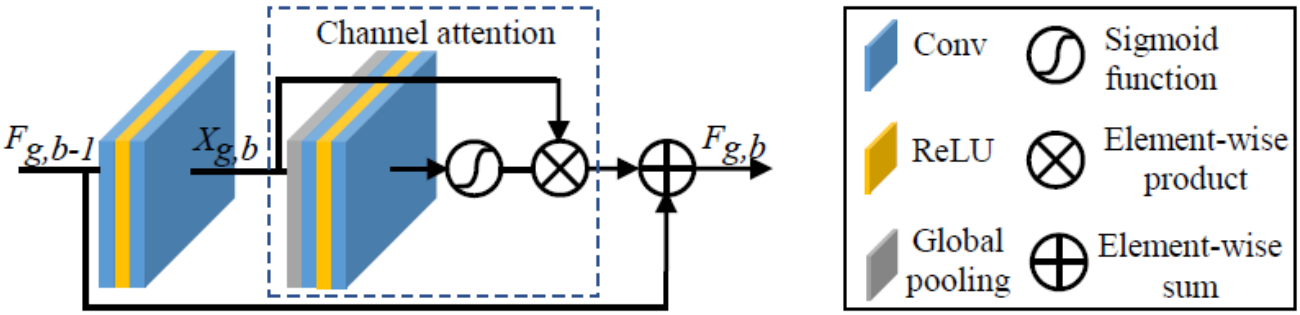

Рисунок 2.21 – Структура блоку RCAB

2.3. Модифікація «Super resolutiom from a single image»

В даній роботі необхідно використати модифікацію SR-алгоритму, яка б дозволяла підвищити роздільну здатність зображення без допоміжних вхідних даних, тому запропоновано модифікацію, яка отримала назву «Super Resolution from a Single Image». Ця модифікація поєднує в собі переваги класичних SR-алгоритмів та SR-алгоритмів, які базуються на послідовності навчальних прикладів. Зображення з низькою роздільною здатністю конвертується в зображення з високою роздільною здатністю без будь-якої іншої додаткової вхідної інформації.

В означеній модифікації SR-алгоритму використовується така властивість зображення, що невеликі фрагменти вихідного зображення мають тенденцію до надлишкового повторення в рамках цього ж зображення, до того ж таке повторення зберігається і в отриманому зображенні з підвищеною роздільною здатністю у різних масштабах.

Природні зображення, як правило, містять повторюваний візуальний вміст. Зокрема, невеликі (наприклад, 5 × 5) ділянки зображення в природному зображенні мають тенденцію багаторазово повторюватися всередині зображення, як в одному масштабі, так і в різних масштабах. Це спостереження є основою для

даного алгоритму для підвищення роздільної якось зображення, а також для інших алгоритмів комп’ютерного зору (наприклад, завершення зображення, зменшення шуму тощо).

На рисунок 2.22 схематично проілюстровано, що мається на увазі під повторенням певних ділянок у межах і поперек масштабів одного зображення. Ділянка вхідного зображення повторюється в іншому масштабі, якщо він виглядає «як є» (без розмивання, підвибірки чи зменшення масштабу) у зменшеній версії зображення. Знайшовши подібний фрагмент у зображенні меншого масштабу, можливо отримати його батьківський елемент високої роздільної здатності з вхідного зображення.

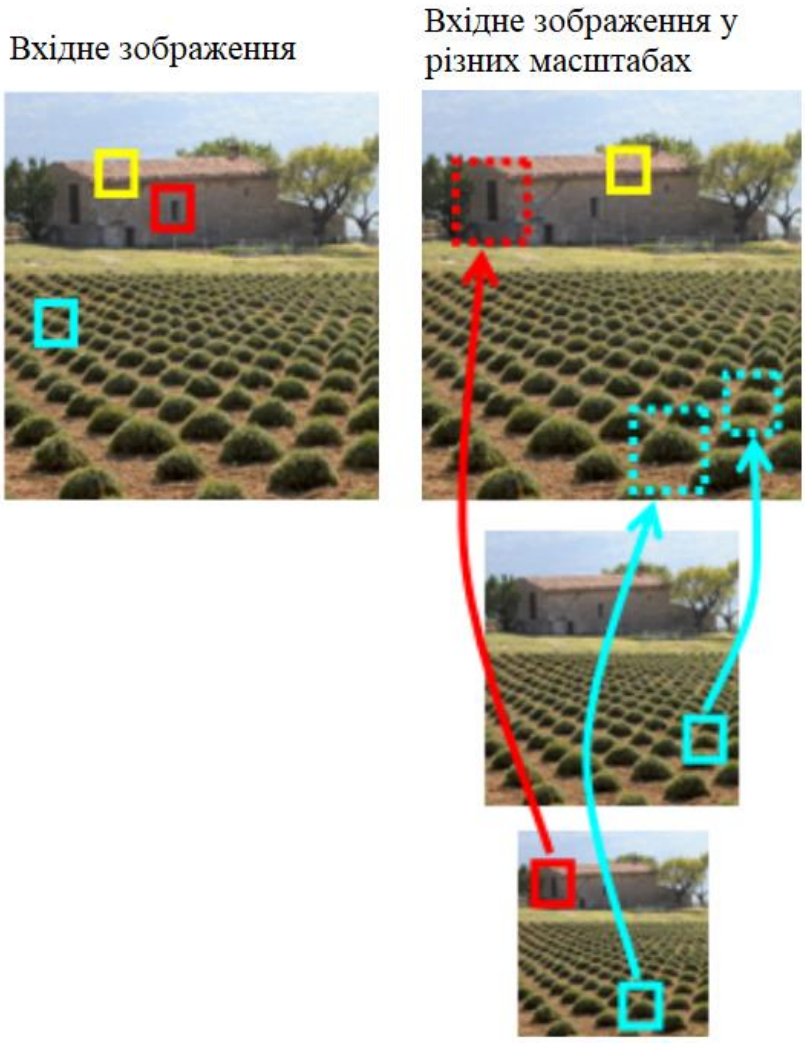

Рисунок 2.22 – Приклад повторення ділянок зображення

Кожна ділянка зображення із низькою роздільною здатністю та його батьківський елемент з високою роздільною здатністю утворюють «пару ділянок із низькою/високою роздільною здатністю» (позначені стрілками на рисунку). Батьківський елемент із високою роздільною здатністю знайденої ділянки з низькою роздільною здатністю надає вказівку на те, як може виглядати (невідомий) батьківський елемент із високою роздільною здатністю вихідної ділянки зображення. Наявність таких пар є основою для SR-алгоритмів на основі прикладів, навіть без зовнішньої бази даних. Тобто головна відмінність даної модифікації алгоритму від класичних SR-алгоритмів в тому, що інформація для підвищення роздільної здатності береться з вхідного зображення.

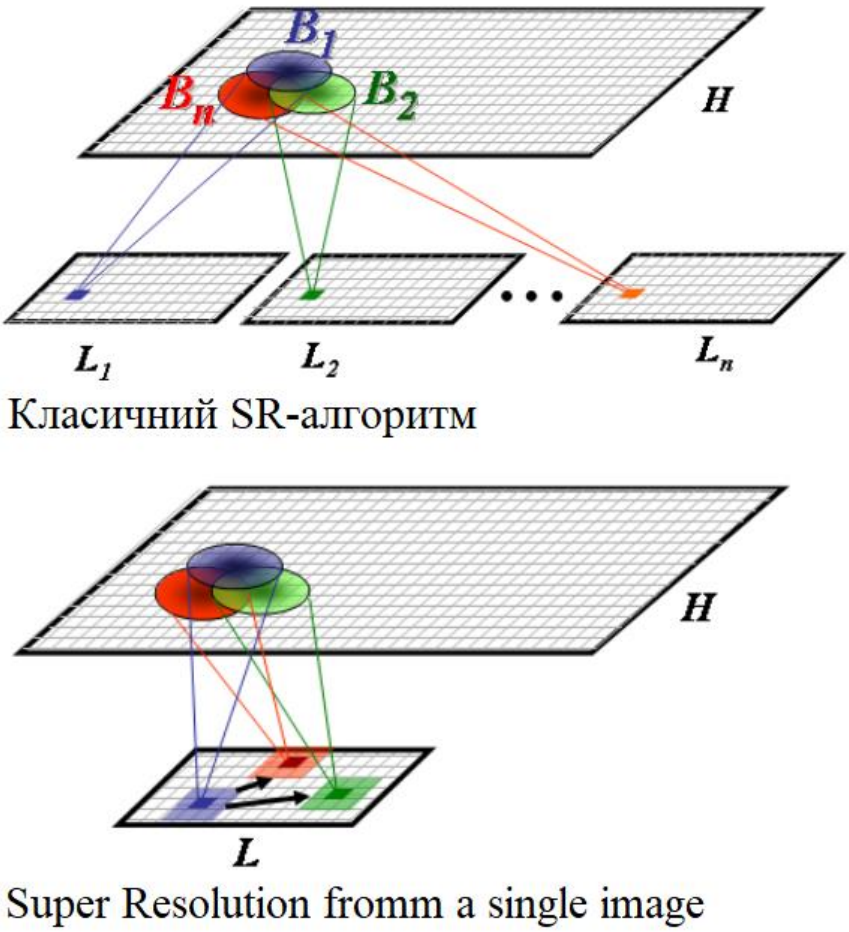

Рисунок 2.23 – Різниця між класничми SR-алгоритмами там модифікацією «Super resolutiom from a single image»

Однак для того, щоб цей підхід був ефективним, достатня кількість таких повторюваних ділянок має існувати в різних масштабах одного зображення. Ділянки з прикладу були вибрані великими для ілюстрації та відображені на зображенні на чіткій повторюваній структурі. Однак, коли використовуються набагато менші ділянки зображення, наприклад, 5 × 5, такі повтори відбуваються в набагато менших масштабах зображення, тобто такі повторювальні структури іноді навіть неможливо сприйняти візуально. Це пов’язано з тим, що дуже маленькі плями часто містять лише край, кут тощо. Такі «латки» зустрічаються багато в декількох масштабах зображення майже будь-якого природного зображення.

Більше того, завдяки перспективній проекції камер, зображення мають тенденцію містити специфічну для сцени інформацію в розмірах, що зменшуються (зменшуються до горизонту), таким чином повторюючись у кількох масштабах одного і того ж зображення.

Послідовність кроків алгоритму:

1. Виконати перетворення вхідного зображення з моделі RGB в YCbCr.

1.1. Для кожного пікселя цільового зображення виконати наступне перетворення:

$Y = 0.299 * R + 0.587 * G + 0.114 * B,$

$C_b = 0.564 * (B - Y),$

$C_r = 0.712 * (R - Y),$

де: R, G, B - значення яскравості каналів пікселя вхідного зображення;
Y, Cb, Cr – значення яскравості каналів пікселя цільового зображення [17].

2. Виконати поділ зображення, яке було отримано на першому кроці, на три кадра, де кожен новий кадр відповідає одному з каналів YCbCr-моделі.

3. Для кадрів, що відповідають каналам Cb і Cr, виконати операцію збільшення роздільної здатності до цільового значення, використовуючи метод бікубічної інтерполяції.

4. Для кадру, який відповідає каналу Y, побудувати піраміду зображень.

4.1. Для вхідного кадру низької роздільної здатності $L = I_0$, побудувати послідовність кадрів нижчої роздільної здатності $I_{-1}, \ldots, I_{-n}$, де n – рівень піраміди. При побудові кадрів нижчої роздільної здатності використовується оператор розмивання Гауса. Масштаб кожного наступного кадру в піраміді розраховується як добуток масштабу попереднього кадру на коефіцієнт α, де $\alpha = 2^{1/3}$ [18].

4.2. Для кадру $L = I_0$, побудувати послідовність порожніх кадрів вищого дозволу $I_1, \ldots, I_n$, де $I_n = H$, використовуючи аналогічне масштабування.

5. Відновити зображення на вершині піраміди.

5.1. Для кожного пікселя вихідного зображення та області, яка його оточує (розміром 5*5 пікселів), виконати пошук схожих ділянок в усіх зменшених копіях вихідного зображення. Для пошуку використовується алгоритм «Approximate Nearest Neighbor search».

5.2. При знаходженні схожої ділянки на зображенні меншого масштабу рівня -l зберігається її образ з початкового зображення і копіюється у відповідне місце в зображенні рівня l2 .

5.3. Для кожного пікселя верхнього зображення піраміди та області, яка його оточує (розміром 10*10 пікселів), виконати пошук ділянки, яка передувала відновленій на тій же позиції, але на зображеннях меншого масштабу. Для знайденої області, використовуючи алгоритм загаданий в кроці 5.1, знайти k схожих областей на тому ж рівні піраміди. Області, які були знайдені на цьому кроці формують базу зображень для використання класичного SR-алгоритму, в якому для підвищення роздільної здатності потрібно кілька зображень.

Використовуючи класичний SR-алгоритм та знайдені області, відновити вихідну ділянку на цільовому зображенні (рис 2.23).

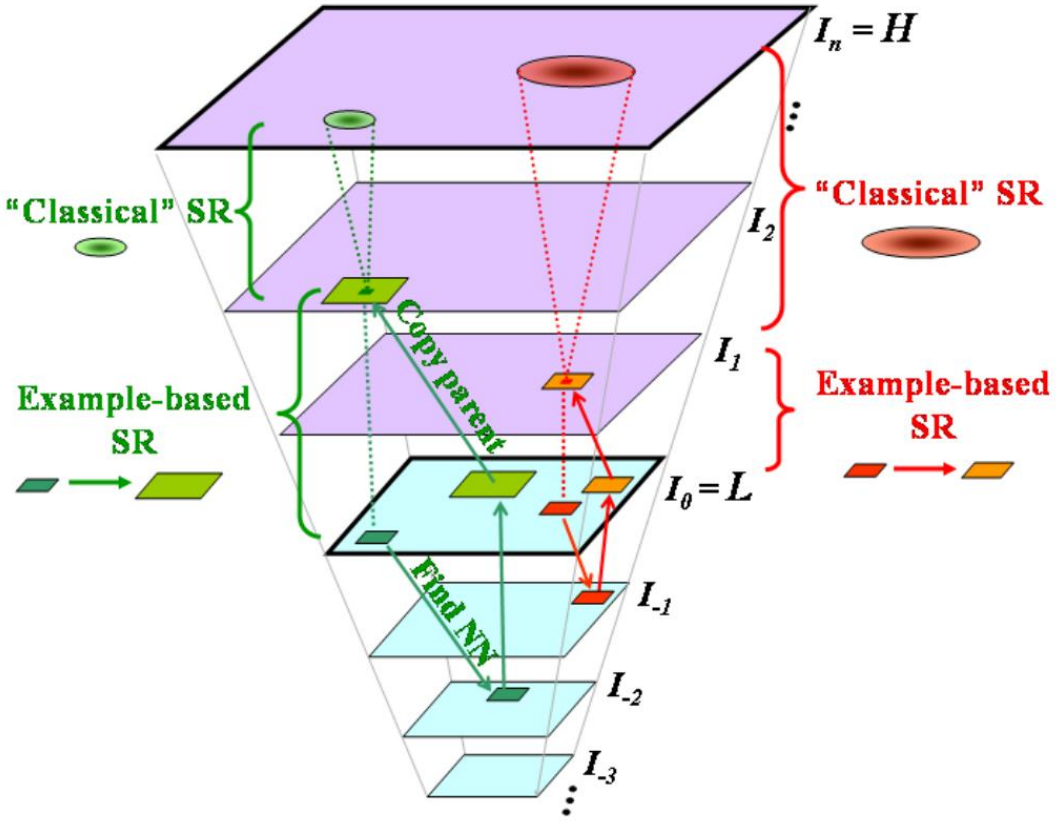

Рисунок 2.23 – Піраміда зображень, за допомогою якої формується база зображень для використання класичного SR-алгоритму

6. Перевірити, чи досягнуте цільове значення роздільної здатності. Якщо ні, то виконати крок 4 по відношенню до кадру на вершині піраміди. Якщо так, то перейти до наступного кроку.

7. Вирівняти роздільну здатність отриманого кадру для отримання цільової роздільної здатності.

8. Виконати об’єднання кадрів, які відповідають каналам YCbCr.

9. Виконати перетворення отриманого зображення з моделі YCbCr в модель RGB.

9.1. Для кожного пікселя цільового зображення виконати наступне перетворення:

$$R = Y + 1.402 * C_r,$$

$$G = Y - 0.344 * C_b - 0.714 * C_r,$$

$$B = Y + 1.772 * C_b,$$

де: R, G, B - значення яскравості каналів пікселя вхідного зображення;
Y, Cb, Cr – значення яскравості каналів пікселя цільового зображення [17].

Схематичне подання алгоритму зображено на рисунку 2.24.

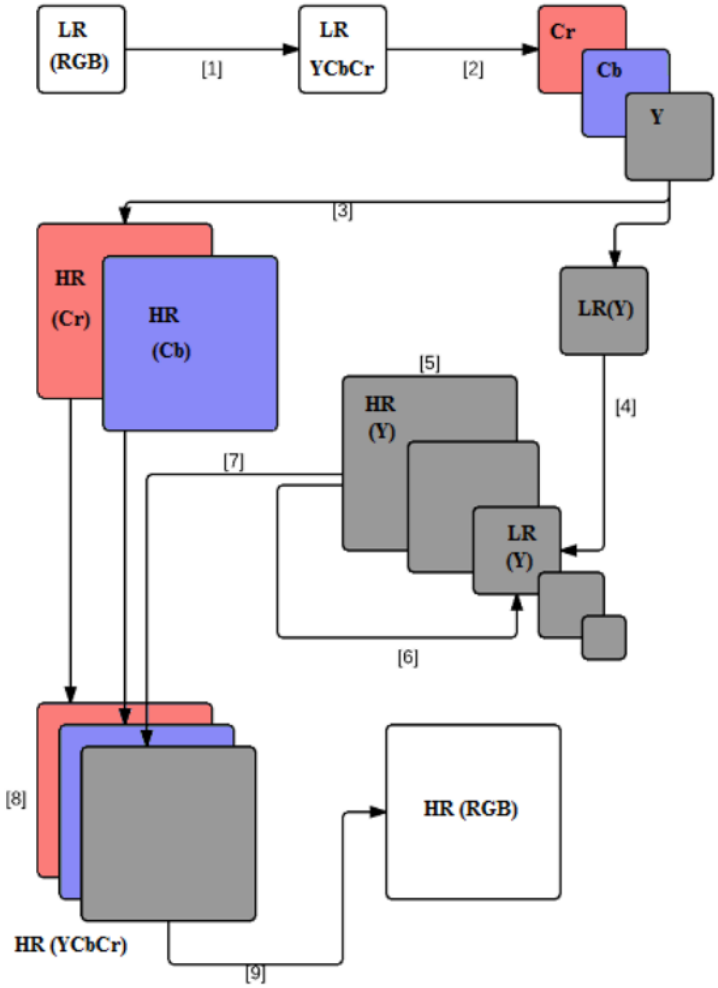

Рисунок 2.24 – Схематичне зображення алгоритму «Super resolutiom from a single image»

В даному алгоритмі використовується бікубічна інтерполяція зображення. У комп'ютерних науках бікубічна інтерполяція є розширенням кубічної інтерполяції для інтерполяції точок даних на регулярній двомірній сітці. Інтерпольована поверхня є більш гладкою, ніж відповідні поверхні, отримані білінійною інтерполяцією або інтерполяцією найближчого сусіда. Бікубічна інтерполяція може супроводжуватися за допомогою багаточленів Лагранжа, кубічних сплайнів або алгоритмами інтерполяції кубічної згортки.

При обробці зображення бікубічну інтерполяцію часто вибирають порівняно з білінійною інтерполяцією або інтерполяцією найближчого сусіда. На відміну від білінійної інтерполяції, яка враховує лише 4 пікселі (2 × 2), бікубічна інтерполяція розглядає 16 пікселів (4 × 4). Пікселі, які ближче до того, який потрібно оцінити, мають більшу вагу в порівнянні з тими, які знаходяться далі. Тому найдальші пікселі мають найменшу вагу. Результати бікубічної інтерполяції набагато кращі в порівнянні з алгоритмами найближчого сусіда або білінійними алгоритмами. Це може бути пов’язано з тим, що під час оцінки потрібного значення враховується більша кількість відомих значень пікселів. Таким чином, це робить його одним із головних стандартних методів інтерполяції. До недоліків даного алгоритму можна віднести те, що він є повільнішими, ніж лінійна інтерполяція.

Для області схожих ділянок використовується алгоритм «Approximate Nearest Neighbor search». Приблизний алгоритм пошуку найближчого сусіда дозволяє повертати точки, відстань яких від запиту не більше ніж в n-разів на відстань від запиту до найближчих точок. Привабливість цього підходу полягає в тому, що в багатьох випадках приблизний найближчий сусід майже такий же хороший, як і точний. Зокрема, якщо вимірювання відстані точно фіксує поняття якості користувача, то невеликі відмінності в відстані не мають значення.

Проблема найближчого сусіда є проблемою значного значення в таких областях, як статистика, розпізнавання образів і стиснення даних. Проблема найближчого сусіда може бути розв’язана за $O(\log n)$ часу в 1-вимірному просторі шляхом бінарного пошуку та за $O(\log n)$ часу на площині за допомогою діаграм

Вороного та розташування точки. Однак із збільшенням розмірності складність вирішення проблеми найближчого сусіда, у часі чи просторі, здається, зростає надзвичайно швидко. Було запропоновано рандомізований алгоритм очікуваного часу $O(\log n)$ для пошуку найближчих сусідів у фіксованому вимірі на основі обчислення діаграм Вороного випадково відібраних підмножин точок (дерево RPO) [19]. Однак у гіршому випадку простір, необхідний його алгоритму, зростає приблизно як $O(n^{\left[\frac{d}{2}\right]+\delta})$, а це занадто багат. Було виявлено, що пошук найближчого сусіда може бути вирішений у лінійному просторі та ледь сублінійному часі $O(n^{f(d)})$, де $f(d) = (\log(2^d - 1))/d$, але такий малий асимптотичне поліпшення насправді не має практичної цінності. Найбільш практичним підходом до проблеми, відомим для більших розмірів, є алгоритм k-d дерева, розроблений Фрідманом, Бентлі та Фінкель [20]. Очікуваний час роботи дерева k-d є логарифмічним, але це справедливо лише за досить обмежувальних припущень щодо вхідного розподілу. Час роботи алгоритму k-d дерева може бути таким же поганим, як і лінійний час для певних вхідних даних. Тож якщо розглядати приблизні найближчі сусіди, а не точні найближчі сусіди, можна досягти ефективної продуктивності як для простору, так і для часу запиту, незалежно від розподілу вхідних даних. Враховуючи будь-яку константу > 0, ми говоримо, що точка p є (1 + e) - найближчим сусідом точки запиту q, якщо відношення відстаней від q до p і від q до її найближчого сусіда не більше (1 + e).

2.4. Архітектурні особливості програмного забезпечення у вигляді веб-сервісу

Веб-сервіс є стандартизованим середовищем для поширення зв'язку між клієнтськими та серверними додатками у WWW (World Wide Web). Веб-сервіс — це програмний модуль, призначений для виконання певного набору завдань.

Веб-сервіси в хмарних обчисленнях можна шукати в мережі, а також відповідним чином викликати. При виклику веб-сервіс зможе надати клієнтові функціональні можливості, який викликає цю веб-службу.

Діаграма, зображена на рисунку 2.25, показує спрощене уявлення про те, як насправді працюватиме веб-сервіс. Клієнт буде викликати серію викликів веб-сервісу через запити до сервера, на якому буде розміщена фактична веб-служба.

Ці запити здійснюються за допомогою так званих віддалених викликів процедур. Виклики віддалених процедур (RPC) – це виклики методів, які розміщуються відповідною веб-службою.

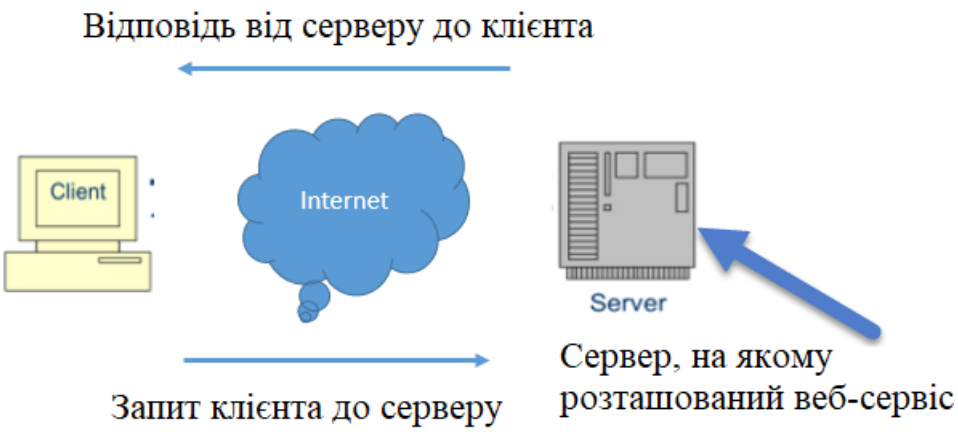

Рисунок 2.25 – Схема роботи клієнт-серверної архітектури

Сучасні бізнес-додатки використовують різноманітні платформи програмування для розробки веб-додатків. Деякі програми можуть бути розроблені на Java, інші в .Net, а деякі інші на Angular JS, Node.js тощо.

Найчастіше ці різнорідні програми потребують певного зв’язку між ними. Оскільки вони створені з використанням різних мов розробки, стає дійсно важко забезпечити точне спілкування між додатками.

Тут на допомогу приходять веб-сервіси. Веб-сервіси забезпечують загальну платформу, яка дозволяє багатьом додаткам, створеним на різних мовах програмування, мати можливість спілкуватися один з одним.

Переваги використання ПЗ у вигляді веб-сервісу:

1. розкриття функціональних можливостей бізнесу в мережі. Веб-сервіс — це блок керованого коду, який надає певну функціональність клієнтським додаткам або кінцевим користувачам. Цю функцію можна викликати через протокол HTTP, що означає, що її також можна викликати через Інтернет. Сьогодні всі програми знаходяться в Інтернеті, що робить призначення веб-сервісів більш корисним. Це означає, що веб-сервіс може бути в будь-якому місці в Інтернеті та надавати необхідну функціональність за потреби;

2. сумісність між додатками – веб-сервіси дозволяють різним програмам спілкуватися один з одним і обмінюватися даними та послугами між собою. Усі типи програм можуть спілкуватися один з одним. Тому замість написання конкретного коду, який можуть зрозуміти лише конкретні програми, тепер ви можете писати загальний код, який можуть розуміти всі програми;

3. стандартизований протокол, який усі розуміють – веб-сервіси використовують стандартизований галузевий протокол для спілкування. Усі чотири рівні (Service Transport, XML Messaging, Service Description та Service Discovery) використовують чітко визначені протоколи в стеку протоколів веб-сервісів;

4. зменшення вартості зв’язку – веб-сервіси використовують протокол SOAP через HTTP, тому ви можете використовувати наявний недорогий Інтернет для впровадження веб-сервісів.

Недоліки вебсервісів:

1. безпека. Веб-сервіси доступні для громадськості через протокол на основі HTTP. Кожен може отримати доступ до веб-сервісів і користуватися ними. Цю помилку можна уникнути за допомогою механізмів аутентифікації;

2. гарантованість виконання є основною проблемою веб-сервісів, оскільки HTTP, який є гіпертекстовим транспортним протоколом, не є надійним протоколом, тобто не дає жодної гарантії доставки відповіді.

На сьогоднішній день існує два основні види веб-сервісів: SOAP веб-сервіси та RESTful веб-сервіси.

SOAP відомий як транспортно-незалежний протокол обміну повідомленнями. SOAP заснований на передачі даних XML у вигляді повідомлень SOAP. Кожне повідомлення має щось відоме як XML-документ. Лише структура XML-документа відповідає певному шаблону, але не зміст. Найкраща частина веб-сервісів і SOAP полягає в тому, що всі вони надсилаються через HTTP, який є стандартним веб-протоколом.

З чого складається повідомлення SOAP:

1. кожен документ SOAP повинен мати кореневий елемент, відомий як елемент <Envelope>. Кореневий елемент є першим елементом у XML-документі;
2. <Envelope> у свою чергу ділиться на 2 частини. Перший – це заголовок, а наступний – тіло;
3. заголовок містить дані маршрутизації, які в основному є інформацією, яка повідомляє XML-документу, якому клієнту його потрібно надіслати;
4. тіло міститиме фактичне повідомлення.

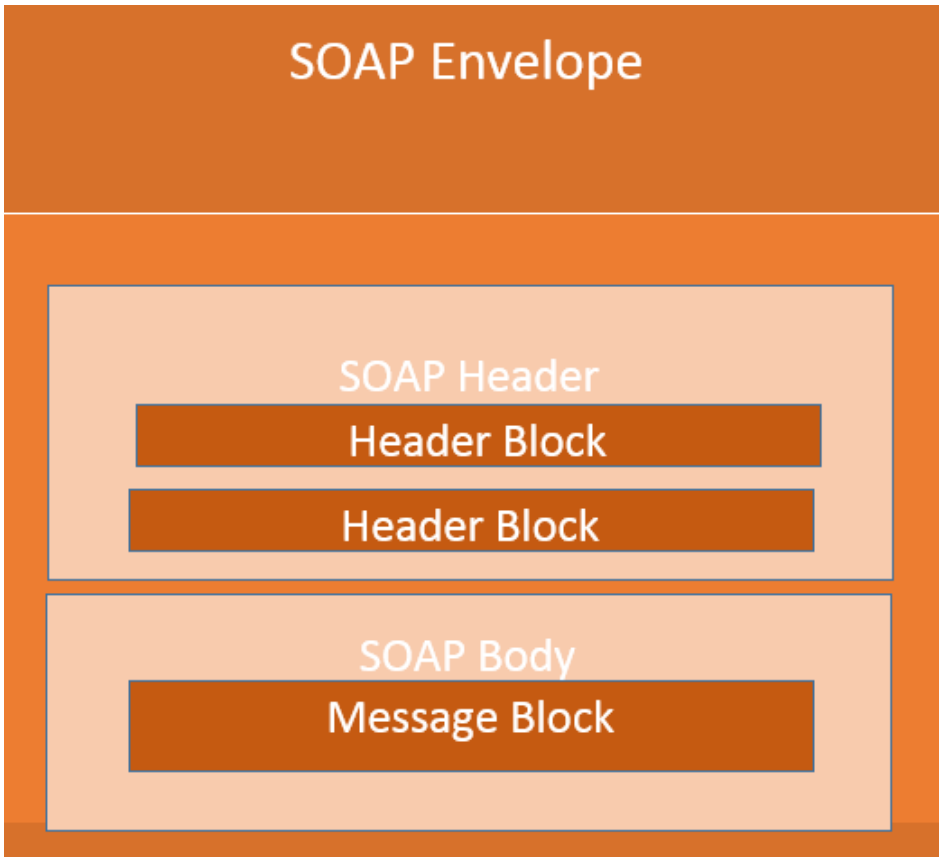

Рисунок 2.26 – Структура SOAP повідомлення

Переваги протоколу SOAP:

1. при розробці веб-сервісів на основі SOAP вам потрібно мати певну мову, яку можна використовувати для веб-сервісів для спілкування з клієнтськими додатками. SOAP є ідеальним середовищем, яке було розроблено для досягнення цієї мети. Цей протокол також рекомендований консорціумом W3C, який є керівним органом для всіх веб-стандартів;

2. SOAP розроблено так, щоб бути незалежним від платформи, а також незалежним від операційної системи. Таким чином, протокол SOAP може працювати з будь-якими програмами на основі мов програмування як на платформі Windows, так і на Linux;

3. SOAP працює за протоколом HTTP, який є протоколом за замовчуванням, який використовується всіма веб-програмами. Таким чином, для роботи веб-сервісів, створених на основі протоколу SOAP, для роботи у всесвітній павутині не потрібно ніяких налаштувань.

Недоліки протоколу SOAP:

1. SOAP зазвичай значно повільніший, ніж інші типи стандартів проміжного програмного забезпечення, оскільки SOAP використовує детальний формат XML. Тому перед створенням додатків на основі SOAP потрібне повне розуміння обмежень продуктивності;

2. SOAP зазвичай обмежується об’єднанням в пул, а не сповіщеннями про події, коли HTTP використовується для транспортування. Крім того, лише один клієнт може скористатися послугами одного сервера в типових ситуаціях;

3. Якщо як транспортний протокол використовується HTTP, зазвичай виникає затримка брандмауера, оскільки брандмауер аналізує транспортний протокол HTTP. Це тому, що HTTP також використовується для перегляду веб-сторінок, і тому багато брандмауерів не розуміють різниці між використанням HTTP у веб-браузері та використанням HTTP у SOAP;

4. SOAP має різні рівні підтримки залежно від підтримуваної мови програмування. Наприклад, SOAP, який підтримується в Python і PHP, не такий потужний, як у Java та .NET.

REpresentational State Transfer (REST) — це архітектурний стиль програмного забезпечення, який визначає набір методів для побудови інтерфейсу програмування веб-додатків (API). REST є одним з найпопулярніших типів API через його простоту та зручний характер.

REST API – це стиль передачі даних, який не залежить від мови програмування. Об’єктами в REST завжди керують з уніфікованого ідентифікатора ресурсу (URI), а корисні навантаження використовуються для надсилання та повернення інформації, яка є занадто великою для обробки як параметра. Корисне навантаження може бути в різних форматах, включаючи HTML, JSON або XML.

Переваги використання архітектури REST:

1. REST API легкий для розуміння та вивчення завдяки своїй простоті, відомому API;

2. завдяки REST API можна організувати складні програми та полегшити використання ресурсів;

3. високим навантаженням можна керувати за допомогою проксі-сервера HTTP і кешу;

4. REST API залежить від кодів, що дає можливість використовувати його для синхронізації даних із веб-сайтом без будь-яких ускладнень;

5. користувачі можуть отримати доступ до тих самих стандартних об’єктів і моделі даних у порівнянні з веб-сервісами на основі SOAP;

6. забезпечує гнучкість форматів шляхом серіалізації даних у форматі XML або JSON;

7. дозволяє використовувати стандартний захист із використанням протоколів OAuth для перевірки ваших запитів REST.

Окрім переваг REST архітектура також має певні недолі, серед яких:

1. відсутність стану: більшість веб-додатків вимагають механізмів контролю стану. Припустимо, ви купуєте веб-сайт, який має механізм створення кошика для покупок. Перед фактичною покупкою необхідно знати кількість

товарів у кошику. Цей тягар підтримки стану лежить на клієнті, що робить клієнтську програму важкою та важкою для обслуговування;

2. проблеми з безпекою. Ось чому REST підходить для загальнодоступних URL-адрес, але він не підходить для конфіденційного передачі даних між клієнтом і сервером.

REST був створений для вирішення проблем SOAP. SOAP — це стандартизований протокол, який надсилає повідомлення за допомогою інших протоколів, таких як HTTP і SMTP. Специфікації SOAP є офіційними веб-стандартами, підтримуваними та розробленими Консорціумом World Wide Web Consortium (W3C). Оскільки SOAP є офіційним протоколом, він має суворі правила та розширені функції безпеки, такі як вбудована відповідність та авторизація ACID (атомність, послідовність, ізоляція, довговічність). Вища складність вимагає більшої пропускної здатності та ресурсів, що може призвести до уповільнення часу завантаження сторінки.

На відміну від SOAP, REST — це не протокол, а архітектурний стиль. Архітектура REST містить набір правил, яких потрібно дотримуватися, якщо ви необхідно створити RESTful веб-сервіс, наприклад, існування без стану та використання кодів статусу HTTP.

Так як REST є вільним архітектурним стилем він має більш гнучку архітектуру. Даний архітектурний стиль складається лише з вільних інструкцій і дозволяє розробникам реалізувати рекомендації по-своєму. REST дозволяє використовувати різні формати повідомлень, такі як HTML, JSON, XML та звичайний текст, тоді як SOAP дозволяє лише XML. REST також є більш легкою архітектурою, тому веб-сервіси RESTful мають кращу продуктивність. Через це він став популярним в епоху мобільних пристроїв, де навіть кілька секунд мають велике значення.

Для розробки вебсервісу покращення якості зображено обрано саме варіант з використанням REST архітектури. Даний вибір зумовлено простотою використання такого архітектурного шаблону при розробці програмного

забезпечення, а також його гнучкістю. Проблеми пов’язані з безпекою вирішені додаванням авторизації.

ВИСНОВКИ ДО ДРУГОГО РОЗДІЛУ МД

В даному розділі дисертації описано загальні принципи роботи SR-алгоритмів, а також проведено порівняння різних існуючих модифікацій SR-алгоритмів. Описано модифікацію алгоритму, яка використовуються у розробленому вебсервісі. Особливістю цієї модифікації є те, що дана модифікація поєднує в собі принципи роботи класичних SR-алгоритмів та SR-алгоритмів, які базуються на бібліотеці навчальних прикладів, завдяки чому зображення з низькою роздільною здатністю конвертується в зображення з високою роздільною здатністю без будь-якої іншої додаткової вхідної інформації. Також описано архітектурні особливості програмного забезпечення у вигляді вебсервісу.

3. СТРУКТУРА ТА ОПИС МОДУЛІВ ВЕБСЕРВІСУ

3.1. Вибір інструментарію програмної реалізації вебсервісу

Для реалізації веб-сервісу з відкритим API було використано кілька різних інструментів.

Для розробки більшої частини веб-сервісу було використано мову програмування C#. C# (вимовляється як Сі Шарп) — це об'єктно-орієнтована мова програмування загального призначення, розроблена компанією Microsoft. Додатки, написані на C#, використовують середовище виконання .NET, бібліотеки класів і в основному власне фреймворк .NET, тому обидві технології часто розглядаються як нероздільні. В даний час .NET є платформою та фреймворком програмування для кросплатформної розробки.

C# — це компільована мова програмування. Щоб запустити програму, вихідний код C# компілюється на проміжну мову (IL). Таким чином, його можна виконувати в різних цільових системах. Під час виконання IL-код компілюється далі в машинний код поточної цільової системи, і ЦП виконує його на ходу.

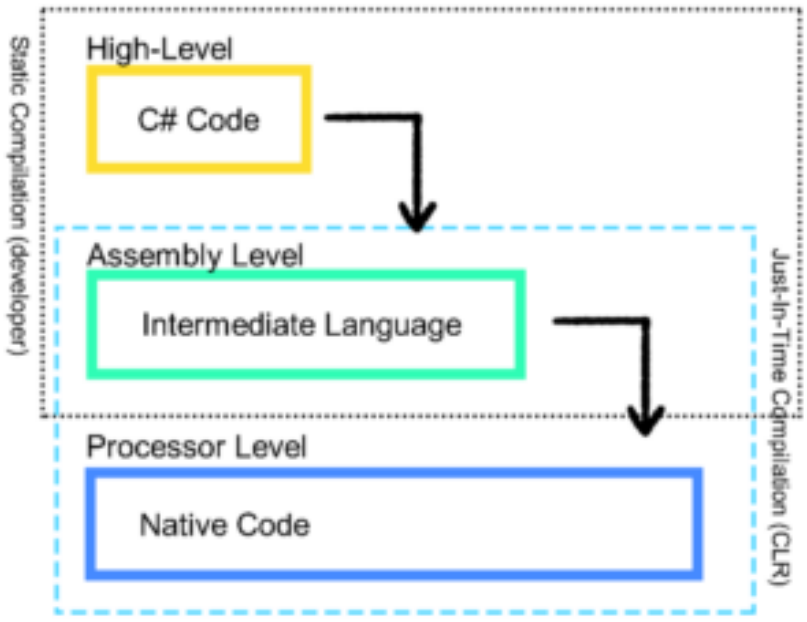

Рисунок 3.1 – Процес компіляції мови C#

Компіляція Just-in-time (JIT) від IL забезпечує універсальність програм C#, які можна запускати скрізь, від Windows до PlayStation. Але це також створює певні накладні витрати в порівнянні, наприклад, з мовою програмування C++, яка компілюється в машинний код цільової системи відразу, перш ніж виконуватися. Але середовище виконання, що використовується з C#, підтримує ряд випадків, коли IL компілюється перед виконанням, щоб забезпечити найкращу можливу продуктивність. Наприклад, усі програми iOS потребують попередньої компіляції.

Переваги використання мови програмування C#:

1. C# - об'єктно-орієнтована мова програмування. З самого початку C# базувався на принципах об’єктно-орієнтованого програмування (ООП). Ця концепція програмування передбачає, що можливо визначити тип і структуру даних, щоб застосувати до них набір стандартних функцій. ООП збирає дані в об’єкти, що полегшує розбивку програми на менші частини, які швидше створювати, керувати та об’єднувати. Використовуючи ООП, об’єктами можна керувати без взаємодії з їхніми внутрішніми атрибутами, описуючи поведінку об’єктів через оголошення класів. Мови ООП створюють програми, які легше тестувати та читати, дають змогу реагувати на будь-які проблеми, що виникають, і загалом означають більш економний підхід до написання коду;

2. мова високого рівня з можливостями доступу до пам'яті. Мова C# вважається мовою високого рівня, оскільки її синтаксис нагадує людську мов;

3. C# це частина платформи .NET. C# сам по собі запропонував би зосередитися на настільному додатку лише для Windows. Завдяки широким можливостям .NET він стає гнучкою мовою, яку можна використовувати на різних платформах. .NET підтримує розробників C# з різними типами середовищ виконання, такими як Mono і CLI від Microsoft. Середовище виконання використовується в програмах .NET для перекладу коду C# у машинні інструкції на будь-якій із підтримуваних ОС. Сумісність мов – це ще одна особливість .NET, яка робить програмування на C# зручним. Сумісність означає, що код C# може взаємодіяти з програмами, написаними на сумісних мовах, таких як C++, F#, Visual Basic і Windows PowerShell. Це дозволяє розробникам використовувати

різні мови в одній збірці. З .NET розробники C# також можуть покладатися на потужний фреймворк .NET Core, розроблений для створення веб-сервісів і програм. Фреймворк поставляється з набором настроюваних компонентів, уніфікованих бібліотек класів та інших функцій, які значно прискорюють процес розробки;

4. C# відноситься до сімейства мов програмування C;

5. C# має вбудований сміттєзбірник Управління розподілом пам'яті є одним із важливих завдань для підтримки продуктивності програми. Для цього в C# є вбудований збірник сміття. сміттєзбірник — це менеджер пам'яті, який відстежує невикористані об'єкти та автоматично звільняє пам'ять. Під час роботи з керованим кодом зазвичай потрібно, щоб розробники написали додатковий код, щоб уникнути витоку пам'яті. Автоматичне керування пам'яттю звільняє розробників від написання команд для відновлення невикористаних об'єктів, очищення пам'яті та виділення її для нових;

6. C# - типобезпечна мова з динамічними можливостями, що означає, що змінна не може змінити свій тип у коді. Типобезпечність гарантує, що змінна буде поводитись у передбачуваний спосіб, і будь-які операції з нею будуть можливі, лише якщо вони збігаються з типом.

Незважаючи на всі переваги C# також має певні недоліки:

1. продуктивність C# не найкраща. Продуктивність мови можна виміряти за часом компіляції та фактичною продуктивністю програми. У порівнянні зі своїм найближчим аналогом Java, C# має подібний час компіляції. Але тести продуктивності програми показують дещо швидші результати для C# . Коли справа доходить до тестів C++ і C#, перший виграє. Існує невелика різниця між тим, як компілюється код C++ і C#. Зазвичай C# проходить дві фази компіляції: перша — це коли код компілюється на проміжну мову на етапі збірки, а друга — компіляція в машинний код, коли він виконується під час виконання. Враховуючи тести, існує значна різниця між мовами, в яких C++ випереджає C#;

2. залежність від платформи .NET. C# багато в чому покладається на ресурси .NET для роботи на різних операційних системах або платформах. Однак

сам по собі він не такий гнучкий, якщо ви не розглядаєте .NET як ваш основний технологічний стек. Це питання порівняння. А саме, скомпільована програма Java може працювати на будь-якій платформі, де доступна віртуальна машина Java. З C# вам потрібно використовувати різні середовища виконання для різних платформ і адаптувати код відповідно до відповідних системних вимог. Хоча .NET надає всі ресурси для цього, відкритий код C# не можна використовувати так легко;

3. складне вивчення. Хоча C# не є найпростішою мовою для вивчення самостійно, використання бібліотек .NET додає ще один рівень складності. Бібліотеки в .NET часто оновлюються.

Також при розробці API було використано фреймворк ASP .NET Core MVC. ASP.NET Core — це нова кросплатформна платформа з відкритим кодом для створення сучасних хмарних додатків, підключених до Інтернету, таких як веб-додатки, програми Інтернету речей та мобільні серверні програми. Програми ASP.NET Core можуть працювати на .NET Core або на.NET Framework. Його було розроблено, щоб забезпечити оптимізовану структуру розробки для програм, які розгортаються в хмарі або запускаються локально. Він складається з модульних компонентів з мінімальними накладними витратами, тому зберігається гнучкість під час створення рішень. ASP.NET Core надає можливість розробляти та запускати кросплатформні програми на Windows, Mac і Linux. Фреймворк є повністю переписаним, що об'єднує раніше окремі ASP.NET MVC і Web API в єдину модель програмування.

Використання ASP . NET Core має наступні переваги:

1. гнучке розгортання: можна включити у певну програму або встановити поряд із користувачем або на всю машину;

2. кросплатформенність: працює на Windows, macOS та Linux; можна перенести на інші ОС. Підтримувані ОС, процесори та сценарії додатків з часом будуть зростати, надані Microsoft, іншими компаніями та окремими особами. Інструменти командного рядка: усі сценарії продукту можна виконувати в командному рядку;

3. сумісність: .NET Core сумісний з .NET Framework, Xamarin і Mono через стандартну бібліотеку .NET;

4. відкритий код. Платформа .NET Core є відкритим вихідним кодом і використовує ліцензії MIT і Apache 2. Документація має ліцензію CC-BY. .NET Core — це проект .NET Foundation;

5. підтримується Microsoft: .NET Core підтримується Microsoft відповідно до підтримки .NET Core.

При розробці програмного забезпечення також варто звертати увагу на вибір IDE (англ. Integrated Drive Electronics) – середовище розробки, тому що від вибору середовища розробки залежить зручність та швидкість процесу розробки. Для розробки запропонованого веб-сервісу було обрано інтегроване середовище розробки Visual Studio 2017 (рисунок 3.2).

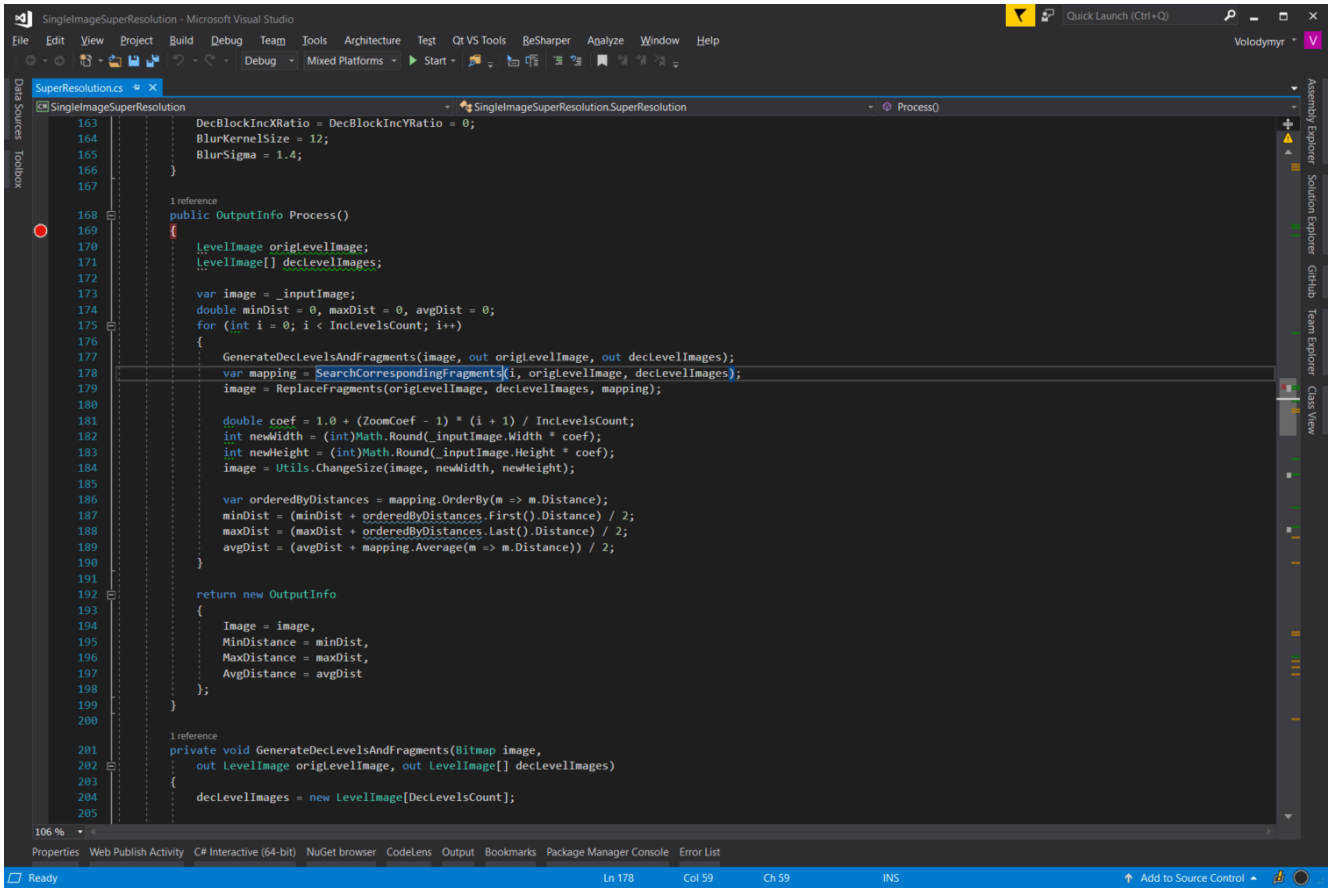

Рисунок 3.2 – Інтегроване середовище розробки Visual Studio 2017

IDE Visual Studio напряму не підтримує жодну мову програмування, рішення чи інструмент; замість цього дане середовище розробки дозволяє підключати функціональні можливості, закодовані у вигляді VSPackage. Після встановлення пакету функція доступна у вигляді сервісу. IDE надає три сервіси: SVsSolution, який надає можливість перераховувати проекти та рішення; SVsUIShell, який надає функціональність вікон та інтерфейсу користувача (включаючи вкладки, панелі інструментів та вікна інструментів); і SVsShell, який займається реєстрацією VSPackages. Крім того, IDE також відповідає за координацію та забезпечення зв'язку між службами. Усі редактори, дизайнери, типи проектів та інші інструменти реалізовані як VSPackages. Visual Studio використовує COM для доступу до VSPackages. Visual Studio SDK також включає керовану платформу пакетів (MPF), яка являє собою набір керованих обгорток навколо COM-інтерфейсів, які дозволяють записувати пакети будь-якою мовою, сумісною з CLI. Однак MPF не надає всю функціональність, яку відкривають інтерфейси COM Visual Studio. Потім послуги можна використовувати для створення інших пакетів, які додають функціональність до Visual Studio IDE.

Підтримка мов програмування додається за допомогою спеціального пакету VSP, який називається мовною службою. Мовний сервіс визначає різні інтерфейси, які реалізація VSPackage може реалізувати, щоб додати підтримку різних функцій. Функціональні можливості, які можна додати таким чином, включають забарвлення синтаксису, завершення операторів, відповідність дужок, підказки з інформацією про параметри, списки учасників та маркери помилок для фонової компіляції. Якщо інтерфейс реалізований, функціональність буде доступна для мови. Мовні послуги надаються на мовній основі. Реалізації можуть повторно використовувати код із синтаксичного аналізатора або компілятора для мови. Мовні послуги можуть бути реалізовані або в рідному коді, або в керованому коді. Для нативного коду можна використовувати або власні інтерфейси COM, або Babel Framework (частина Visual Studio SDK). Для керованого коду MPF включає обгортки для написання керованих мовних служб [21].

Visual Studio (як і будь-яка інша IDE) містить редактор коду, який підтримує підсвічування синтаксису та завершення коду за допомогою IntelliSense для змінних, функцій, методів, циклів і запитів LINQ. IntelliSense підтримується для включених мов, а також для XML, каскадних таблиць стилів і JavaScript під час розробки веб-сайтів і веб-додатків. Пропозиції автозавершення відображаються в безрежимному списку над вікном редактора коду, поруч із курсором редагування.

Visual Studio містить дебагер, який працює і як дебагер на рівні джерела, і як дебагер на рівні машини. Він працює як з керованим кодом, так і з рідним кодом і може використовуватися для налагодження програм, написаних будь-якою мовою, яку підтримує Visual Studio. Крім того, він також може підключатися до запущених процесів, відстежувати й налагоджувати ці процеси. Якщо вихідний код для запущеного процесу доступний, він відображає код під час його виконання. Якщо вихідний код недоступний, він може показати розбирання. Налагоджувач Visual Studio також може створювати дампи пам’яті, а також завантажувати їх пізніше для налагодження. Також підтримуються багатопотокові програми.

Для тестування API розроблено веб-сервіс було використано програму Postman. Postman – це комп’ютерний додаток, який використовується для тестування API. Postman надсилає запит API на веб-сервер і отримує відповідь, яка б вона не була. Під час надсилання та отримання запитів у Postman не потрібно додаткової роботи чи налаштування фреймворку. Широко використовується тестувальниками та розробниками для кращого тестування програми. Легко інтегрувати з конвеєром безперервної інтеграції (CI) та безперервної розробки.

Postman із багатьма функціями та простим у роботі використовувався мільйонами тестувальників. Використовуючи його простий і зручний інтерфейс, можна легко відправити запит, просто заповнивши необхідні дані, вибравши метод HTTP і натиснувши кнопку «Відправити». Іншою найбільш широко

використовуваною функцією є автоматизація, яка дозволяє налаштовувати тести та писати набори тестів (рисунок 3.3).

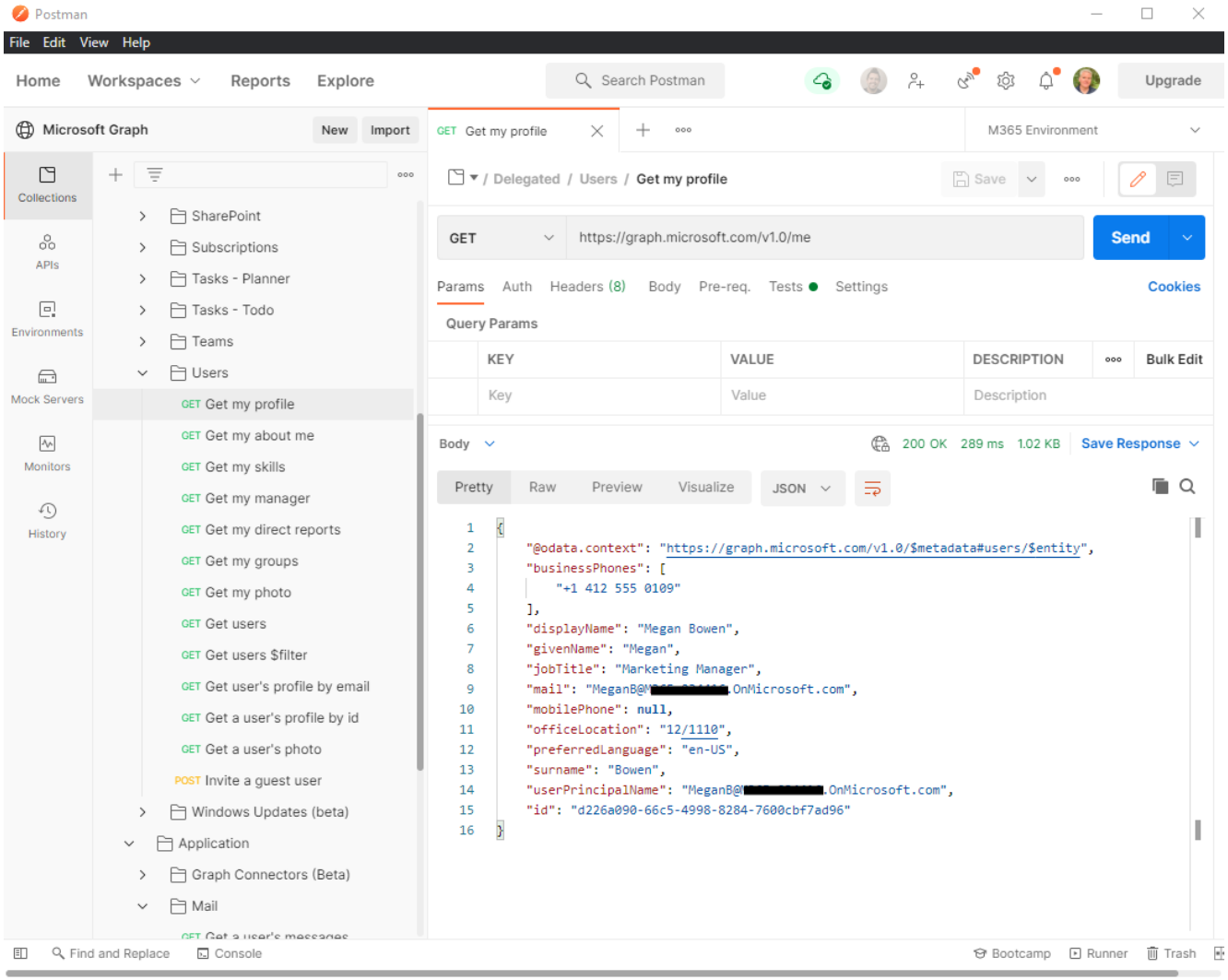

Рисунок 3.3 – Інтерфейс додатку Postman

Незважаючи на те, що він безкоштовний, Postman пропонує вам платну версію з професійними функціями, такими як доступ кількох користувачів, контроль над ролями, доступом, автентифікація SSO тощо. Дозволяючи користувачам налаштовувати необхідне середовище, писати специфікації та, нарешті, контролювати кожен крок, все, що робить Postman ідеальним інструментом тестування.

3.2. Архітектура вебсервісу

Розроблений веб-сервіс можна поділити за функціональністю на кілька модулів, кожен з яких складається з кількох класів. До таких модулів відносяться: модуль обробки зображення за допомогою алгоритму «Super resolutiom from a single image», API модуль – призначений для взаємодії з іншими програмними засобами, а також модуль Utils, в якому реалізовані допоміжні функції (рисунок 3.4).

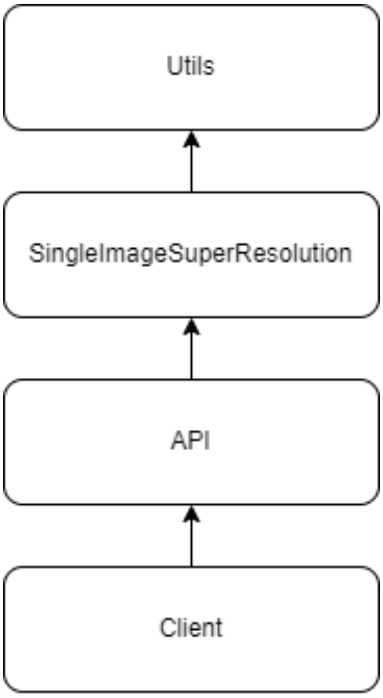

Рисунок 3.4 – Схема взаємодії модулів розробленого веб-сервісу

API веб-сервісу отримує запит від користувача, після чого викликається модуль покращення роздільної здатності, після чого модуль роздільної здатності використовує функціонал модулю Utils.

Модуль Utils реалізовує наступний фунціонал.

Алгоритм «Approximate Nearest Neighbor search». Даний алгоритм призначений для пошуку схожих ділянок зображення. Основну логіку даного алгоритму реалізовано за допомогою додаткової бібліотеки написаної на мові програмування C++.

Для інтеграції даного функціоналу в .Net проект було розроблено додатковий клас-обгортку на мові програмування C#, який отримав назву AnnWrapper (рисунок 3.5).

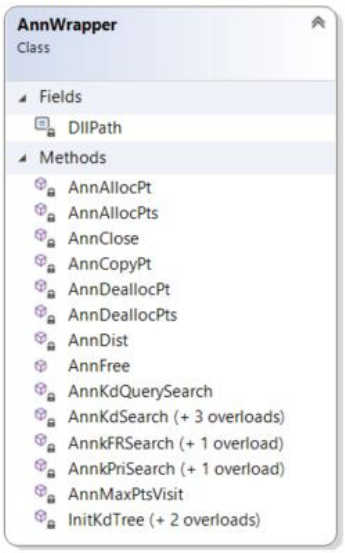

Рисунок 3.5 – Методи класу AnnWrapper

Також було реалізовано класи для певних маніпуляцій з зображенням. До даних маніпуляцій відноситься перетворення з колірної моделі RGB (Red, Green, Blue) в модель YCbCr (рисунок 3.6), а також обернене перетворення, тобто утворення зображення в колірний моделі RGB із зображення YCbCr.

```
2 references | 0 exceptions
public static byte[] ARGBtoY(byte[] argb) // YIQ
{
    int length = argb.Length / ColorComponentsCount;
    byte[] result = new byte[length];
    for (int i = 0; i < result.Length; i++)
    {
        result[i] = (byte)Math.Round(
            argb[i * ColorComponentsCount + RedOffset] * 0.299 +
            argb[i * ColorComponentsCount + GreenOffset] * 0.587 +
            argb[i * ColorComponentsCount + BlurOffset] * 0.114);
    }

    return result;
}
```

Рисунок 3.6 – Метод для перетворення зображення з колірної моделі RGB в модель YCbCr

Дане перетворення використовується на одному з кроків алгоритму «Super resolutiom from a single image», який був описаний вище.

Також в алгоритмі використвується розмивання Гауса. Розмивання Гауса реалізовано за допомогою класу GaussianBlur. Функція розмиття за Гаусом отримується шляхом розмивання (згладжування) зображення за допомогою функції Гаусса для зменшення рівня шуму. Його можна розглядати як нерівномірний фільтр низьких частот, який зберігає низьку просторову частоту та зменшує шум зображення та незначні деталі зображення (рисунок 3.7). Зазвичай це досягається шляхом згортання зображення з ядром Гаусса. Гауссове ядро в двовимірній формі виражається як $G_{2D}(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2-y^2}{2\sigma^2}}$, де σ — стандартне відхилення розподілу, а x та y — індекси розташування. Значення σ контролює дисперсію навколо середнього значення гауссового розподілу, яке визначає ступінь ефекту розмиття навколо пікселя[22].

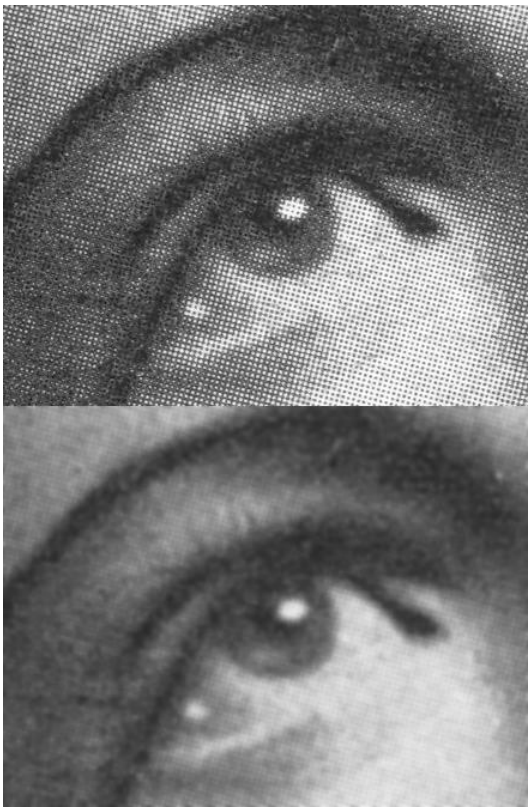

Рисунок 3.7 – Приклад використання розмивання Гауса

3.3. Модуль обробки зображення за допомогою SR-алгоритму

Модуль обробки зображення за допомогою SR-алгоритму є основним модулем розроблено веб-сервісу. Даний модуль реалізує класи та методи з логікою алгоритму «Super resolutiom from a single image». Даний модуль складається з наступних класів: FragmentsMapping, LevelImage, LevelImageFragment, OutputInfo, PointD, SuperResolution (рисунок 3.8)

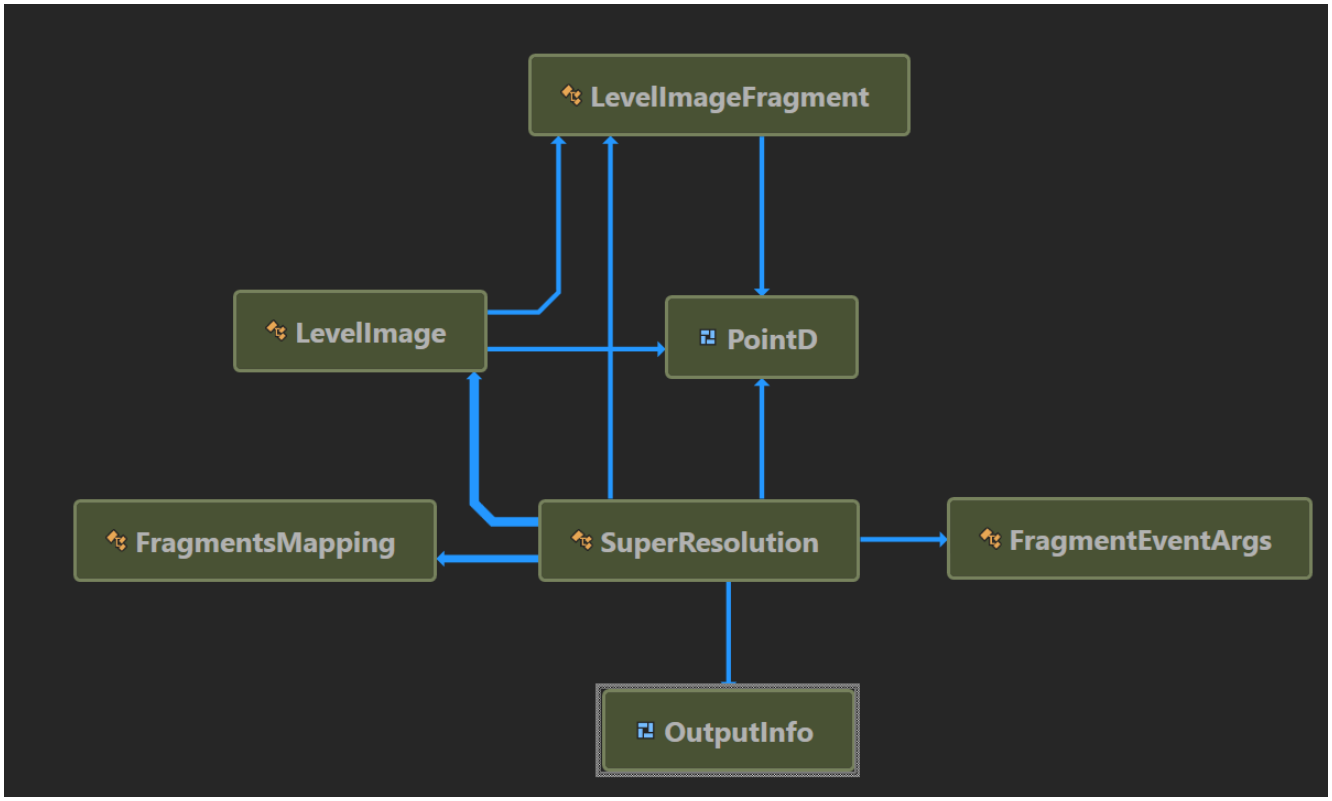

Рисунок 3.8 – Структура модулю обробки зображення за допомогою SR-алгоритму

Головним класом даного модулю є SuperResolution. Саме цей клас реалізовує основну частину алгоритму «Super resolutiom from a single image». Даний клас отримує на вході зображення, яке необхідно обробити, а також додаткові дані, такі як коефіцієнт масштабування, який вказує у скільки разів потрібно збільшити роздільну вихідного зображення, стандартне відхилення для розмиття Гауса тощо.

3.4. API розробленого вебсервісу

API вебсервісу підвищення роздільної здатності зображення було розроблено з використанням фреймворку ASP .NET Core MVC. ASP.NET MVC це багатофункціональна платформою для створення вебдодатків та API-інтерфейсів за допомогою структури проектування Model-View-Controller.

Структура архітектури MVC поділяє додаток на три основні групи компонентів: моделі, представлення та контролери. Це дозволяє реалізувати принципи розподілу завдань. Відповідно до цієї структури запити користувачів надсилаються до контролера, який відповідає за роботу з моделлю для виконання дій користувачів та (або) отримання результатів запитів.

Такий розподіл обов'язків дозволяє масштабувати програму в контексті складності, тому що простіше писати код, виконувати налагодження та тестування компонента (моделі чи контролера) з одним завданням. Набагато важче оновлювати, тестувати та налагоджувати код, залежності якого знаходяться у двох або трьох цих областях.

Крім того, що ASP.NET Core MVC чудово підходить для створення веб-сайтів, ця платформа має потужну підтримку для побудови веб-API. Платформа підтримує узгодження вмісту HTTP із вбудованою підтримкою для форматування даних у вигляді JSON або XML.

Для API вебсервісу збільшення роздільної здатності зображення достатньо лише одного контролеру, який отримав назву SuperResolutionController. В даному контролері реалізовані методи, які приймають на вхід зображення низької роздільної здатності, а також додаткові параметри, такі як коефіцієнт масштабування тощо. Далі виконується обробка зображення, після чого клієнт отримує відповідь із зображенням високої роздільної здатності.

Прикладом такого методу є метод ProcessImage (рисунок 3.9). Даний метод є POST методом, а отже приймає дані в тілі повідомлення. Метод призначений для обробки одного зображення. Тіло повідомлення складається з одного зображення, а також додаткової інформації, яка передається у форматі JSON.

```
[HttpPost]
[Route(template: "ProcessImage")]
0 references
public IActionResult ProcessImage([ModelBinder(BinderType = typeof(JsonModelBinder))] ProcessImage data,
    IFormFile file)
{
    if (file == null || file.Length == 0)
    {
        return BadRequest(error: "Missing file.");
    }

    var memoryStream = new MemoryStream();
    file.CopyTo(memoryStream);

    var superResolution = new SuperResolution(memoryStream);
    var output = superResolution.Process();

    using (var stream = new MemoryStream())
    {
        output.Image.Save(stream, System.Drawing.Imaging.ImageFormat.Png);
        return File(stream.ToArray(), contentType: "image/jpeg");
    }

}
```

Рисунок 3.9 – Метод ProcessImage

Також особливістю API є наявність автентифікації, що дозволяє обмежити коло користувачів, які можуть виконувати запити до вебсервісу. У якості виду аутентифікації обрано Basic authentication. Для Basic authentication HTTP запит містить поле заголовка у формі Authorization: Basic <дані користувача>, де дані користувача — це логін та пароль, об’єднані символом двокрапки та закодовані у вигляді рядка Base64.

ВИСНОВКИ ДО ТРЕТЬОГО РОЗДІЛУ МД

Даний розділ присвячено опису програмної реалізації даного програмного забезпечення. Описано інструментарій за допомогою якого було розроблено вебсервіс, архітектуру створеного програмного забезпечення, а також описано модулі вебсервісу та його API. При розробці вебсервісу було використано мову програмування C#, як середовище розробки використано Visual Studio.

4. ТЕСТУВАННЯ ВЕБСЕРВІСУ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

4.1. Тестування вебсервісу

Тестування розробленого веб-сервісу можна поділити на дві частини: функціональну, тобто тестування самого роботоспроможності самого веб-сервісу, наприклад, перевірка API, а також тестування алгоритму «Super resolutiom from a single image».

Функціональне тестування це досить проста задача, так як її можливо виконати за допомогою спеціального програмного забезпечення такого як Postman. Дане програмне забезпечення призначене саме для тестування API. Postman вже був описаний вище.

Для тестування API зроблено 10 POST-запитів до вебсервісу з різними зображеннями. У всіх випадках отримано успішну відповідь, відповідь включає в себе HTTP-код (для успішних викликів код 200 (OK)), а також зображення з підвищеною роздільною здатністю. Приклад запиту зображено на рисунку 4.1

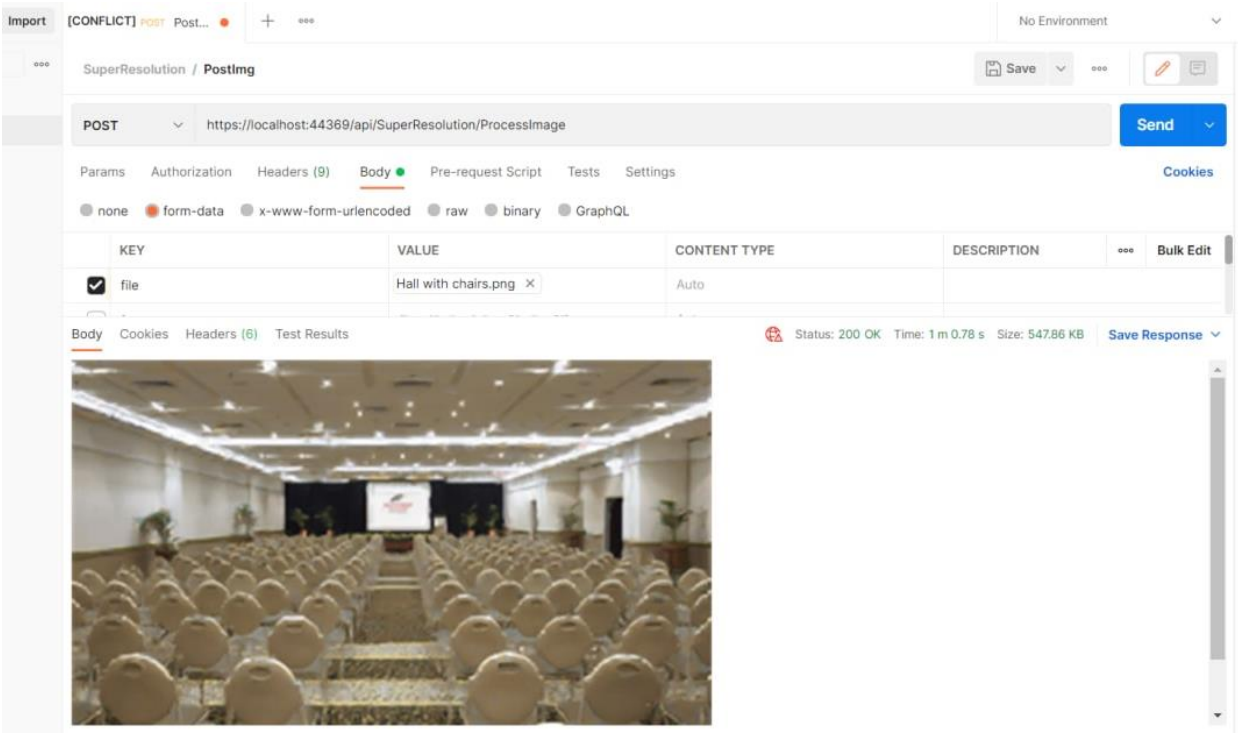

Рисунок 4.1 – Тестування вебсервісу за допомогою додатку «Postman»

Крім того, було виконано кілька запитів з некоректними вхідними даними, наприклад, без надсилання зображення. В таких випадках було отримано результат з описом помилки.

Тестування алгоритму «Super resolutiom from a single image» є більш складною задачею, адже потрібно оцінити чи не було зіпсовано зображення під час збільшення роздільної здатності. Крім того також потрібно порівняти результати даного алгоритму з іншими алгоритмами, призначеними для збільшення роздільної здатності. Візуальне порівняння вхідного зображення низької роздільної здатності та оброблених зображень показано на рисунку 4.2.

Рисунок 4.2 – Візуальне порівняння оригінального зображення та зображень зі збільшеною роздільною здатністю: а) оригінал; б) збільшення роздільної здатності за допомогою бікубічної інтерполяції; в) збільшення роздільної здатності за допомогою алгоритму SRSI; г) справжнє зображення оригіналу з більшою роздільною здатністю

Візуальне порівняння якості цифрових зображень може бути суб’єктивним. З цієї причини необхідно встановити кількісні/емпіричні показники для порівняння впливу алгоритмів покращення зображення на якість зображення.

Одним з таких показників є SSIM (Structural Similarity Index) - Індекс структурної схожості. SSIM це показник, який кількісно визначає погіршення якості зображення, викликане обробкою, наприклад стисненням даних, або втратами при передачі даних. Це повна опорна метрика, яка вимагає двох зображень — еталонного та обробленого зображення. Оброблене зображення зазвичай стискається. Його можна, наприклад, отримати, зберігши еталонне зображення у форматі JPEG (на будь-якому рівні якості), а потім зчитуючи його назад. SSIM найбільш відомий у відеоіндустрії, але має потужне застосування для фотозйомки.

Іншим популярним критерієм для перевірки якості зображення є PSNR (Peak Signal to Noise Ratio) - співвідношення пікового рівня сигналу до шуму. Термін PSNR - вираз для співвідношення між максимально можливим значенням (потужністю) сигналу та потужністю спотворення шуму, що впливає на якість його представлення. Оскільки багато сигналів мають дуже широкий динамічний діапазон (відношення між найбільшим і найменшим можливими значеннями змінної величини), PSNR зазвичай виражається в логарифмічній шкалі децибел.

Використовуючи той самий набір тестових зображень, різні алгоритми покращення зображення можна систематично порівнювати, щоб визначити, чи дає конкретний алгоритм кращі результати. Досліджуваним показником є відношення пік-сигнал/шум. Типові значення PSNR при стисненні зображення та відео з втратами становлять від 30 до 50 дБ за умови, що бітова глибина становить 8 біт, де вище, тим краще. Якість обробки 12-бітових зображень вважається високою, якщо значення PSNR становить 60 дБ або вище [23]. Для 16-бітових даних типові значення PSNR становлять від 60 до 80 дБ [24].

Для тестування було обрано саме варіант PSNR. У тестуванні для порівняння методів збільшення роздільної якості зображень за критерієм PSNR використовувалися: бікубічна інтерполяція та алгоритм SRSI.

Результати порівняння підвищення роздільної здатності зображення за допомогою методів бікубічної інтерполяції та алгоритму SRSI зображено в таблиці 4.1. Для порівняння було використано шість тестових зображень. Для підвищення роздільної здатності кожного з тестових зображень використано метод бікубічної інтерполяції та алгоритм SRSI, крім того тестування проводилося для двох різних коефіцієнтів масштабування.

Таблиця 4.1 - Порівняльна таблиця

Тестове зображення	Коефіцієнт масштабування	PSNR методу бікубічної інтерполяції, dB	PSNR алгоритму SRSI, dB
Зображення 1	2	32,4	35,1
	4	29,1	30,6
Зображення 2	2	34,6	35,1
	4	30,8	32,3
Зображення 3	2	31,9	33,2
	4	29,0	30,1
Зображення 4	2	32,4	34,8
	4	31,5	32,0
Зображення 5	2	35,3	37,2
	4	31,5	33,1
Зображення 6	2	34,2	37,3
	4	30,6	32,2

Відповідно до отриманих результатів алгоритм SRSI показав кращі результати, ніж метод бікубічної інтерполяції, отже при використанні алгоритму SRSI можна отримати результат з достатнім рівнем деталізації.

4.2. Аналіз результатів роботи вебсервісу

В процесі тестування перевірено роботу API, а також роботу алгоритму «Super Resolution from a Single Image». Результати, отримані в ході функціонального тестування, свідчать про коректну роботу API, отже, даний API можна використовувати в інших програмних засобах. Результати тестування алгоритму «Super Resolution from A Single Image» за критерієм PSNR свідчать про те, що даний алгоритм дозволяє якісно збільшити роздільну здатність зображення.

Тобто результати тестування свідчать, що досягнуто мети - створити вебсервіс покращення роздільної якості зображення з використанням модифікації SR-алгоритму.

Крім того, розроблений вебсервіс має потенціал для покращення. Основним напрямком покращення є розробка нових методів API. Наприклад, можливо додати спеціальні методи для використання API у чат-ботах різних месенджерів. Також можливе додавання методів, які будуть приймати кілька список зображень, а результати обробки будуть надсилатись на електронну пошту.

Також можливе покращення роботи самого алгоритму «Super Resolution from A Single Image». Наприклад, можливе покращення швидкодії на кроці пошуку схожих елементів за допомогою використання інших методів пошуку схожих ділянок зображення.

ВИСНОВКИ ДО ЧЕТВЕРТОГО РОЗДІЛУ МД

В даному розділі описано процес тестування розробленого вебсервісу, а також проведено якісну оцінку роботи використаного SR-алгоритму для підвищення роздільної якості зображення. Крім того, було запропоновано можливі варіанти покращення створеного вебсервісу у майбутньому, такі як додавання нових методів та покращення алгоритму.

ВИСНОВКИ

В ході роботи над магістерською дисертацією досліджено принципи роботи Super Resolution–алгоритмів, проведено порівняння різних методів та підходів у SR-алгоритмах, досліджено особливості архітектури вебсервісів, а також оглянуто технології, які використовуються при розробці програмного забезпечення у вигляді вебсервісів. Було обрано модифікацію SR-алгоритму, яка дозволяє підвищити роздільну здатність зображення без додаткових вхідних даних. Дана модифікація отримала назву «Super Resolution from a Single Image». На базі використання даного алгоритму було розроблено вебсервіс з відкритим API для підвищення роздільної здатності зображення.

У першому розділі дисертації проаналізовано актуальність проблеми, а саме необхідність створення вебсервісу з відкритим API для підвищення роздільної здатності зображення. Аналіз показав, що проблема є нині актуальною, адже завдання покращення роздільної здатності зображень є важливим для різних практичних застосувань. Серед таких застосувань можна виділити обробку та аналіз медичних зображень, супутникових фотознімків, обробку даних з пристроїв відеоспостереження і т.д. Проведено порівняння існуючих програмних засобів, які можуть виконувати поставлену задачу, але всі вони не повністю відповідають поставленим вимогам.

В другому розділі дисертації описано основні принципи роботи SR-алгоритмів, а також проведено порівняння різних модифікацій SR-алгоритмів. Було обрано алгоритм для використання у розроблюваному вебсервісі, описано його переваги та недоліки. Крім того, було описано архітектурні особливості програмного забезпечення у вигляді вебсервісу.

Третій розділ присвячено опису програмної реалізації даного програмного забезпечення. Описано інструментарій за допомогою якого було розроблено вебсервіс, архітектуру створеного програмного забезпечення, а також описано модулі вебсервісу та його API.

В четвертому розділі описано процес тестування розробленого вебсервісу, а також проведено якісну оцінку роботи використаного SR-алгоритму для підвищення роздільної якості зображення. Крім того було запропоновано можливі варіанти покращення створеного вебсервісу у майбутньому.

Отже, поставлену задачу було вирішено цілком успішно. Розроблений вебсервіс дозволяє якісно підвищити роздільну здатність зображення без додаткових вхідних даних, має відкритий API, що дозволяє використовувати його в інших програмних засобах, а також має потенціал для покращення.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Графічний редактор Adobe Photoshop. Adobe Photoshop : URL: https://www.adobe.com/ (дата звернення: 06.09.2021).
2. Сервіс для редагування зображень letsenhance.io: URL: https://letsenhance.io/ (дата звернення: 06.09.2021).
3. W. T. Freeman, T. R. Jones, and E. C. Pasztor. Example-based super-resolution. Comp. Graph. Appl., (2), 2002.
4. M. Irani and S. Peleg. Proving resolution by image registration. CVGIP, (3), 1991.
5. Chao Dong, Chen Change Loy, Kaiming He and Xiaoou Tang, Fellow. Super-Resolution Using Deep Convolutional Networks. [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1501.00092.pdf.
6. Jiwon Kim, Jung Kwon Lee and Kyoung Mu Lee. Accurate Image Super-Resolution Using Very Deep Convolutional Networks. [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1511.04587.pdf.
7. Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the Super-Resolution Convolutional Neural Network. [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1608.00367.pdf.
8. Wenzhe Shi, Jose Caballero, Ferenc Huszar, Johannes Totz. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1609.05158.pdf.
9. Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, Kyoung Mu Lee. Enhanced Deep Residual Networks for Single Image Super-Resolution. [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1707.02921.pdf.
10. Namhyuk Ahn, Byungkon Kang and Kyung-Ah Sohn. Fast, Accurate, and Lightweight Super-Resolution with Cascading Residual Network. [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1803.08664.pdf.

11. Yuchen Fan, Honghui Shi, Jiahui Yu, Ding Liu, Wei Han, Haichao Yu, Zhangyang Wang, Xinchao Wang, and Thomas S. Huang. Balanced Two-Stage Residual Networks for Image Super-Resolution. [Електронний ресурс]. – Режим доступу: https://openaccess.thecvf.com/content_cvpr_2017_workshops/w12/papers/Fan_Balanced_Two-Stage_Residual_CVPR_2017_paper.pdf .
12. Jiwon Kim, Jung Kwon Lee and Kyoung Mu Lee. Deeply-Recursive Convolutional Network for Image Super-Resolution. [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1511.04491.pdf.
13. Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Fast and Accurate Image Super-Resolution with Deep Laplacian Pyramid Networks. [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1710.01992.pdf.
14. Yanting Hu, Xinbo Gao, Jie Li, Yuanfei Huang and Hanzi Wang. Single Image Super-Resolution via Cascaded Multi-Scale Cross Network. [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1802.08808.pdf.
15. Chaofeng Wang, Zheng Li, Jun Shi. Lightweight Image Super-Resolution with Adaptive Weighted Learning Network. [Електронний ресурс]. – Режим доступу: https://arxiv.org/ftp/arxiv/papers/1904/1904.02358.pdf.
16. Yulun Zhang , Kunpeng Li , Kai Li , Lichen Wang , Bineng Zhong , and Yun Fu. Image Super-Resolution Using Very Deep Residual Channel Attention Networks. [Електронний ресурс]. – Режим доступу: https://arxiv.org/pdf/1807.02758.pdf.
17. David Salomon. Data Compression: Third Edition. Springer. 2004. 330 с.
18. C. Ledig, L. Theis, F. Huszar, J. Caballero,A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi.. B Photo-realistic single image super-resolution using a generative adversarial network. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), November 2017.
19. K. L. Clarkson. A randomized algorithm for closest-point queries. SIAM Journal on Computing, 1988.

20. J. H. Friedman, J. L. Bentley, and R.A. Finkel. An algorithm for finding best matches in logarithmic expected time. ACM Transactions on Mathematical Software, September 1977.
21. Language Service Essentials. MSDN. Microsoft. Archived from the original on January 12, 2009.
22. Siddharth Misra, Yaokun Wu. Machine Learning for Subsurface Characterization. 2020. 289-314 с.
23. Faragallah Osama S., El-Hoseny Heba, El-Shafai Walid, El-Rahman. A Comprehensive Survey Analysis for Present Solutions of Medical Image Fusion and Future Directions. 2021 [Електронний ресурс]. – Режим доступу: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9311108
24. Raouf Hamzaoui, Dietmar Saupe. Fractal Image Compression. 2006. 168–169 с.