Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

ДОСЛІДЖЕННЯ МЕТОДІВ ОТРИМАННЯ ЗОБРАЖЕНЬ З ВИСОКОЮ РОЗДІЛЬНОЮ ЗДАТНІСТЮ
(тема)

Виконав:
студент 2 курсу, групи ІНФм-22-2
Лещенко Н.К.
(прізвище, ініціали)

Спеціальності 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник проф. Машталір В.П.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Кобилін О.А.
(підпис) (прізвище, ініціали)

2024 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)

Кафедра Інформатики
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Лещенко Наталії Костянтинівни
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження методів отримання зображень з високою роздільною здатністю

затверджена наказом по університету від 3 листопада 2023 року № 1280Ст

2. Термін подання студентом роботи до екзаменаційної комісії 23 грудня 2023 р.

3. Вихідні дані до роботи математичні моделі покращення зображень, теоретичні відомості про методи покращення зображень та методи визначення якості зображення, тестовий набір зображень.

4. Перелік питань, що потрібно опрацювати в роботі
1. Огляд методів отримання зображень високої роздільної здатності.
2. Огляд методів для оцінювання якості зображення.
3. Реалізація програми для отримання зображень високої роздільної здатності.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) актуальність проблеми покращення зображень, постановка задачі, обрані методи для отримання зображень високої роздільної здатності, тестові зображення.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	03.11.2023	
2	Аналіз завдання, підбір літератури	4.11.23-8.11.23	
3	Аналіз літератури з досліджуваної проблеми	9.11.23-11.11.23	
4	Аналіз методів отримання зображень з високою роздільною здатністю	12.11.23-17.11.23	
5	Розробка програми для отримання зображень з високою роздільною здатністю	18.12.23-22.12.23	
6	Програмна реалізація	22.12.23-28.12.23	
7	Оформлення пояснювальної записки	29.12.23-3.12.23	
8	Перевірка на плагіат	08.12.2023	
9	Рецензування	10.12.2023	
10	Підготовка презентації та доповіді	23.12.2023	
11	Занесення роботи в електронний архів	01.01.2024	
12	Попередній захист кваліфікаційної роботи	03.01.2024	

Дата видачі завдання 3 листопада 2023 р.

Студент _____
(підпис)

Керівник роботи _____ проф. Машталір В.П.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 74 с., 3 табл., 44 рис., 1 дод., 40 джерел.

НАДРОЗДІЛЬНІСТЬ, ІНТЕРПОЛЯЦІЯ, ДИСКРЕТИЗАЦІЯ, ГЕНЕРАТИВНА ЗМАГАЛЬНА МЕРЕЖА, ЗГОРТКОВА НЕЙРОННА МЕРЕЖА, ФІЛЬТР ЛАНЦОША.

Об’єктом дослідження є отримання зображень високої роздільної здатності із зображень поганої якості.

Метою дослідження є розробка та вдосконалення методів, що базуються на штучних нейронних мережах, які дозволяють отримувати зображення високої роздільної здатності з зображень низької роздільної здатності.

Проведено дослідження методів покращення зображень, методів визначення якості зображень. Досліджено метод отримання зображень високої роздільної здатності за допомогою генеративно-змагальної мережі. Розроблена модель генеративно-змагальної мережі для покращення зображень.

У результаті дослідження здійснена програмна реалізація системи отримання зображень високої роздільної здатності.

SUPER RESOLUTION, INTERPOLATION, DISCRETISATION, GENERATIVE ADVERSARIAL NETWORK, CONVOLUTIONAL NEURAL NETWORK, LANCZOS FILTER.

The object of research is to obtain high-resolution images from low-resolution images.

The purpose of the study is to develop and improve methods based on artificial neural networks that allow obtaining high-resolution images from low-resolution images.

The researchers investigated image enhancement methods and methods for determining image quality. The method of obtaining high-resolution images using a generative-competitive network is investigated. A model of a generative-competitive network for image enhancement is developed.

As a result of the study, the software implementation of the system for obtaining high-resolution images was carried out.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШНМ – штучна нейронна мережа

МРТ – магнітно-резонансна томографія

SR – Super Resolution

HR – High Resolution

LR – Low Resolution

DL – Deep Learning

CNN – Convolutional Neural Networks (згорткові нейроні мережі)

MSE – Mean Squared Error (середньоквадратична похибка)

IQA – Image Quality Assessment (оцінка якості зображення)

PSNR – Peak Signal-to-Noise Ratio (пікове відношення сигналу до шуму)

SSIM –Structure Similarity (індекс структурної подібності)

MS-SSIM – Multi-Scale extension of SSIM (багатомасштабне розширення SSIM)

MOS – Mean Opinion Score (середній бал оцінки)

API – Application Programming Interface (інтерфейс прикладного програмування)

FSIM – Feature Similarity (індекс подібністі функцій)

GAN – Generative Adversarial Network (генеративна змагальна мережа)

SRGAN – Super resolution Generative Adversarial Network

SRCNN – Super Resolution Convolutional Neural Network

VDSR – Very Deep Super Resolution

VGG – Visual Geometry Group

FSRCNN – Fast Super Resolution Convolutional Neural Network

ESPCNN – Efficient Sub-pixel Convolutional Neural Network

EDSR – Enhanced Deep Residual Networks for Single Image Super Resolution

ESRGAN – Enhanced Super Resolution Generative Adversarial Networks

DCGAN – Deep Convolutional GAN

RRDB – Residual-in-Residual Dense Block (щільний блок «залишок у залишку»)

ReLu – Rectified Linear Unit

RCAB – Residual Channel Attention Block

RG – Residual group

RIR – Residual In Residual

RCAN – Residual Channel Attention Network

IDN – Information Distillation Network (інформаційна дистиляційна мережа)

CMSC – Cascaded Multi-Scale Cross Network

MSC – Multi-Scale Cross

MDSR – Multi-Scale Deep Super Resolution System

LAPSRN – Laplacian Pyramid Super Resolution Network

CARN – Cascading Residual Network

CARN-M – CARN-Mobile

BTSRN – Balanced Two-Stage Residual Networks

DRCN – Deeply-Recursive Convolutional Network

DRNN – Deep Recursive Residual Network

BN – Batch Normalization (нормалізація пакетів)

CSIQ – Colour and Stereo Image Quality

ВСТУП

Зображення з високою роздільною здатністю відіграють ключову роль в багатьох областях сучасної науки і техніки, та щодня допомагають нам у повсякденному житті.

Щоб отримати зображення з високою роздільною здатністю, необхідно покращити зображення з низькою роздільною здатністю. Для цього використовують різноманітні методи покращення зображення, що підбираються залежно від того, які характеристики має вихідне зображення.

Актуальність дослідження полягає у тому, що використання більш ефективних методів отримання зображень високої роздільної здатності буде сприяти підвищенню ефективності у різних сферах нашого життя.

Наприклад, більш точні зображення з камер спостереження на вулицях чи в будівлях допоможуть поліції швидше і точніше знаходити людей, краще розпізнавати номера машин чи відслідковувати речі, які були загублені або вкрадені.

Останні роки людство постійно досліджує космос, кожен день супутники та зонди відправляють на Землю зображення комет, планет, метеоритів, астероїдів, зірок та інших космічних тіл. При покращенні якості цих зображень людство має змогу відкривати для себе все більше нових космічних тіл і явищ.

Зображення гарної якості також дуже важливі у сфері медицини. Наприклад, під час процесу отримання зображень з МРТ існує багато факторів, що впливають на якість зображення і вихідне зображення може бути недостатньо зрозумілим та деталізованим для лікарів, щоб визначити точний діагноз. У цьому випадку зображення високої якості вкрай необхідні для того, щоб рятувати людські життя.

До причин, що знижують якість зображень, можна віднести:

– недостатню контрастність об’єктів на зображенні;

– недостатню або надмірну освітленість об’єктів зйомки;

– відсутність різкості при отриманні зображення;

– наявність шумів на зображенні;

– деталі на зображенні можуть бути розмиті;

– занадто дрібні розміри деталей, які важко розрізнити.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Надроздільність

1.1.1 Загальні відомості про надроздільність

Поняття надроздільної здатності було вперше використано в роботі для покращення роздільної здатності оптичної системи за межами дифракційної межі. В останні два десятиліття поняття надвисокої роздільної здатності (SR) визначається як метод отримання зображень високої роздільної здатності (HR) з відповідного зображення низької роздільної здатності (LR) (рис. 1.1).

а) б)

Рисунок 1.1 – Приклад отримання зображення високої роздільної здатності з зображення низької роздільної здатності:
а) зображення низької роздільної здатності; б) зображення високої роздільної здатності

Надвисока роздільна здатність зображень, незважаючи на те, що дослідження в цій галузі ведуться десятиліттями, залишається складним завданням комп’ютерного зору, оскільки для будь-якого зображення з низькою роздільною здатністю може існувати декілька зображень з високою

роздільною здатністю з невеликими відмінностями у куті зйомки, кольорі, яскравості та інших змінних.

Крім того, існує фундаментальна невизначеність між даними LR та HR, оскільки зменшення вибірки різних зображень HR може призвести до схожого зображення LR, що робить це перетворення процесом «багато до одного».

У минулому класичні методи SR, такі як: статистичні методи, методи на основі прогнозування, методи на основі патчів, методи на основі граней та методи розрідженого представлення використовувалися для досягнення надвисокої роздільної здатності.

Однак, останнім часом розвиток обчислювальних потужностей і великих даних змусив використовувати глибинне навчання (DL) для вирішення проблем SR. За останнє десятиліття дослідження SR на основі глибинного навчання показали вищу продуктивність [1–3].

1.1.2 Визначення надроздільності

SR зображення фокусується на відновленні HR зображення з вхідного LR зображення, LR зображення I_{xLR} можна представити як:

$$I_{xLR} = d(I_{yHR}, \partial), \tag{1.1}$$

де d – функція деградації SR, яка відповідає за перетворення HR зображення в LR зображення;

I_{yHR} – це вхідне HR зображення;

∂ – вхідні параметри функції деградації зображення.

Параметрами деградації зазвичай є коефіцієнт масштабування, тип розмиття та шум. На практиці процес деградації і залежні від нього параметри невідомі, і для отримання HR зображень методом SR використовуються лише

LR зображення. Процес SR відповідає за прогнозування оберненої функції деградації d, тобто $g = d^{-1}$.

$$g(I_{xLR}, \delta) = d^{-1}(I_{xHR}) = I_{yE} \approx I_{yHR}, \quad (1.2)$$

де g – функція SR;

δ – вхідні параметри функції g;

I_{yE} – визначена HR, що відповідає вхідному I_{xLR} зображенню.

Варто також зауважити, що функція надроздільної здатності, як показано у формулі вище є погано поставленою, оскільки функція g є неін'єктивною функцією. Таким чином, існує нескінченна кількість значень I_{yE}, для яких виконується умова:

$$d(I_{yE}, \partial) = I_{xLR}. \quad (1.3)$$

Процес деградації вхідних LR зображень невідомий, і на нього впливають численні фактори такі як: шум, спричинений сенсором, артефакти, створені через стиснення з втратами, розмиття руху та розфокусовані зображення [4, 5].

У літературі більшість досліджень використовують одну функцію пониження дискретизації зображення як функцію деградації зображення:

$$d(I_{yHR}, \partial) = (I_{yHR}) \downarrow_{s_f}, \{s\} \subseteq \partial, \quad (1.4)$$

де $\downarrow_{s_f}$ – оператор зменшення вибірки;

s_f – коефіцієнт масштабування.

Однією з часто використовуваних у SR функцій зниження дискретизації зображень для глибокого навчання надроздільності є бікубічна інтерполяція зі згладжуванням.

У деяких роботах, наприклад, дослідники використовували більше операцій у функції зниження дискретизації зображень (рис. 1.2), і загальна операція даунсемплінгу становить:

$$d(I_{yHR}, \partial) = (I_{yHR} \otimes \kappa) \downarrow_{s_f} + n_\sigma, \{\kappa, s, \sigma\} \subseteq \partial, \quad (1.5)$$

де $I_{yHR} \otimes \kappa$ – згортка HR зображення I_{yHR} з ядром розмиття κ;

n_σ – адитивний білий Гаусівський шум зі стандартним відхиленням σ.

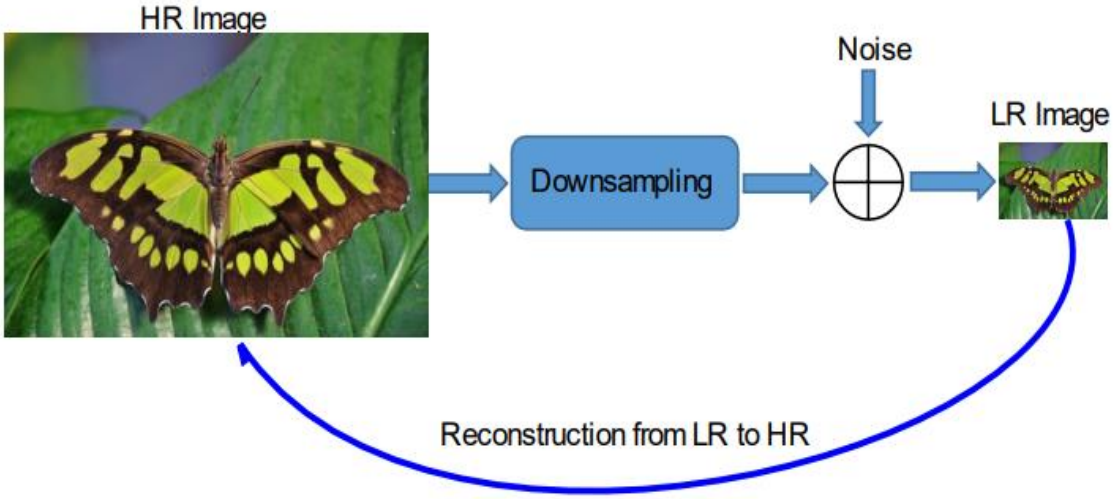


Рисунок 1.2 – Зменшення та збільшення дискретизації у надвисокій роздільній здатності, додавання шуму для імітації реалістичної деградації зображення

Функція деградації, визначена вище і показана на рисунку 1.2 є ближчою до реальної функції, оскільки враховує більше параметрів, ніж проста функція деградації з пониженням дискретизації.

Нарешті, метою SR є мінімізація функції втрат наступним чином:

$$\hat{\phi} = \left[min\, \mathcal{L}\,(I_{yE}, I_{yHR})\right]_\phi + \; h\Psi(\phi), \quad (1.6)$$

де h – параметр компромісу;

$\Psi(\phi)$ – член регуляризації;

$\mathcal{L}(I_{yE}, I_{yHR})$ – функція втрат між вихідним зображенням HR надроздільності та фактичним зображенням HR.

Найпоширенішою функцією втрат, що використовується в SR, є середньоквадратична похибка на основі пікселів (MSE), яку також можна назвати втратою пікселів. В останні роки використовуються комбінації різних функцій втрат [6].

1.2 Методи перевірки якості SR зображень

Якість зображення може мати кілька визначень залежно від методу вимірювання, і, як правило, це міра якості візуальних атрибутів і сприйняття глядачами. Методи оцінки якості зображення характеризуються на суб'єктивні методи (людське сприйняття зображення є природним і якісним) та об'єктивні методи (кількісні методи, за допомогою яких якість зображення можна обчислити чисельно). Візуальні аспекти зображення, пов'язані з якістю, є гарним показником, але цей метод вимагає більше ресурсів, особливо якщо набір даних великий. Таким чином, для задач комп'ютерного зору більш придатними є об'єктивні методи [7, 8]. Методи IQA в основному поділяються на три категорії, а саме: еталонне зображення, засноване на ознаках фактичного зображення, і сліпе IQA без інформації про базову істину [9].

1.2.1 Метод пікового співвідношення сигналу до шуму

Як і в більшості інформаційних систем, пікове відношення сигнал-шум (PSNR) є методом вимірювання для аналізу потужності сигналу в порівнянні з потужністю шуму, особливо в зображеннях, PSNR використовується як кількісна міра якості стиснення зображення. У сфері надвисокої роздільної

здатності PSNR зображення визначається максимальним значенням пікселя і середньоквадратичною похибкою між еталонним зображенням і SR зображенням, також відомою як потужність шуму, що погіршує зображення.

Для заданого максимального значення пікселя M і еталонного зображення I_r , що має t пікселів, та SR зображення I_y, пікове відношення сигнал-шум визначається як:

$$PSNR = 10\,log_{10}\left(\frac{M^2}{MSE}\right), \tag{1.7}$$

де M – глибина 8-бітового колірного простору, тобто має максимальне значення 255.

MSE визначається за допомогою наступної формули:

$$MSE = \frac{1}{t}\sum_{i=1}^{t}(I_r(i) - I_y(i))^2. \tag{1.8}$$

Як бачимо, PSNR пов’язане з окремими значеннями інтенсивності пікселів зображення SR та еталонного зображення і є піксельною метрикою якості зображення. У деяких випадках ця метрика якості може вводити в оману, оскільки загальне зображення може не бути візуально подібним до еталонного. Цю метрику все ще використовують для порівняння зображень, особливо для порівняння результатів алгоритмів SR з раніше опублікованими результатами, щоб порівняти роботу будь-якого нового методу в галузі SR.

MSE для кольорових зображень усереднюється для кольорових каналів, а альтернативним підходом є вимірювання PSNR для каналів яскравості або каналів відтінків сірого, оскільки людське око більш чутливе до змін яскравості на відміну від зміни кольоровості .

1.2.2 Метод показника структурної схожості

Візуальне сприйняття людини ефективно виокремлює структурну інформацію в зображенні, проте PSNR не враховує структурний склад зображення.

Метрика індексу структурної подібності (SSIM) була запропонована для вимірювання структурної подібності між зображеннями шляхом порівняння контрасту, яскравості та структурних деталей на еталонному зображенні [10].

Зображення I_r із загальною кількістю пікселів P, контрастом C_I та яскравістю L_I можна позначити як стандартне відхилення та середнє значення інтенсивності зображення:

$$L_I = \frac{1}{M}\sum_{i=1}^{M} I_r(i), \tag{1.9}$$

$$C_I = \sqrt{\left(\frac{1}{M-1}\sum_{i=1}^{M}(I_r(i) - L_I)^2\right)}. \tag{1.10}$$

де $I_r(i)$ – піксель еталонного зображення.

Розглянемо порівняння на основі контрасту та яскравості між еталонним і оцінюваним зображеннями:

$$Com_l\left(I_r, \hat{I}\right) = \frac{2L_{I_r}L_{\hat{I}} + \mu_1}{{L_{I_r}}^2 + L_{\hat{I}}^2 + \mu_1}, \tag{1.11}$$

$$Com_c\left(I_r, \hat{I}\right) = \frac{2C_{I_r}C_{\hat{I}} + \mu_2}{{C_{I_r}}^2 + C_{\hat{I}}^2 + \mu_2}. \tag{1.12}$$

де μ_1 – постійний член, що має значення $\mu_1 = (k_1 S)^2$ і гарантує, що $k_1 \ll 1$;

μ_2 – постійний член, що має значення $\mu_2 = (k_2 S)^2$ і гарантує, що $k_2 \ll 1$.

Нормалізовані значення пікселів $I_r - L_{I_r}/C_{I_r}$ представляють структуру зображення, в той час як внутрішній добуток цих значень є еквівалентом

структурної подібності між еталонним зображенням I_r та оціненим зображенням $\hat{I}$.

Коваріація задається через:

$$\sigma_{I_r,\hat{I}} = \frac{1}{M-1}\sum_{i=1}^{M}\left(I_r(i) - L_{I_r}\right)\left(\hat{I}(i) - L_{\hat{I}}\right). \quad (1.13)$$

Функція структурного порівняння $Com_s\left(I_r, \hat{I}\right)$ задається формулою:

$$Com_s\left(I_r, \hat{I}\right) = \frac{\sigma_{I_r,\hat{I}} + \mu_3}{C_{I_r}C_{\hat{I}} + \mu_3}, \quad (1.14)$$

де μ_3 – константа стабільності.

Остаточний індекс структурної подібності (SSIM) задається за допомогою:

$$SSIM\left(I_r, \hat{I}\right) = \left\{Com_l\left(I_r, \hat{I}\right)\right\}^{\alpha}\left\{Com_c\left(I_r, \hat{I}\right)\right\}^{\beta}\left\{Com_s\left(I_r, \hat{I}\right)\right\}^{\gamma}, \quad (1.15)$$

де α, β, γ – керуючі параметри, які можна регулювати, щоб збільшити важливість яскравості, контрасту та структурного порівняння при обчисленні SSIM.

Традиційно PSNR використовується в задачах комп’ютерного зору для оцінки, але SSIM базується на людському сприйнятті структурної інформації в зображенні, тому цей метод широко використовується для порівняння структурної подібності між зображеннями.

1.2.3 Метод оцінки думок

Оцінка думок – це якісний метод, який належить до суб’єктивної категорії IQA. У цьому методі тестувальників якості просять оцінити якість

зображень на основі певних критеріїв, наприклад, різкості, природного вигляду та кольору, де остаточна оцінка є середнім значенням оцінок.

Цей метод має обмеження, такі як нелінійність між оцінками, варіація результатів через зміну критеріїв тестування та людські помилки.

У SR певні методи показали хороші об’єктивні показники якості, але погані суб’єктивні результати, особливо у випадку реконструкції людського обличчя. Таким чином, метод оцінювання думок також використовується в дослідженнях для вимірювання якості людського сприйняття.

Оцінювання думок використовує людей-рейтингерів для ручного оцінювання зображень, хоча цей метод може забезпечити точні результати, якщо йдеться про людське сприйняття. Проте цей метод вимагає багато ресурсів, особливо для великих наборів даних.

Розробка методів IQA є відкритим питанням, і в останні роки різні дослідники пропонували метрики SR, але ці методи не були широко використані спільнотою SR. Метрика індексу подібності ознак (FSIM) оцінює якість зображення шляхом виділення характерних точок, які розглядаються зоровою системою людини на основі величини градієнта та фазової конгруентності.

Багатомасштабна структурна подібність (MS-SSIM) використовує багатомасштабність для врахування варіацій в умовах перегляду для вимірювання якості зображення, MS-SSIM забезпечує більшу гнучкість у вимірюванні якості зображення, ніж одномасштабна SSIM. SSIM та MS-SSIM погано працюють на деформованих та зашумлених зображеннях, тому вони використовували чотири-компонентний зважений метод, який коригував вагу оцінок на основі локальних особливостей, тоді як у випадку деформованих за контрастом зображень, таких як TID2013 та CSIQ, SSIM не працює належним чином.

Тож якість сприйняття та деформація зображення суперечать одне одному, зі зменшенням деформації, якість сприйняття також має погіршуватися.

Таким чином, точне вимірювання якості зображень SR все ще залишається відкритою областю досліджень.

1.3 Постановка задачі дослідження

Зображення надвисокої роздільної здатності з’явилися як важливий напрямок в обробці зображень, що має на меті підвищити якість і деталізацію цифрових зображень, виходячи за межі обмежень, які накладає апаратне забезпечення для обробки зображень.

Хоча технології надвисокої роздільної здатності продемонстрували значний прогрес, вони не позбавлені певних викликів і проблем, починаючи від алгоритмічних складнощів і закінчуючи застосуванням у реальному світі.

Однією з алгоритмічних проблем є інтенсивність обчислень. Серед головних проблем у сфері надвисокої роздільної здатності є задача щодо обчислювальної інтенсивності відповідних алгоритмів. Реконструкція зображень з високою роздільною здатністю вимагає складних математичних операцій і великої обчислювальної потужності, що може бути складним і слабким місцем, особливо для застосунків, що працюють в режимі реального часу. Досягнення балансу між обчислювальною ефективністю та якістю зображень залишається постійним викликом для дослідників.

Багато алгоритмів надвисокої роздільної здатності покладаються на великі навчальні набори даних, щоб добре узагальнювати різноманітні зображення. Однак отримання високоякісних навчальних даних може бути складним завданням, особливо для спеціалізованих областей або унікальних сценаріїв зйомки. Відсутність репрезентативних наборів даних може призвести до низької продуктивності та обмеженого застосування методів надвисокої роздільної здатності.

Алгоритми надвисокої роздільної здатності часто стикаються з компромісом між точністю та швидкістю. Застосунки в режимі реального

часу, такі як потокове відео або медична візуалізація, вимагають швидкої обробки, що ускладнює реалізацію алгоритмів, які підтримують високу точність при дотриманні часових обмежень. Досягнення балансу між цими суперечливими вимогами є постійним викликом, так як ми завжди намагаємось отримувати результат як найшвидше, точність і якість зображення через це погіршується. І навпаки, коли необхідно мати зображення високої точності, то ми втрачаємо багато часу на обробку даних [11].

Також надроздільність зустрічається з проблемами застосування у реальному свті. Обмежена інформація на зображеннях з низькою роздільною здатністю не дає змоги отримати зображення високої роздільної здатності, яке було б точним.

Надвисока роздільна здатність має на меті відновлення дрібних деталей із зображень низької роздільної здатності. Однак на зображеннях низької роздільної здатності часто бракує важливої інформації, що ускладнює точне прогнозування високочастотних деталей. Шум, артефакти стиснення та інші спотворення ще більше ускладнюють завдання, створюючи значні проблеми для алгоритмів надвисокої роздільної здатності.

Моделі надвисокої роздільної здатності, навчені на конкретних типах сцен, можуть не впоратися з узагальненням різноманітного контенту зображень. Варіації умов освітлення, текстур і типів об’єктів можуть призвести до неоптимальної продуктивності або навіть відмови в певних сценаріях. Досягнення надійності та адаптивності до різних сцен є постійним викликом [12].

Перехід алгоритмів надвисокої роздільної здатності від дослідницьких середовищ до реальних застосувань представляє свій власний набір викликів. Інтеграція в існуючі системи, сумісність з різноманітним обладнанням та відповідність вимогам конкретних галузей вимагають ретельного розгляду. Подолання розриву між теоретичними досягненнями та практичною реалізацією залишається ключовою перешкодою. Налаштування для отримання зображень високої роздільної здатності зображень живих істот

дуже відрізняється від зображень неживих об’єктів, наприклад будівель чи пейзажів.

Хоча візуалізація з надвисокою роздільною здатністю має величезні перспективи для покращення якості зображень, її шлях позначений кількома викликами. Подолання цих викликів вимагає міждисциплінарної співпраці, що включає в себе знання з комп’ютерного зору, машинного навчання і специфічних знань у конкретних галузях. Оскільки дослідники продовжують вирішувати ці питання, сфера надвисокої роздільної здатності готова досягти значних успіхів, відкриваючи нові можливості для високоякісних зображень у різних галузях.

Об’єктом дослідження є отримання зображень високої роздільної здатності із зображень поганої якості [13].

Метою дослідження є розробка та вдосконалення методів, що базуються на штучних нейронних мережах, які дозволяють отримувати зображення високої роздільної здатності з зображень низької роздільної здатності.

Для досягнення мети необхідно вирішити такі завдання:

– оглянути методи отримання зображень високої роздільної здатності;

– оглянути методи оцінки якості зображень;

– реалізувати програмне забезпечення для отримання зображень високої роздільної здатності.

2 ДОСЛІДЖЕННЯ МЕТОДІВ НАДРОЗДІЛЬНОСТІ

2.1 Огляд сучасних методів надроздільності

2.1.1 Метод надроздільності перед дискретизацією

Вивчення функцій відображення для перед-дискретизації з LR зображення безпосередньо до HR зображення виконується за допомогою фреймворку, де спочатку робиться вибірка з LR зображення, а різні шари згортки використовуються для вилучення зображень ітеративним способом за допомогою глибоких нейронних мереж. Використовуючи цю концепцію, був представлений фреймворк SR на основі попередньої вибірки (SRCNN) (рис. 2.1).

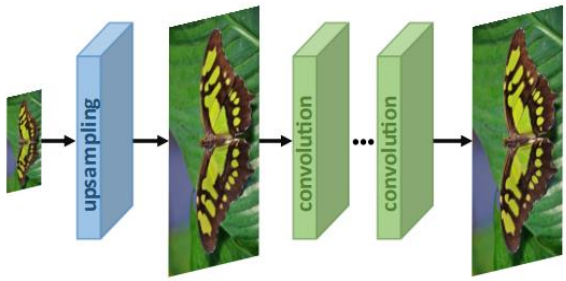


Рисунок 2.1 – Мережевий конвеєр надвисокої роздільної здатності на основі попередньої дискретизації

SRCNN було використано для вивчення наскрізного відображення перетворення з LR до HR зображень за допомогою CNN. Використовуючи класичні методи апсемплінгу, LR зображення спочатку перетворюється на HR зображення, а потім глибокі CNN використовуються для вивчення представлення для відображення HR зображення [14].

Оскільки шар попередньої дискретизації вже виконує фактичне перетворення пікселів, мережі потрібно уточнити результати за допомогою CNN, це призводить до зменшення складності навчання. Порівняно з

одномасштабними CNN, які використовують певні масштаби вхідних даних, ці моделі здатні обробляти зображення довільного розміру для уточнення і мають подібну продуктивність.

В останні роки багато прикладних досліджень використовували цей фреймворк, відмінності в цих моделях полягають у рівнях глибокого навчання, що застосовуються після вибірки.

Єдиним недоліком цієї моделі є використання заздалегідь визначеного класичного методу попередньої вибірки, що часто призводить до розмиття зображення, посилення шуму в апсемпліфікованому зображенні, що в подальшому впливає на якість кінцевого HR зображення. Крім того, розміри зображення збільшуються на початку застосування методу.

Таким чином обчислювальні витрати та вимоги до оперативної пам’яті цього фреймворку вищі, ніж в інших фреймворків (рис. 2.2).

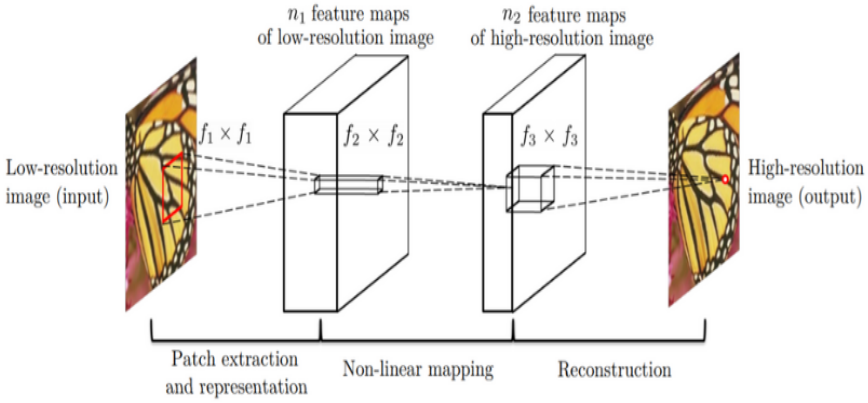


Рисунок 2.2 – Відображення роботи SRCNN

SRCNN – це проста архітектура CNN, що складається з трьох шарів: для вилучення патчів, нелінійного відображення та реконструкції. Шар виділення патчів використовується для виділення щільних патчів з вхідних даних і їх представлення за допомогою згорткових фільтрів [15]. Шар нелінійного відображення складається зі згорткових фільтрів 1×1, які використовуються для зміни кількості каналів і додавання нелінійності. Фінальний шар реконструкції відновлює зображення з високою роздільною здатністю.

Для навчання мережі використовується функція втрат MSE, а для оцінки результатів – PSNR.

Дуже глибока надроздільна здатність (VDSR) є вдосконаленням SRCNN, з додаванням деяких функцій. Як видно з назви, замість меншої мережі з великими згортковими фільтрами використовується глибока мережа з маленькими згортковими фільтрами 3×3. Це було засновано на архітектурі VGG.

Мережа намагається вивчити залишок вихідного зображення та інтерпольованого вхідного, а не вивчити пряме відображення, як це робить SRCNN, що показано на малюнку вище. Це спрощує завдання. Початкове зображення низької роздільної здатності додається до виходу мережі, щоб отримати остаточний результат HR. Градієнтне відсікання використовується для навчання глибокої мережі з вищою швидкістю навчання (рис. 2.3).

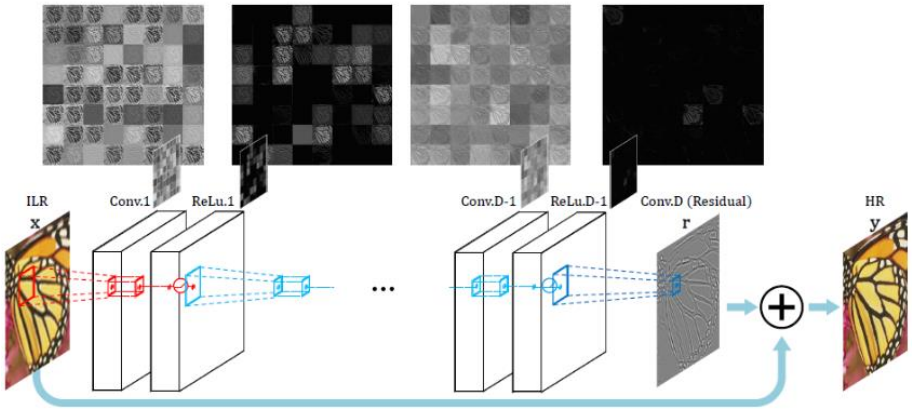


Рисунок 2.3 – Відображення роботи VDSR

2.1.2 Метод надроздільності після дискретизації

Щоб мінімізувати вимоги до пам’яті та підвищити обчислювальну ефективність, в SR було використано метод пост-дискретизації, він був використаний в SR для використання глибокого навчання для вивчення функцій відображення в низьковимірному просторі.

Завдяки низьким обчислювальним витратам і використанню низьковимірного простору для глибокого навчання ця модель отримала широке застосування в SR, оскільки це зменшує складність (рис. 2.4).

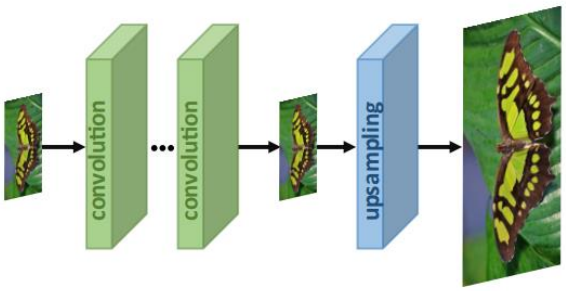


Рисунок 2.4 – Мережевий конвеєр надвисокої роздільної здатності на основі пост-дискретизації

Оскільки процес виділення ознак у SR з попередньою перед-дискретизацією відбувається у просторі з високою роздільною здатністю, необхідна обчислювальна потужність також є вищою [16, 17]. У SR після перед-дискретизації ця проблема вирішується шляхом вилучення ознак у просторі з нижчою роздільною здатністю, а перед-дискретизація виконується лише наприкінці, що значно зменшує обчислення. Крім того, замість простої бікубічної інтерполяції для апсемплінгу використовується навчена апсемплінг функція у вигляді субпіксельної згортки, що робить мережу здатною до наскрізного навчання.

Як видно з наведеного нижче рисунка 2.5, основні відмінності між SRCNN та FSRCNN полягають у деяких аспектах. На початку відсутня попередня обробка або підвищена вибірка. Виділення ознак відбувається в просторі з низькою роздільною здатністю [18]. Згортка 1×1 використовується після початкової згортки 5×5 для зменшення кількості каналів, а отже, менших обчислень і пам’яті, подібно до того, як розробляється мережа Inception.

Замість великого згорткового фільтра використовується кілька згорток 3×3, подібно до того, як працює мережа VGG, за рахунок спрощення архітектури для зменшення кількості параметрів.

Повторна вибірка здійснюється за допомогою навченого деконволюційного фільтра, що покращує модель (рис. 2.5).

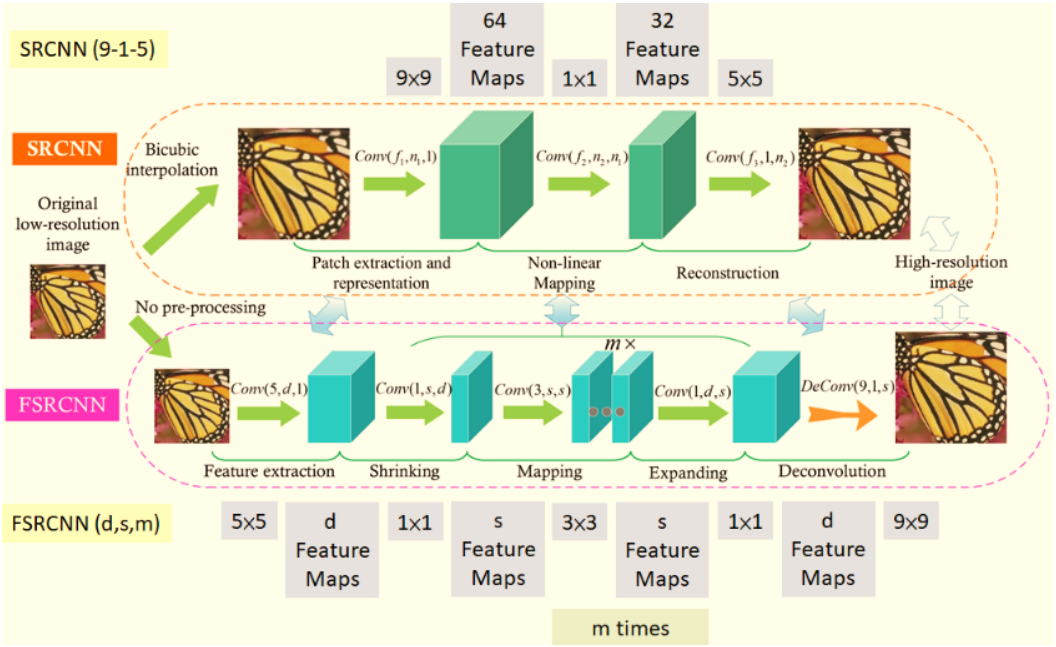


Рисунок 2.5 – Відображення роботи FSRCNN в порівнянні з SRCNN

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

ESPCN вводить поняття субпіксельної згортки для заміни деконволюційного шару при підвищенні вибірки. Це вирішує дві пов'язані з цим проблеми [19, 20]. Перша проблема стосується того, що деконволюція відбувається у просторі з високою роздільною здатністю, а отже є більш затратною в обчислювальному плані.

Також це вирішує проблему шахової дошки при деконволюції, яка виникає через перекриття операцій згортки.

Субпіксельна згортка працює шляхом перетворення глибини в простір. Пікселі з кількох каналів зображення з низькою роздільною здатністю переставляються в один канал у зображенні з високою роздільною здатністю.

Для прикладу, вхідне зображення розміром 5×5×4 може перегрупувати пікселі в останніх чотирьох каналах в один канал, в результаті чого буде отримано зображення з роздільною здатністю 10×10 (рис. 2.6).

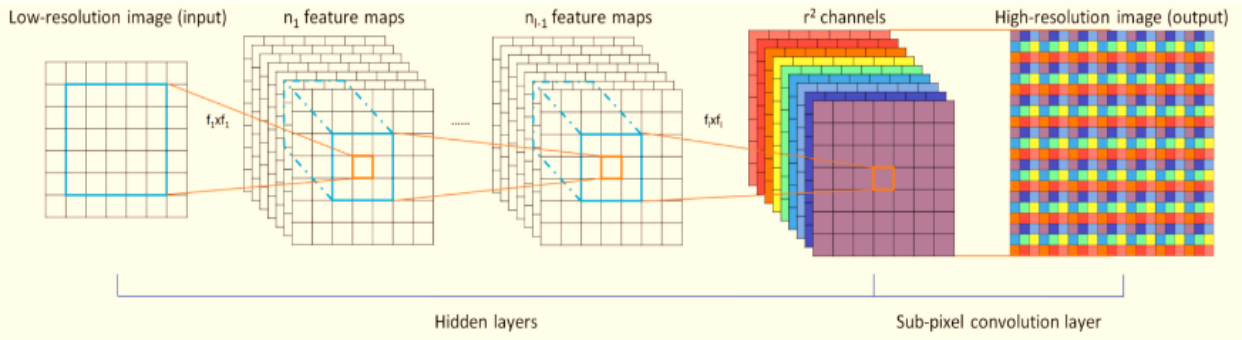


Рисунок 2.6 – Відображення роботи ефективної субпіксельної згорткової нейронної мережі (ESPCN)

2.1.3 Метод залишкового навчання

Залишкове навчання широко використовувалося в галузі SR, поки не було запропоновано ResNet для навчання за залишковими даними, як показано на рисунку 2.7.

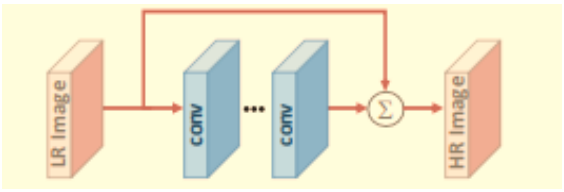


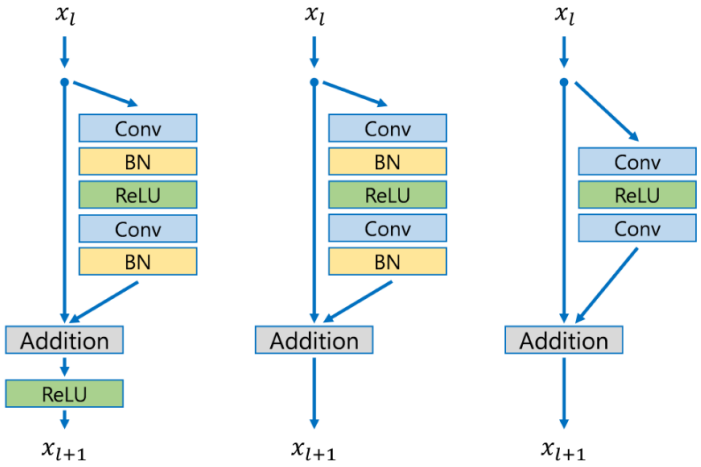
Рисунок 2.7 – Відображення роботи залишкового навчання

Існує два підходи, локальне та глобальне навчання за залишковими даними. Локальний підхід до навчання за залишковими даними пом’якшує проблему деградації, яка викликана збільшенням глибиною мережі. Крім того, локальне залишкове навчання також покращує швидкість навчання і зменшує складність навчання. Це часто використовується в галузі SR.

Глобальне залишкове навчання – це підхід, в якому вхідні дані та кінцевий результат корелюються, і в зображенні SR вихідний HR сильно корелює з вхідним LR зображенням [21]. Таким чином, навчання за глобальними залишками між LR та HR зображеннями є важливим у SR. При навчанні за глобальними залишками модель вивчає лише карту залишків, яка

перетворює LR зображення в HR зображення, генеруючи відсутні високочастотні деталі на LR-зображенні.

Крім того, залишки мінімальні, що значно зменшує складність навчання і складність моделі у глобальному навчанні за залишками. Загалом, обидва методи використовують залишки для зв'язку вхідного зображення з вихідним HR зображенням. У випадку глобального залишкового навчання зв'язок здійснюється безпосередньо, а у випадку локального залишкового навчання використовуються різні шари різної глибини для з'єднання вхідного зображення (з використанням локальних залишків) з вихідним (рис. 2.8).



а) б) в)

Рисунок 2.8 – Залишкові блоки різних моделей:
а) залишковий блок ResNet; б) залишковий блок SRResNet;
в) залишковий блок EDSR

Архітектура EDSR базується на архітектурі SRResNet, що складається з декількох залишкових блоків. Основна відмінність EDSR від SRResNet полягає в тому, що шари пакетної нормалізації видалено. BN нормалізує вхідні дані, обмежуючи таким чином діапазон мережі. Видалення BN призводить до покращення точності [22, 23]. Шари BN також споживають пам'ять, і їх видалення призводить до зменшення пам'яті на сорок відсотків, що робить навчання мережі більш ефективним.

MDSR є розширенням EDSR з декількома вхідними та вихідними модулями, які дають відповідну роздільну здатність при коефіцієнтах дискретизації 2×, 3× та 4× (рис. 2.9).

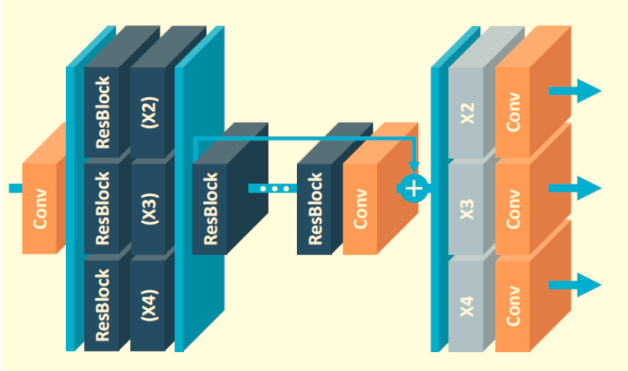


Рисунок 2.9 – Архітектура мультимасштабної мережі SR мережі (MDSR)

На початку присутні модулі попередньої обробки для специфічних для масштабу вхідних даних, що складаються з двох залишкових блоків з ядрами 5×5. Велике ядро використовується для шарів попередньої обробки, щоб мережа залишалася неглибокою, але при цьому досягалося високе сприйнятливе поле [24].

Наприкінці модулів попередньої обробки, специфічних до масштабу, знаходяться спільні залишкові блоки, які є спільним блоком для даних усіх роздільних здатностей. Нарешті, після спільних залишкових блоків розташовані модулі перед-дискретизації, специфічні для кожного масштабу. Хоча загальна глибина MDSR у 5 разів більша порівняно з одномасштабним EDSR, кількість параметрів лише у 2,5 рази, а не у 5 разів, завдяки спільним параметрам. MDSR досягає порівнянних результатів з EDSR, навіть якщо мережа має менше параметрів, ніж моделі EDSR разом.

Каскадний механізм як на локальному, так і на глобальному рівнях, щоб об’єднати функції з декількох шарів і дати мережі можливість отримувати більше інформації [25]. На додаток до CARN, пропонується менша за розміром CARN-M, яка має легшу архітектуру, без значного погіршення результатів, за допомогою рекурсивної мережевої архітектури (рис. 2.10).

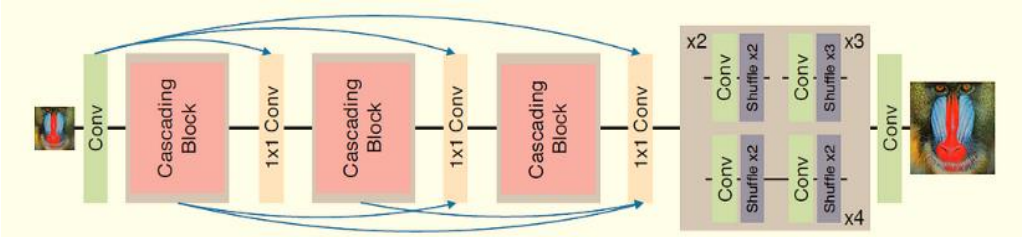


Рисунок 2.10 – Архітектура мережі CARN

Глобальні зв’язки в CARN візуалізовані вище. Кульмінація кожного каскадного блоку зі згорткою 1×1 отримує входи від усіх попередніх каскадних блоків і початковий вхід, що призводить до ефективної передачі інформації. Кожен залишковий блок у каскадному блоці закінчується згорткою 1×1, яка має зв’язки з усіх попередніх залишкових блоків разом з основним входом, подібно до того, як працює глобальне каскадування.

Залишковий блок у ResNet замінено новим блоком Residual-E, який натхненний глибинними згортками у MobileNet [26]. Замість глибинних згорток використовуються групові згортки, і результати показують зменшення кількості обчислень в 1,8 – 14 разів, залежно від розміру групи. Для подальшого зменшення кількості параметрів використовується спільний залишковий(рекурсивний) блок (рис.2.11).

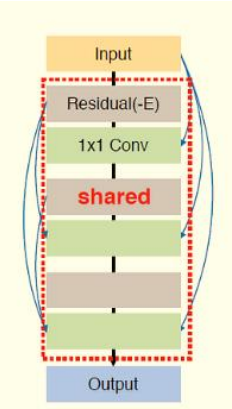


Рисунок 2.11 – Спільний рекурсивний блок мережі CARN

Це призводить до зменшення кількості параметрів до трьох разів від початкової кількості.

2.1.4 Багатоступеневі залишкові мережі

Для того, щоб вирішити задачу виділення ознак окремо в просторі з низькою та високою роздільною здатністю, в деяких архітектурах розглядається багатоступеневий дизайн для покращення їхньої продуктивності. На першому етапі відбувається прогнозування грубих ознак, тоді як на наступних етапах відбувається їх покращення [27].

BTSRN складається з двох етапів: етапу низької роздільної здатності (LR) та етапу високої роздільної здатності (HR). Етап LR складається з 6 залишкових блоків, тоді як етап HR містить 4 залишкових блоки (рис. 2.12).

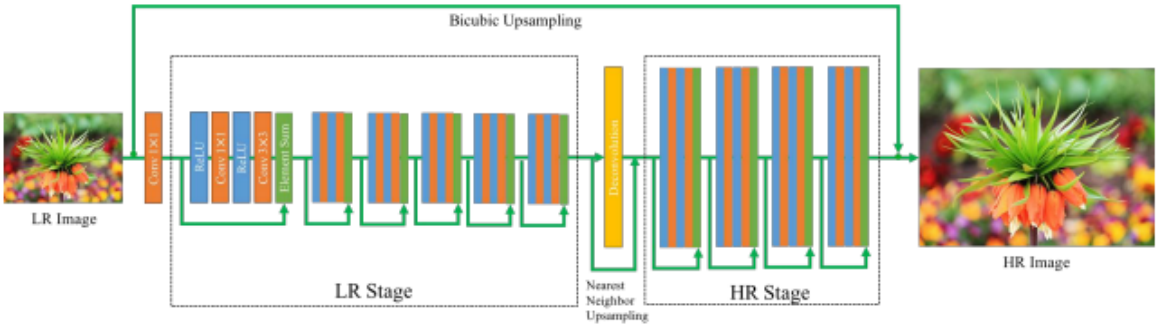


Рисунок 2.12 – Архітектура мережі BTSRN

Згортка на етапі HR вимагає більше обчислень, ніж на етапі LR, оскільки розмір вхідних даних є більшим. Кількість блоків на обох етапах визначається таким чином, щоб досягти компромісу між точністю та продуктивністю.

Вихід LR етапу піддається дискретизації перед передачею на HR етап. Це робиться шляхом додавання результатів шару деконволюції та дискретизації за методом найближчого сусіда. Пропонується новий залишковий блок, PConv, що досягає гарного компромісу між точністю та продуктивністю, виходячи з результатів. Подібно до EDSR, уникаємо пакетної нормалізації, щоб запобігти повторному центруванню та повторному масштабуванню, оскільки вона виявилася шкідливою. Це пов’язано з тим, що надвисока роздільна здатність є регресійною задачею, і цільові результати сильно корелюють зі статистикою першого порядку вхідних даних (рис. 2.13).

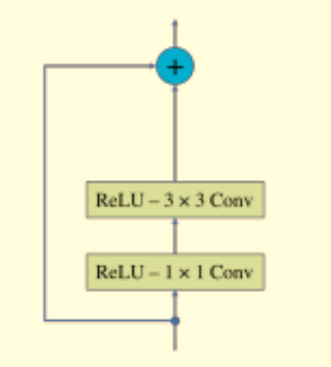


Рисунок 2.13 – Залишковий блок PConv

2.1.5 Рекурсивні мережі

Рекурсивні мережі використовують спільні мережеві параметри у згорткових шарах, щоб зменшити обсяг пам’яті (рис. 2.14).

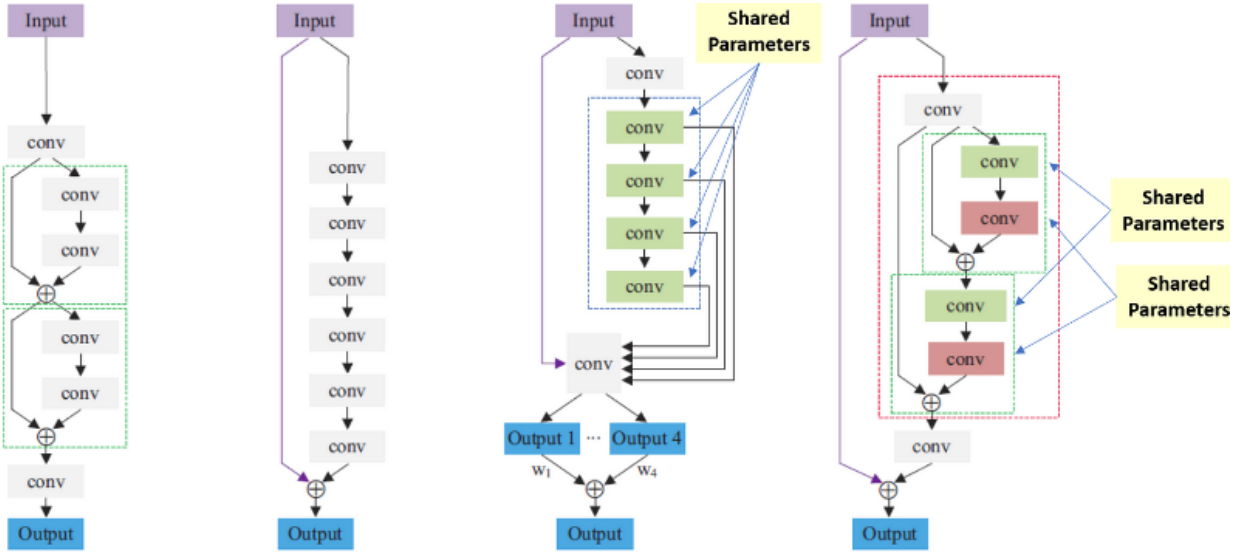


Рисунок 2.14 – Архітектура мереж із рекурсивними блоками:
а) архітектура ResNet; б) архітектура VDSR; в) архітектура DRCN;
г) архітектура DRRN

Глибока рекурсивна згорткова мережа (DRCN) передбачає застосування одного і того ж шару згортки кілька разів. Як видно на рисунку вище, шари

згортки в залишковому блоці є спільними. Вихідні дані з усіх проміжних спільних згорткових блоків, разом з вхідними даними, надсилаються до шару реконструкції, який генерує зображення високої роздільної здатності, використовуючи всі вхідні дані [28, 29]. Оскільки для генерації вихідних даних використовується декілька входів, цю архітектуру можна розглядати як ансамбль мереж (рис. 2.15).

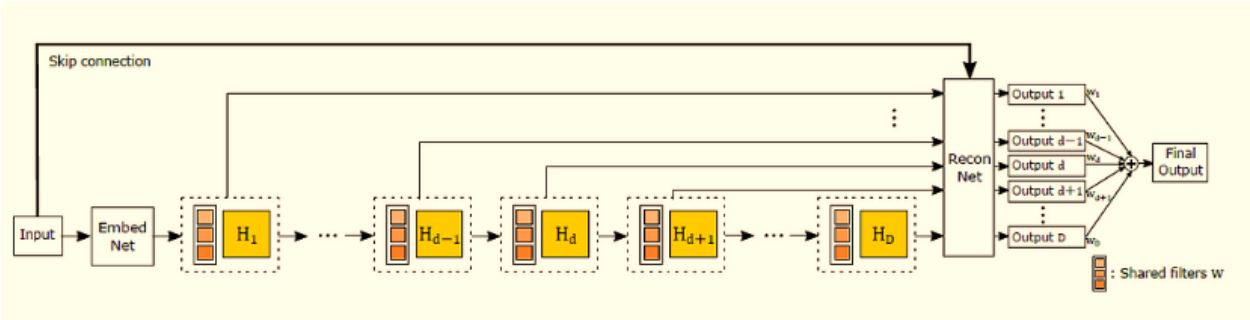


Рисунок 2.15 – Вихідні дані з усіх проміжних спільних згорткових блоків

Глибока рекурсивна залишкова мережа (DRRN) є покращенням DRCN завдяки наявності залишкових блоків у мережі поверх простих згорткових шарів.

Параметри кожного залишкового блоку є спільними для інших залишкових блоків, як видно на зображенні вище. DRRN перевершує SRCNN, ESPCN, VDSR і DRCN, маючи при цьому порівнянну кількість параметрів.

2.1.6 Мережі прогресивної реконструкції

CNN зазвичай видають результати за один знімок, але отримання зображення з високою роздільною здатністю та великим коефіцієнтом масштабування є складним завданням для нейронної мережі [30].

Щоб вирішити цю проблему, деякі мережеві архітектури збільшують роздільну здатність зображень поетапно (рис. 2.16).

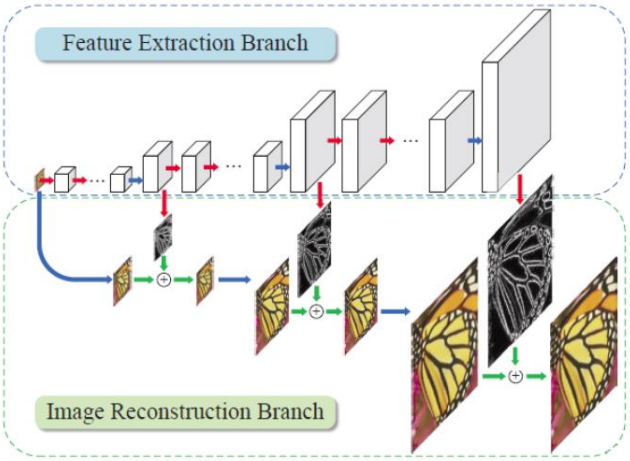


Рисунок 2.16 – Структура LAPSRN

LAPSRN, складається зі структури піраміди Лапласа, яка може збільшувати зображення до 2×, 4× та 8× за допомогою покрокового підходу. LAPSRN складається з декількох етапів. Мережа складається з двох гілок: гілки виділення ознак і гілки реконструкції зображення. Кожен ітераційний етап складається з блоку вбудовування ознак та блоку вибірки ознак, як показано на рисунку нижче.

Вхідне зображення пропускається через шар вбудовування ознак для вилучення ознак у просторі з низькою роздільною здатністю, які потім підвищуються за допомогою транспонованої згортки [31]. Результатом навчання є залишкове зображення, яке додається до інтерпольованого вхідного зображення для отримання зображення з високою роздільною здатністю (рис. 2.17).

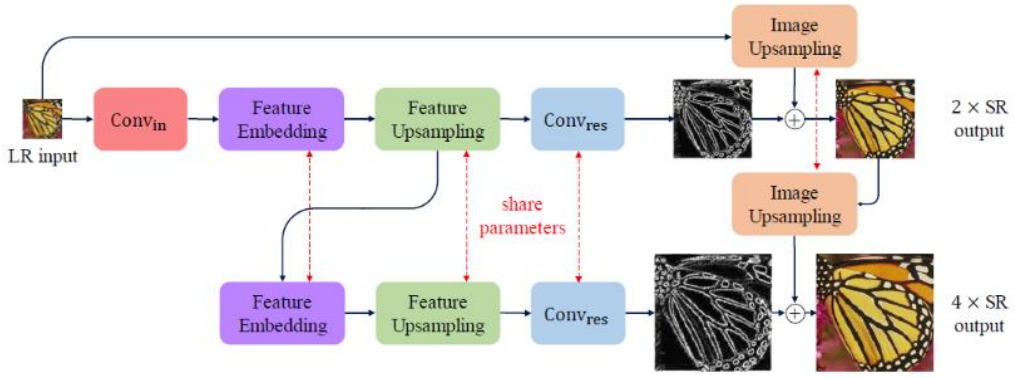


Рисунок 2.17 – Детальна мережева архітектура LAPSRN

Вихідні дані блоку дискретизації ознак також передається на наступний етап, який використовується для уточнення виходу з високою роздільною здатністю цього етапу і масштабування його до наступного рівня. Оскільки вихідні дані з нижчою роздільною здатністю використовуються для уточнення на наступних етапах, відбувається спільне навчання, яке допомагає мережі працювати краще. Щоб зменшити обсяг пам'яті мережі, параметри Feature Embedding, Feature Upsampling і т.д. використовуються на всіх етапах рекурсивно (рис. 2.18).

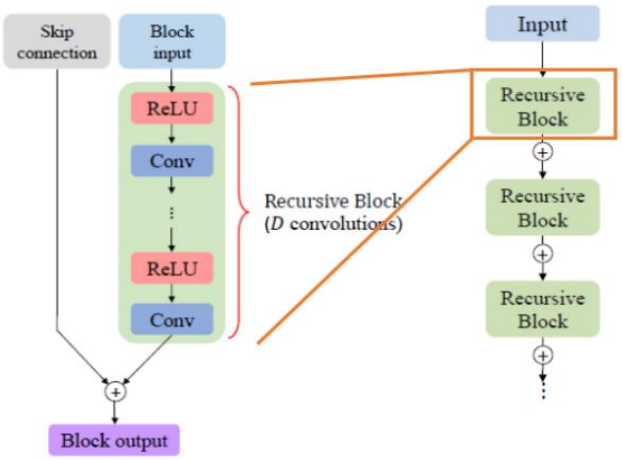


Рисунок 2.18 – Структура рекурсивного блоку

У блоці вбудовування ознак окремий залишковий блок складається зі спільних параметрів згортки, щоб ще більше зменшити кількість параметрів.

Оскільки кожен вхід LR може мати декілька представлень HR, функція втрат L2 дає згладжений вихід за всіма представленнями, через що зображення виглядають нечіткими [32]. Для вирішення цієї проблеми використовується функція втрат Шарбоньє, яка краще справляється з викидами.

2.1.7 Багатогалузеві мережі

Наразі спостерігається тенденція: глибші мережі дають кращі результати. Але навчати глибші мережі складно через проблему потоку

інформації. Залишкові мережі певною мірою вирішують цю проблему, використовуючи короткі зв’язки.

Багатогалузеві мережі працюють над покращенням інформаційного потоку, маючи кілька гілок, через які може проходити інформація, що призводить до об’єднання інформації з кількох рецептивних полів і до кращого навчання [33].

Як і інші фреймворки надвисокої роздільної здатності, CMSC має шар вилучення ознак, каскадні підмережі та шар реконструкції, як показано нижче на рисунку 2.19.

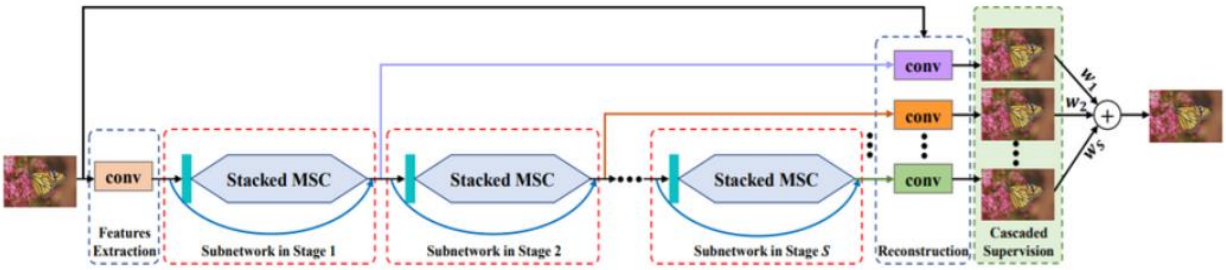


Рисунок 2.19 – Структура мережі CMSC

Каскадна підмережа складається з двох гілок. Кожна гілка має різні розміри фільтрів, а отже, і різні сприйнятливі поля. Злиття інформації з різних сприйнятливих полів у модулі призводить до кращого інформаційного потоку.

Кілька блоків MSC накладаються один за одним, щоб поступово зменшити різницю між вихідним зображенням і зображенням HR ітеративно. Виходи з усіх блоків передаються разом до блоку реконструкції, щоб отримати остаточний вихід HR (рис. 2.20).

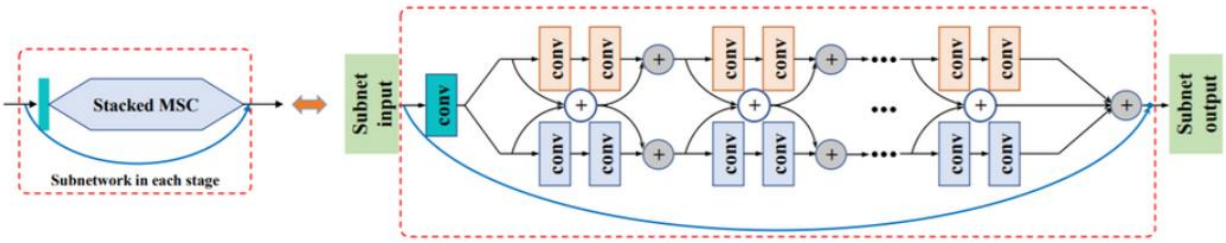


Рисунок 2.20 – Структура каскадної підмережі CMSC

Інформаційна дистиляційна мережа (IDN) пропонується для досягнення швидких і точних результатів для задачі надвисокої роздільної здатності. Як і інші багатогілкові мережі, IDN використовує можливості декількох гілок для покращення інформаційного потоку в глибокій мережі.

Архітектура IDN складається з F-блоку для вилучення ознак, декількох D-блоків та R-блоку для транспонованої згортки для досягнення навченого масштабування. D-блок складається з двох блоків: блоку покращення та блоку стиснення (рис. 2.21).

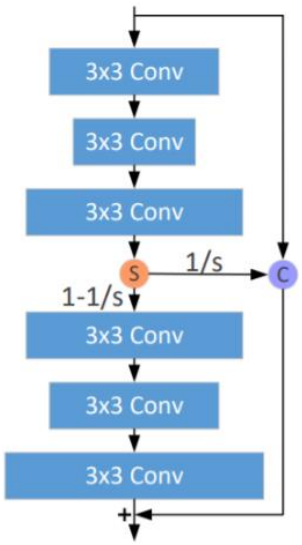


Рисунок 2.21 – Структура блоку покращення

Вхідні дані проходять через три згорточні фільтри розміром 3×3, а потім розрізаються. Одна частина зрізу об’єднується з початковим вхідним сигналом і передається через коротке з’єднання до кінцевого шару. Решта пропускається через інший набір згорткових фільтрів розміром 3×3. Кінцевий результат генерується шляхом підсумовування вхідних даних та кінцевого шару. Така структура допомагає одночасно вловлювати як короткострокову, так і довгострокову інформацію [34]. Блок стиснення приймає вихідний сигнал блоку підсилення і пропускає його через згортковий фільтр 1×1, щоб стиснути (або зменшити) кількість каналів.

2.1.8 Мережі, засновані на увазі

Мережі, про які йшлося вище, надають однакового значення всім просторовим локаціям і каналам. Загалом, вибіркова увага до різних областей на зображенні може дати набагато кращі результати (рис. 2.22).

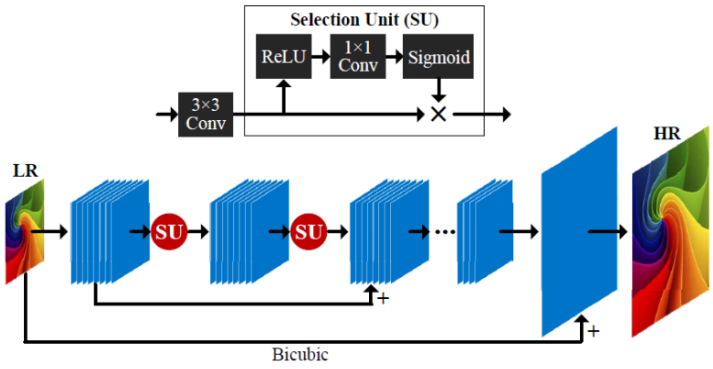


Рисунок 2.22 – Структура SelNet

SelNet пропонує новий модуль селекції в кінці згорткових блоків, який допомагає вирішити, яку інформацію передавати, вибірково. Модуль селекції складається з активації ReLu, за якою слідує згортка 1×1 і сигмоїдний гейтинг. Блок селекції – це множення модуля селекції та ідентифікаційного з'єднання.

Субпіксельний шар (подібний до ESPCN) зберігається в кінці мережі для досягнення навченого масштабування. Мережа запам'ятовує залишкове HR зображення, яке потім додається до інтерпольованих вхідних даних, щоб отримати остаточне HR зображення [35].

Для того, щоб тренувати глибші мережі, мережі залишкової уваги до каналів (RCAN) пропонують модулі RIR з увагою до каналів. Вхідні дані в RCAN проходять через один згортковий фільтр для виділення ознак, який потім обходить до кінцевого шару з довгим пропуском. З'єднання з довгим пропуском додається для передачі низькочастотних сигналів з LR зображення, в той час як основна мережа (тобто RIR) зосереджується на захопленні високочастотної інформації (рис. 2.23).

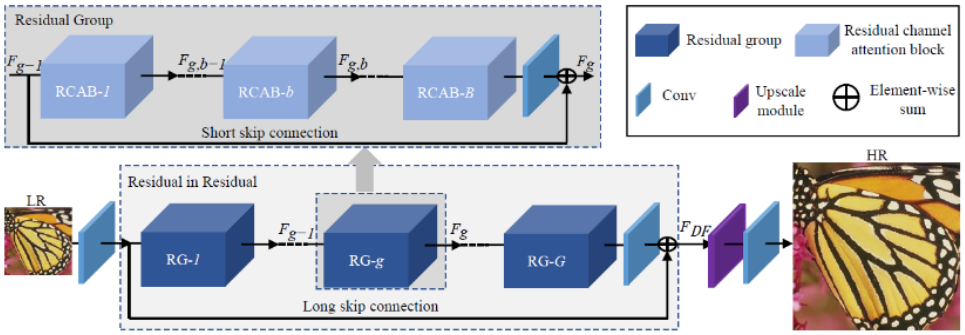


Рисунок 2.23 – Структура модулів RIR

RIR складається з декількох блоків RG, кожен з яких має структуру, показану вище. Кожен блок RG має кілька модулів RCAB разом з пропускним з'єднанням, так званим коротким пропускним з'єднанням, для передачі низькочастотного сигналу (рис. 2.24).

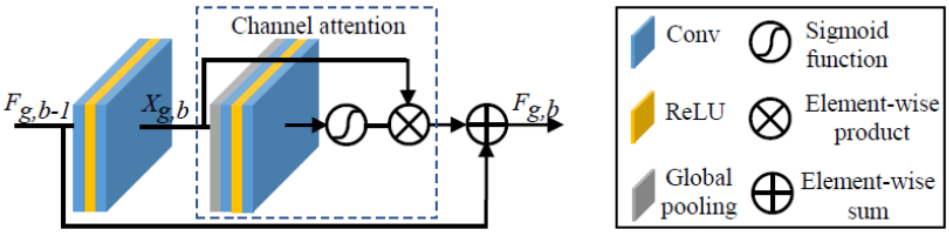


Рисунок 2.24 – Структура RCAB

RCAB має структуру, що складається з модуля для досягнення канальної уваги, подібно до блоків Squeeze і Excite у SqueezeNet. Канальна увага множиться на вихід сигмоїдної стробуючої функції згорткового блоку. Потім цей вихід додається до вхідного з'єднання швидкого доступу, щоб отримати остаточне вихідне значення блоку RCAB.

2.1.9 Генеративні змагальні мережі

Генеративні мережі (GAN) намагаються оптимізувати якість сприйняття, щоб створювати зображення, приємні для людського ока.

SRGAN використовує архітектуру на основі GAN для створення візуально приємних зображень. Він використовує мережеву архітектуру SRResnet як внутрішню частину і застосовує багатозадачну втрату для покращення результатів.

Втрати ж складаються з трьох частин:

– втрати MSE, що фіксують схожість пікселів;

– втрати перцептивної схожості, які використовуються для захоплення високорівневої інформації за допомогою глибокої мережі;

– втрати від дискримінатора.

Хоча отримані результати мали порівняно нижчі значення PSNR, модель досягла більшого значення MOS, тобто кращої перцептивної якості в результатах.

Розглянемо мережу EnhanceNet, вона використовує повністю згорткову мережу із залишковим навчанням, яка використовує додатковий член у функції втрат для захоплення більш тонкої інформації про текстуру (рис. 2.25).

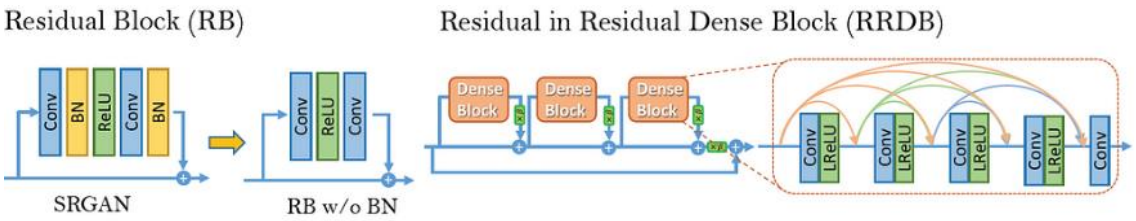


Рисунок 2.25 – Структура EnhanceNet

На додаток до вищеописаних втрат у SRGAN, для захоплення більш тонкої інформації про текстуру використовується втрата текстури, подібна до тієї, що використовується у Style Transfer.

ESRGAN покращує SRGAN, додаючи релятивістський дискримінатор. Перевага полягає в тому, що мережа навчена не тільки визначати, яке зображення справжнє, а яке ні, але й робити реальні зображення менш реальними порівняно зі згенерованими, допомагаючи таким чином обдурити дискримінатор [36]. Пакетна нормалізація в SRGAN також вилучена, а для

кращого потоку інформації використовуються щільні блоки (натхненні DenseNet). Ці щільні блоки називаються RRDB.

2.2 Опис моделі SRGAN

Вхідними даними служить зображення. Позначимо зображення, яке отримується в результаті роботи нейронної мережі як I^{SR}, зображення низької роздільної здатності як I^{LR}. Зображення з високою роздільною здатністю (без обробки) позначимо як I^{HR}. Під час тренування I^{LR} виходить шляхом застосування Гаусівського фільтра до I^{HR}, після чого виконується операція даунсемплінгу з коефіцієнтом даунсемплінгу r. Для зображення з C колірними каналами, I^{LR} описується дійсним тензором розміру $W \times H \times C$ та I^{HR}, а I^{SR} описується як $rW \times rH \times C$ відповідно.

Кінцева мета – навчити генеративну функцію G, яка оцінює для заданого вхідного зображення LR його відповідний HR аналог. Для цього треба навчити генеруючу мережу у вигляді прямого поширення CNN G_{θ_G} за параметром θ_G:

$$\theta_G = \{W_{1:L}; b_{1:L}\}, \tag{2.1}$$

де $W_{1:L}$ – ваги нейронної мережі глибиною в L шарів;

$b_{1:L}$ – зміщення нейронної мережі глибиною в L шарів.

Цей параметр обчислюється шляхом оптимізації складовою функції втрат l^{SR}. Для тренувальних зображень I_n^{HR} з відповідними I_n^{LR}, де $n = 1, \dots, N$, знаходимо:

$$\hat{\theta}_G = \arg\min_{\theta_G} \frac{1}{N} \sum_{n=1}^{N} l^{SR}\left(G_{\theta_G}(I_n^{LR}), I_n^{HR}\right). \tag{2.2}$$

SRGAN використовує функцію нескінченних втрат l^{SR}, яка є зваженою сумою двох компонентів втрат: втрат контенту і втрат від супротивника. Ці втрати дуже важливі для продуктивності архітектури генератора.

Функція втрат l^{SR} складається з наступних компонентів:

$$l^{SR} = l_X^{SR} + 10^{-3} l_{Gen}^{SR}, \tag{2.3}$$

де l_X^{SR} – середня квадратична помилка при порівнянні згенерованого зображення l^{SR} з його оригіналом I^{HR};

l_{Gen}^{SR} – змагальна функція втрат.

Попіксельні втрати MSE для архітектури SRResnet, які є найбільш поширеними втратами MSE для зображень з високою роздільною здатністю, не можуть впоратися з високочастотним контентом у зображенні, що призводить до отримання надто гладких зображень. Тому використовується втрата різних шарів VGG. Ця втрата VGG базується на шарах активації ReLU попередньо навченої 19-шарової мережі VGG.

Попіксельна MSE втрата визначається наступним чином:

$$l_{MSE}^{SR} = \frac{1}{r^2 WH} \sum_{x=1}^{rW} \sum_{y=1}^{rH} \left(I_{x,y}^{HR} - G_{\theta_G}(I^{LR})_{x,y} \right)^2. \tag{2.4}$$

Через $\phi_{i,j}$ позначаємо карту ознак, де j – значення після активації згортки, а i – шар мережі після операції максимального пулінгу. З цього слідує, що VGG втрата визначається як:

$$l_{VGG/i,j}^{SR} = \frac{1}{W_{i,j} H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}\left(G_{\theta_G}(I^{LR})\right)_{x,y})^2, \tag{2.5}$$

де $W_{1:L}$ – ваги нейронної мережі глибиною в L шарів;

$b_{1:L}$ – зміщення нейронної мережі глибиною в L шарів.

До перцептивних втрат додаємо генеративну складову GAN. Це спонукає мережу надавати перевагу рішенням, які знаходяться на множині природних зображень, намагаючись обдурити мережу дискримінатора. Генеративна втрата l_{Gen}^{SR} визначається на основі ймовірностей дискримінатора $D_{\theta_D}(G_{\theta_G}(I^{LR}))$:

$$l_{Gen}^{SR} = \sum_{n=1}^{N} - \log D_{\theta_D}(G_{\theta_G}(I^{LR})), \tag{2.6}$$

де $D_{\theta_D}(G_{\theta_G}(I^{LR}))$ – це ймовірність розпізнавання дискримінатором того, що створене генератором зображення $G_{\theta_G}(I^{LR})$ є зображенням високої роздільної здатності I^{HR}.

Архітектура генератора містить залишкову мережу замість мережі глибокої згортки, тому що залишкові мережі легко тренувати і вони можуть бути значно глибшими, щоб генерувати кращі результати. Це пов’язано з тим, що залишкова мережа використовує тип зв’язків, який називається «пропускні зв’язки» (рис. 2.26).

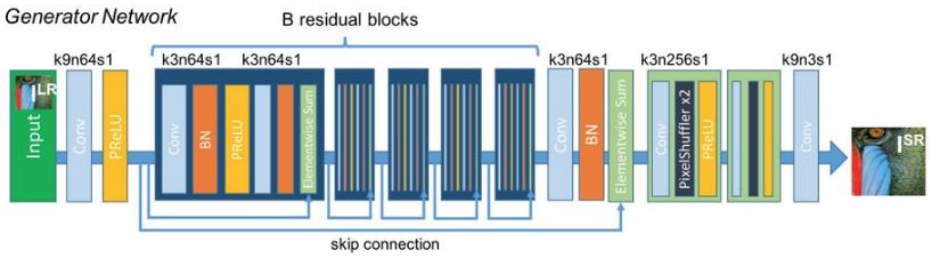


Рисунок 2.26 – Архітектура мережі генератора

Як показано на рисунку вище, існує певна кількість залишкових блоків, створених за допомогою ResNet. У залишковому блоці використовуються два згорточні шари з невеликими ядрами 3×3 і 64 картами ознак, за якими слідують шари пакетної нормалізації та Parametric ReLU як функція активації:

$$f(x) = \begin{cases} x, x > 0 \\ ax, x \leq 0 \end{cases}, \tag{2.7}$$

де x – вхідний сигнал;

a – коефіцієнт, який обчислюється в результаті навчання моделі.

Роздільна здатність вхідного зображення збільшується за допомогою двох навчених шарів субпіксельної згортки.

Ця архітектура генератора також використовує параметричний ReLU як функцію активації, яка замість фіксованого значення параметра випрямляча (альфа), як у Leaky ReLU, використовує фіксоване значення. Він адаптивно вивчає параметри випрямляча і покращує точність при незначних додаткових обчислювальних витратах.

Під час навчання зображення з високою роздільною здатністю (HR) перетворюється на зображення з низькою роздільною здатністю (LR). Архітектура генератора намагається підвищити дискретизацію зображення з низької роздільної здатності до надвисокої [37]. Після цього зображення передається на дискримінатор, який намагається відрізнити зображення з надвисокою роздільною здатністю від зображення з високою роздільною здатністю і генерує втрату противника, яка потім поширюється в архітектурі генератора (рис. 2.27).

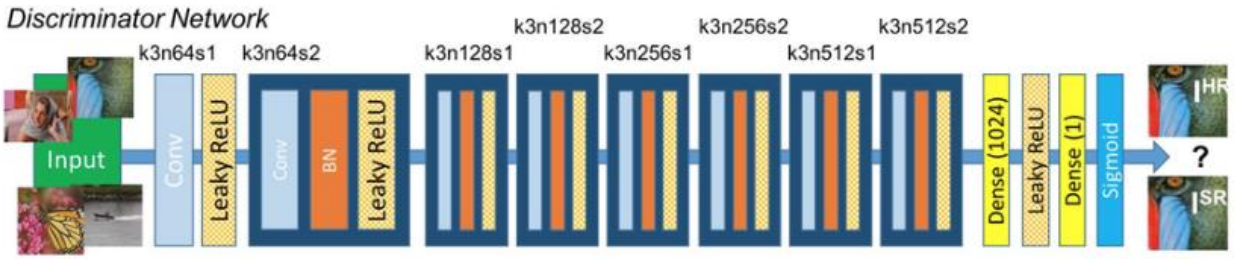


Рисунок 2.27 – Архітектура мережі дискримінатора

Завдання дискримінатора полягає в тому, щоб розрізняти реальні HR зображення та згенеровані SR зображення. Архітектура дискримінатора, що використовується в цій роботі, подібна до архітектури DCGAN з активацією Leaky ReLU. Мережа містить вісім згорткових шарів з ядрами фільтрів 3×3, збільшених у 2 рази з 64 до 512 ядер. Для зменшення роздільної здатності

зображення щоразу, коли кількість ознак подвоюється, використовується стрічкова згортка. До отриманих 512 карт ознак додаються два щільні шари, між якими застосовується фунція Leaky ReLU з $a = 0{,}02$:

$$f(x) = \frac{1}{1+e^{-x}}. \tag{2.8}$$

Фінальна сигмоїдна функція активації для отримання ймовірності класифікації зразків для другого повнозв’язного шару має вигляд:

$$\sigma(x) = \frac{1}{1+e^{-x}}. \tag{2.9}$$

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Вибір мови програмування та бібліотек

Для вирішення різноманітних завдань у сфері машинного навчання та обробки зображень використовуються різні мови програмування, що дозволяє підбирати інструментарій, оптимально відповідний конкретним вимогам завдань. У цьому контексті важливу роль відіграють різноманітні бібліотеки, які включають в себе реалізації найбільш популярних методів та алгоритмів, роблячи розробку застосунків у сфері штучного інтелекту більш ефективною та доступною.

Для даного дослідження було обрано мову програмування Python для вирішення поставлених завдань. Цей вибір обумовлений безліччю факторів, серед яких важливими є:

– популярність у науковому середовищі: Python визначається як одна з найпопулярніших мов програмування в академічному середовищі, особливо для завдань, пов’язаних з машинним навчанням та обробкою інтелектуальних даних . Це робить програми, реалізовані на Python, зрозумілими та доступними для великої кількості науковців і дослідників;

– різноманіття бібліотек та зручність роботи: Python має значну кількість бібліотек, що спрощують використання складних алгоритмів машинного навчання, а також багато бібліотек для роботи з обробкою зображень. Зручність встановлення бібліотеки забезпечується за допомогою вбудованих у мову менеджерів пакетів;

– високий рівень та швидкість розробки: Python є високорівневою мовою програмування, що дозволяє швидко створювати прототипи програм, що сприяє скороченню часу розробки. Модульність Python дозволяє використовувати кожен файл в проєкті як окремий скрипт чи імпортовати окремі методи.

Важливо відзначити, що Python, будучи інтерпретованою мовою програмування, може виявити меншу швидкість у порівнянні з компільованими мовами.

Однак цей недолік частково компенсується використанням бібліотек, в яких була проведена оптимізація з використанням коду на інших мовах. Такий підхід дозволяє зберегти високу продуктивність розробки та забезпечити ефективність виконання завдань у сфері машинного навчання та обробки зображень.

При написанні програмної реалізації використовувалася версія мови 3.6.2. Для програмної реалізації було вирішено використовувати такі основні бібліотеки:

– NumPy;

– OpenCV;

– TensorFlow.

Також використовувалися різні допоміжні пакети, такі, як tqdm для візуалізації прогресу в командному рядку, shutil для роботи з файлами і папками і datetime для визначення часу роботи алгоритмів.

NumPy – це бібліотека для мови програмування Python, яка надає підтримку для великих, багатовимірних масивів та матриць, разом з великою колекцією функцій, що дозволяють здійснювати операції над цими масивами. Вона є важливою частиною екосистеми Python для наукових обчислень та роботи з даними.

Основні характеристики NumPy:

– основною структурою даних в NumPy є масив (ndarray), який представляє собою багатовимірний масив значень одного типу даних. Масиви в NumPy можуть бути одновимірними, двовимірними або багатовимірними;

– NumPy надає велику кількість функцій та методів для виконання операцій над масивами, таких як сума, середнє значення, мінімум, максимум, сортування тощо. Ці операції можна виконувати ефективно та векторизовано, що робить NumPy ефективним для обробки великих обсягів даних;

– механізм броадкастінгу в NumPy дозволяє виконувати операції над масивами різних розмірностей та форм, що робить код більш зрозумілим та компактним;

– NumPy має вбудовану підтримку для генерації випадкових чисел, що корисно у статистичних дослідженнях та машинному навчанні;

– NumPy містить функції для виконання операцій лінійної алгебри, таких як розв’язання систем лінійних рівнянь, обчислення власних значень та власних векторів, операції над матрицями тощо;

– NumPy інтегрується з іншими бібліотеками для обробки даних та візуалізації, такими як Pandas, Matplotlib і SciPy;

– операції в NumPy реалізовані на мові C, що робить їх вкрай ефективними та швидкими.

Отже, NumPy є фундаментальним інструментом для роботи з науковими обчисленнями та аналізом даних у Python. Багато інших бібліотек, таких як TensorFlow, PyTorch та SciPy, використовують NumPy під капотом.При створенні програмної реалізації використовувалася версія NumPy 1.13.3.

OpenCV – це відкрита бібліотека для комп’ютерного зору та машинного навчання. Вона була розроблена для сприяння розробці програм для обробки зображень та комп’ютерного зору. OpenCV написана на C++ та має вбудовані зв’язки з Python, що робить її популярною у розробці застосунків для обробки зображень та відео [38].

Основні характеристики OpenCV:

– OpenCV надає широкий спектр функцій для завантаження, збереження, обробки та аналізу зображень. Це включає в себе операції фільтрації, видалення шуму, визначення контурів, розпізнавання обличчя, виявлення об’єктів тощо;

– OpenCV дозволяє розробляти застосунки для обробки відео, включаючи виявлення та відстеження об’єктів, аналіз руху, розпізнавання обличчя в режимі реального часу тощо;

– OpenCV включає в себе інструменти для роботи з алгоритмами машинного навчання, виявлення об’єктів та розпізнавання образів;

– OpenCV дозволяє проводити калібрування камери для поліпшення точності сприйняття об’єктів у тривимірному просторі. Вона також надає функції для відновлення 3D об’єктів за допомогою зображень;

– OpenCV підтримує відкриті стандарти, включаючи OpenCL та CUDA для оптимізації використання апаратного забезпечення. Вона є кросплатформеною і може працювати на Windows, Linux, macOS та інших операційних системах;

– на відміну від більшості бібліотек, OpenCV має бінди для багатьох мов програмування, включаючи C++, Python, Java та інші.

OpenCV використовується в різних галузях, включаючи комп’ютерний зір, розпізнавання об’єктів, робототехніку, медицину та інші. Завдяки своїй потужній функціональності та активній спільноті, OpenCV є важливим інструментом для розробників, які працюють у сфері комп’ютерного зору та обробки зображень. У програмній реалізації була використана версія бібліотеки 3.3.0.

TensorFlow – це відкрита бібліотека для числових обчислень та глибокого навчання, розроблена компанією Google. Основна мета TensorFlow – надати зручний ефективний інструмент для розробки та тренування моделей машинного або глибокого навчання.

Основні характеристики TensorFlow:

– TensorFlow використовує концепцію графів обчислень, де операції представлені вузлами, а дані – ребрами. Це дозволяє ефективно виконувати паралельні обчислення та оптимізації;

– TensorFlow надає різні рівні API для зручності використання: від низькорівневих (TF Graph) до високорівневих (Keras). Це дозволяє як експертам у глибокому навчанні, так і початківцям зручно використовувати бібліотеку;

– TensorFlow надає можливості оптимізації для використання різних пристроїв, таких як CPU, GPU та TPU. Крім того, TensorFlow Serving дозволяє легко розгортати навчені моделі у виробничне середовище;

– TensorFlow має модульну структуру, що дозволяє додавати різні функції та розширювати функціоналість бібліотеки. Це сприяє створенню великої спільноти та багатьох додаткових інструментів;

– TensorFlow користується великою та активною спільнотою. Існує безліч документації, туторіалів та проєктів, які використовують TensorFlow, що полегшує вивчення та розробку;

– TensorFlow використовується в різних областях, включаючи комп'ютерний зір, обробку природних мов, медицинську аналітику, аналіз даних та інше;

– TensorFlow включає в себе різні компоненти, такі як TensorFlow Lite для мобільних та вбудованих пристроїв, TensorFlow.js для використання в браузерах, TensorFlow Extended (TFX) для розгортання моделей у виробничному середовищі тощо.

TensorFlow є однією з найпопулярніших та найвикористовуваніших бібліотек для глибокого навчання, і вона активно використовується у багатьох наукових, промислових та дослідницьких проєктах [39]. У програмній реалізації використана версія Tensorflow 1.3.0.

3.2 Підготовка бази зображень

У процесі навчання штучних нейронних мереж для досягнення оптимальних результатів важливо забезпечити наявність достатньо великої кількості якісного навчального та тестового матеріалу.

Щоб забезпечити правильну обробку алгоритмом, важливо, щоб зображення мали однаковий формат, якість та розмір. Засоби для цього існують у вигляді різних вільно доступних ресурсів з базами зображень, таких

як CIFAR-10, MNIST, Labelme і інші, які часто поширюються у вигляді .gz або .zip архівів. Для використання у нашому проєкті ми обрали базу зображень ImageNet. Це одна з найбільших та найпопулярніших баз зображень, яка використовується для навчання та тестування моделей глибокого навчання. Ця база була створена з метою сприяння розробці та вдосконаленню алгоритмів розпізнавання об’єктів на зображеннях.

Вибір ImageNet обумовлений кількома причинами:

– ImageNet включає в себе вражаючу кількість більш ніж 14 мільйонів зображень, які охоплюють більше двадцяти тисяч категорій об’єктів. Кожне зображення має призначену мітку, яка ідентифікує клас чи категорію об’єкта на зображенні;

– унікальність ImageNet полягає в тому, що більшість зображень були проаналізовані та класифіковані людьми. Це дозволяє мати високоякісні та достовірні мітки для кожного зображення, що є важливим для тренування точних моделей глибокого навчання;

– ImageNet охоплює широкий спектр категорій, включаючи фотографії людей, тварин, медичні зображення, природні сцени та багато іншого. Це робить її відмінним ресурсом для навчання моделей на різноманітних завданнях;

– категорії в ImageNet організовані у ієрархічну структуру, де вищі рівні представляють більш загальні категорії, а нижчі рівні деталізують класифікацію. Це дозволяє використовувати базу для різноманітних завдань та рівнів складності;

– ImageNet відома своєю участю у відомому конкурсі з комп’ютерного зірю, який називається ImageNet Large Scale Visual Recognition Challenge. Цей конкурс став ключовим для вдосконалення та порівняння різних методів розпізнавання об’єктів;

– ImageNet використовується у багатьох наукових та дослідницьких проєктах з розробки та оцінки моделей глибокого навчання. Багато сучасних архітектур нейронних мереж були навчені та протестовані саме на цій базі.

Для тестування були обрані кольорові фотографії тварин з метою отримання більш деталізованих результатів. Зображення з ImageNet були збережені у форматі JPEG з кольоровим простором RGB, що є оптимальним для зберігання реалістичних зображень з великою кількістю деталей та економією дискового простору.

Також для полегшення обчислень було вирішено використовувати зменшений розмір зображень – 96×96 пікселів.

3.3 Опис програмної реалізації

3.3.1 Загальна структура програми

Програма складається з наступних модулів:

– модуль підготовки зображень;

– модуль алгоритмів інтерполяції;

– модуль генеративної ШНМ SRGAN;

– модуль згорткової ШНМ SRCNN;

– модуль порівняння зображень.

Кожен модуль існує в самостійній папці, повністю незалежний від інших. Рішення розробляти програму з використанням різних модулів було прийнято після аналізу існуючих реалізацій різних моделей. Багато з них були структуровані так, що навіть малі зміни параметрів вимагали значних змін у коді.

Вибір модульної структури програми та централізованого зберігання всіх параметрів дозволяє легко включати нові моделі та алгоритми інтерполяції з мінімальними змінами. Наприклад, модуль інтерполяції та модуль згорткової нейронної мережі були інтегровані в проєкт на основі існуючих реалізацій вкрай швидко.

Це вдалося частково автоматизувати процес порівняння та оцінки ефективності різних алгоритмів. У кореневій папці програми знаходиться

конфігураційний файл config.py, який містить загальні налаштування для всіх модулів, а також параметри, що визначають джерело даних. Файл включає шляхи до папок з вихідними зображеннями, розмір зображень, розмір пакета вихідних даних під час навчання та інші.

Всі модулі використовують єдине джерело даних, розташоване в папці «data». В папці «train» знаходяться вихідні зображення високої роздільної здатності для тренування нейронних мереж, а в папці «test» – зображення такої ж роздільної здатності для тестування.

Папки «train_low» і «test_low» містять зображення низької роздільної здатності, що використовуються як вхідні дані для алгоритмів збільшення зображень і моделей ШНМ під час тренування та тестування відповідно.

Взаємодія з користувачем вирішується за допомогою консольного інтерфейсу, який надає різну інформацію та відображає прогрес роботи алгоритмів. Результати кожного модуля зберігаються у відповідних файлах, а сам кожен модуль є самостійним і може бути запущений окремо.

3.3.2 Модуль підготовки зображень

Модуль prepare_data відповідає за підготовку зображень і включає два скрипти: download_images.py і prepare_data.py. Скрипт download_images.py відповідає за завантаження, збереження і первинну обробку зображень.

Для використання цього скрипта необхідно розмістити файл links.txt із посиланнями на зображення з бази ImageNet в одній папці з ним. Файл із посиланнями можна завантажити з сайту бази ImageNet. Якщо потрібно використовувати інший файл з посиланнями, слід замінити значення змінної links_file в файлі config.py.

Під час реалізації виникла проблема, пов’язана з недоступністю деяких зображень за вказаними посиланнями в базі.

Ці ситуації вдалося виявити за такими ознаками:

– зміна посилання на зображення, що свідчило про переадресацію на сторінку із повідомленням про недоступність зображення. Цю проблему вирішено порівнянням посилань у файлі links.txt і фактичного посилання, з якого завантажено зображення;

– зображення мало розміри 1×1 піксель. Причиною цієї проблеми була помилка на сервері із зображенням. Вирішено це перевіркою розмірів зображення по висоті і ширині порівняно з розмірами, заданими в файлі конфігурації. Це також дозволило уникнути збору зображень із надто малою роздільною здатністю.

Отже, скрипт download_images.py перевіряє кожне посилання з файлу links.txt, а також перевіряє, щоб ширина і висота зображення не були меншими, ніж зазначено в конфігураційному файлі. Функції бібліотеки OpenCV використовуються для зміни розмірів зображень та їх збереження у форматі jpg.

Первинна обробка зображень включає зменшення розмірів зображення і кадрування до розмірів, заданих в конфігураційному файлі. Зменшення розмірів виконується відповідно до масштабу, вказаного у конфігураційному файлі за допомогою алгоритму бікубічної інтерполяції.

Скрипт prepare_data.py відповідає за збереження завантажених зображень у вигляді масиву NumPy в файл з роздільною здатністю npy. Це зроблено для того, щоб під час навчання не потрібно було повторно завантажувати та обробляти всі зображення, а замість цього одразу отримувати інформацію у вигляді масивів високої роздільної здатності.

Для цього використовуються бібліотеки OpenCV (для збереження зображень), NumPy (для перетворення зображень у масив, з яким може працювати OpenCV) і бібліотека shutil (для роботи з папками). Для завантаження зображень використовується стандартна бібліотека Python urllib.request.

3.3.3 Модуль алгоритмів інтерполяції

Цей модуль містить в собі реалізацію збільшення зображень за допомогою алгоритмів бікубічної інтерполяції і фільтра Ланцоша. Для цього використовуються функція бібліотеки OpenCV resize з параметрами interpolation = cv2.INTER_CUBIC для бікубічної інтерполяції і interpolation = cv2.INTER_LANCZOS4 для фільтра Ланцоша. Результати роботи кожного алгоритму зберігаються в окремій папці, шлях до якої задається в конфігураційному файлі.

Лістинг 3.1 Реалізація бікубічної інтерполяції для зміни розмірів зображення:

```
import cv2
def resize_image_bicubic(image_path, output_path, scale_factor):
    img = cv2.imread(image_path)
    height, width = img.shape[:2]
    new_height = int(height * scale_factor)
    new_width = int(width * scale_factor)
    resized_img = cv2.resize(img, (new_width, new_height), interpolation=cv2.INTER_CUBIC)
    cv2.imwrite(output_path, resized_img)
```

Лістинг 3.2 Реалізація фільтра Ланцоша для зміни розмірів зображення:

```
def resize_image_lanczos(image_path, output_path, scale_factor):
    img = cv2.imread(image_path)
    height, width = img.shape[:2]
    new_height = int(height * scale_factor)
    new_width = int(width * scale_factor) resized_img = cv2.resize(img, (new_width, new_height), interpolation=cv2.INTER_LANCZOS4)
```

```
    cv2.imwrite(output_path, resized_img)
```

3.3.4 Модуль порівняння зображень

Даний розділ складається з єдиного скрипта compare.py, в якому впроваджені метрики порівняння зображень, такі як PSNR, SSIM і MSE. Навіть при наявності бібліотек, які вже мають готові реалізації цих метрик, не вдалося знайти бібліотеку, де б усі три метрики були представлені одночасно. З цієї причини, а також для уникнення нових залежностей від сторонніх бібліотек, було вирішено створити власну реалізацію цих метрик.

У цьому модулі були створені функції getMSE, getPSNR і getSSIM, в які передаються масиви, отримані за допомогою функції читання зображень бібліотеки OpenCV cv2.imread. Ці функції повертають значення відповідної метрики.

Лістинг 3.3 Реалізація функції для порівняння зображень:

```
import tensorflow as tf
from tensorflow.keras.metrics import Mean
from skimage.metrics import peak_signal_noise_ratio as psnr
from skimage.metrics import structural_similarity as ssim
import numpy as np
def getMSE(original, generated):
    mse = tf.keras.losses.MeanSquaredError()
    mse_value = mse(original, generated).numpy()
    return mse_value
def getPSNR(original, generated):
    psnr_value = psnr(original, generated, data_range=original.max())
    return psnr_value
def getSSIM(original, generated):
```

```
    ssim_value, _ = ssim(original, generated, full=True)
    return ssim_value
mse_result = getMSE(original_image, generated_image)
psnr_result = getPSNR(original_image, generated_image)
ssim_result = getSSIM(original_image, generated_image)
```

Модуль порівнює оригінальні зображення, збережені в data/test, із зображеннями, отриманими в результаті роботи алгоритмів і ШНМ. Шлях до папок із згенерованими зображеннями вказується в файлі конфігурації.

Отримані результати зберігаються у формі csv-файлів, що дозволяє швидко завантажувати їх у різні табличні редактори для подальшого аналізу та представлення. Крім того, виводяться середні результати по всіх метриках для кожного алгоритму в консольне вікно.

У цьому модулі використані бібліотеки OpenCV і NumPy для математичних операцій над зображеннями. OpenCV також використовується для читання зображень. Додатково був задіяний стандартний пакет Python csv для збереження результатів порівняння зображень у форматі csv.

3.3.5 Модуль SRGAN

Цей розділ включає скрипти та код, необхідні для функціонування реалізованої моделі GAN (SRGAN). Модуль складається з наступних файлів: srgan.py, test.py та train.py. Крім того, в модулі є папка results, в якій зберігаються зображення, отримані в результаті роботи моделі, та папка backup, яка містить проміжний або фінальний стан моделі в залежності від того, чи завершено навчання чи ні.

У всіх файлах модуля використовується бібліотека TensorFlow. З метою полегшення читання коду були написані обгорткові методи. Виявлено, що виділення таких функцій в окремий модуль або клас є загальною практикою,

існують також бібліотеки, такі як TensorLayer, які надають розширені можливості для роботи з TensorFlow.Файл layer.py містить функції, які відповідають за певні аспекти моделі, такі як генератор і дискримінатор.

У файлі srgan.py реалізована модель SRGAN у вигляді окремого класу. Генератор і дискримінатор представлені як методи класу. Конструктор моделі отримує параметри: data, low_res, is_training та batch_size.

Файл train.py містить функцію, яка здійснює навчання нейронної мережі. Функція завантажує тестовий набір даних і зображень низької роздільної здатності, нормалізує їх і передає в конструктор моделі SRGAN. У цьому файлі також обирається оптимізатор для втрат, використовуючи функцію бібліотеки TensorFlow tf.train.AdamOptimizer.

У файлі test.py реалізована функція для тестування нейронної мережі. Функція завантажує тестові зображення з низькою і високою роздільною здатністю та читає бекап із збереженим станом мережі. Після обробки кожної пари тестових зображень результат зберігається у форматі JPEG.

Обидва файли train.py і test.py використовують пакет datetime для вимірювання часу та, у випадку test.py, додатково використовують OpenCV для збереження результатів у вигляді JPEG зображень.

3.4 Дослідження отриманих результатів

3.4.1 Опис експериментів

Навчання нейронних мереж проводилось на комп’ютері з наступною конфігурацією: процесор Intel Core i7-8550U CPU 1.80GHz, відеокарта Radeon(TM) 540 Graphics 2GB, оперативна пам’ять 8 Гб. Обчислення проводилися на відеокарті. Операційна система Windows 10 Pro 64-bit. Для навчання нейронних мереж було використано 5072 кольорових зображень розміру 96×96 пікселів.

Процес підготовки зображень описаний в попередньому розділі. Кількість зображень, використаних при тестуванні, склала 256. Для навчання та тестування використовувалися різні зображення, тому що це дозволило б у разі виникнення проблеми перенавчання відразу її помітити. На вхід алгоритмів надходили зображення розміром 24×24 пікселя. Ці зображення збільшувалися в 4 рази, тобто до вихідний розмірів 96×96 пікселів. Розмір навчального набору (Батч) дорівнював 16. Таким чином, кількість ітерацій, тобто тренувальних наборів, оброблених кожною мережею, склала 317. Кількість повних проходжень тренувальних наборів для обох нейронних мереж (циклів) дорівнювала 200.

Після навчання моделі SRGAN було проведено повторне навчання, але в цей раз після обробки кожного тренувального набору підключався модуль тестування і результат обробки зберігався у вигляді зображення. Це було зроблено для того, щоб мати можливість побачити динаміку поліпшення якості генеруються зображень в міру навчання нейронної мережі.

Приклад такої динаміки представлений на рисунках 3.1 – 3.5.

Рисунок 3.1 – Цикл навчання №1

Рисунок 3.2 – Цикл навчання №10

Рисунок 3.3 – Цикл навчання №50

Рисунок 3.4 – Цикл навчання №100

Рисунок 3.5 – Цикл навчання №200

Як можна помітити, візуально складно побачити різницю між 100-м і 200-м циклом, однак якщо подивитися на будь-яку з вибраних метрик якості, то на продовженні всього процесу навчання зберігається позитивна динаміка. Картинка поступово стає більш чіткою і на ній можна побачити більше деталей.

Показники представлені нижче (табл. 3.1).

Таблиця 3.1 – Зміна значення якості в процесі навчання

Номер циклу	PSNR	SSIM	MSE
1	12,00	0,13	1577,13
10	22,01	0,73	291,47
50	27,01	0,87	125,32
100	27,37	0,89	118,07
200	27,47	0,90	116,02

Динаміка змін показників на прикладі PSNR представлена на рисунку 3.6. Як видно з рисунка 3.6, швидкість навчання починає значно сповільнюватися вже після 10-го циклу, але тривала динаміка поліпшення показника зберігається аж до 200-го циклу. Після того, як був закінчений процес навчання і тестування нейронний мереж, були згенеровані результати роботи алгоритмів інтерполяції (рис. 3.6).

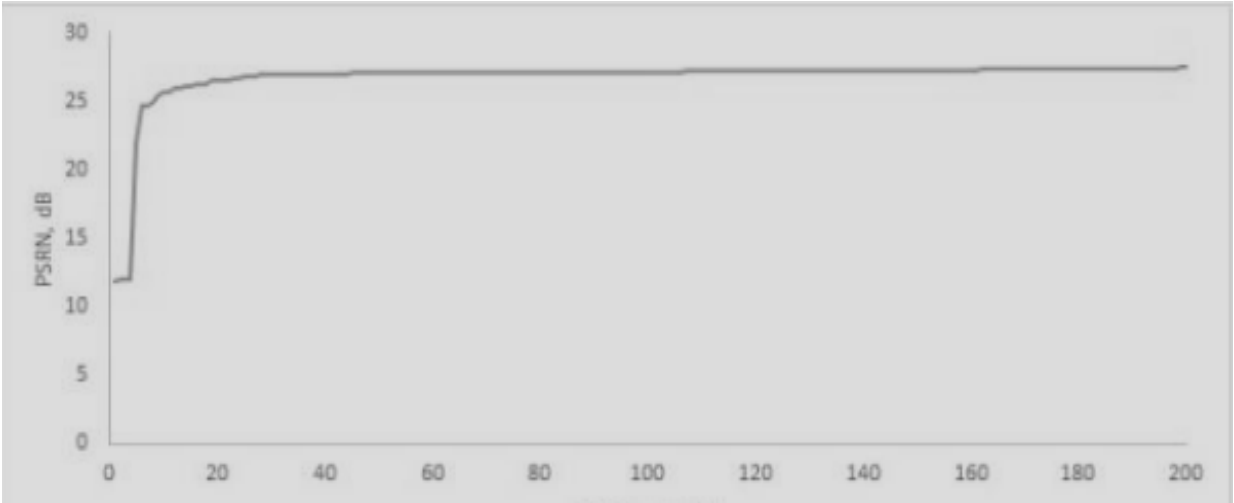


Рисунок 3.6 – Графік зміни показника PSNR в процесі навчання

3.4.2 Порівняння навчання моделей ШНМ і роботи алгоритмів

Для моделі SRCNN час навчання склав 53 хвилини 26 секунд, тобто середній час на одну повну обробку тестових зображень склав 16 секунд. У

моделі SRGAN навчання зайняло значно більше часу – всього близько 8 години 43 хвилин. Відповідно, середній час на обробку всіх тестових наборів дорівнює 2 хвилинам 37 секундам.

Навантаження на процесор при навчанні було приблизно однакове для обох моделей і становило в середньому 36%. Навантаження на відеокарту в середньому теж було приблизно однакове більшу частину часу роботи мереж і становило 21%. При проведенні тестування моделей час обробки тестових зображень склав 27 секунд для моделі SRCNN і 33 секунди для SRGAN. Алгоритм бікубічної інтерполяції і фільтр Ланцоша час на обробку тестових зображень зайняв 0,002005 і 0,561503 секунд відповідно.

Для обох нейронних мереж були використані функції бібліотеки TensorFlow для збереження проміжного або фінального стану. Файл зі збереженим станом нейронної мережі SRCNN становить 441 мегабайт, SRGAN – 490 мегабайт.

3.4.3 Порівняння результатів роботи алгоритмів і моделей ШНМ за обраних метрик

Були обрані наступні метрики для перевірки: MSE, PSNR і SSIM. Приклад результату, отриманого в результаті роботи алгоритмів і ШНМ, представлений на рисунках 3.7 – 3.12. Спочатку подивимось на оригінальне забраження:

Рисунок 3.7 – Оригінальне зображення

Рисунок 3.8 – Зменшене зображення

Рисунок 3.9 – Зображення після бікубічної інтерполяції

Рисунок 3.10 – Зображення після обробки фільтра Ланцоша

Рисунок 3.11 – Зображення після SRCNN

Рисунок 3.12 – Зображення після SRGAN

Метрики щодо рисунків представлених вище, буде наведено нижче у таблиці 3.2.

Таблиця 3.2 – Порівняння результатів на прикладі одного зображення

Метрика	**Бікубічна інтерполяція**	**Фільтр Ланцоша**	**SRCNN**	**SRGAN**
MSE	774,61	786,71	182,08	115,09
PSNR	19,24	19,17	25,53	27,53
SSIM	0,41	0,40	0,80	0,91

Середні показники результатів обробки всіх зображень представлені в таблиці 3.3.

Таблиця 3.3 – Порівняння середніх показників всіх зображень

Метрика	**Бікубічна інтерполяція**	**Фільтр Ланцоша**	**SRCNN**	**SRGAN**
MSE	588,61	603,65	200,80	130,25
PSNR	21,26	21,15	26,94	27,63
SSIM	0,57	0,56	0,89	0,92

Як видно з таблиці 3.3, результат роботи моделі SRGAN має найкращі показники. Обидві моделі ШНМ значно перевершують алгоритми інтерполяції, проте між самими моделями відмінність в показниках не така значна.

ВИСНОВКИ

У рамках кваліфікаційної роботи була здійснена програмна реалізація , використовуючи алгоритми бікубічної інтерполяції, фільтра Ланцоша, та моделей удосконалення зображень, зокрема SRCNN і SRGAN (додаток А).

Аналіз сучасних методів удосконалення зображень підкреслив, що, незважаючи на існуючі швидкі та ефективні алгоритми, постійно з’являються нові підходи. У останні роки все частіше для вирішення цього завдання використовуються штучні нейронні мережі, зокрема різні варіації згорткових нейронних мереж. Недавно, генеративно-змагальні нейронні мережі.

Вибір алгоритмів інтерполяції базується на їх ефективності в контексті співвідношення часу виконання та якості результату. Також враховано їх широке використання в різних застосунках.

Обрання моделі SRCNN обумовлено тим, що ця модель вже завоювала визнання у вирішенні завдань покращення зображень. З іншого боку, SRGAN було обрано через вражаючі результати роботи, і його новизна на ринку програмних реалізацій.

Створена програмна реалізація дозволяє не лише порівнювати зазначені методи, але також легко підключати інші алгоритми та моделі удосконалення зображень.

У дослідженні описано та реалізовано об’єктивні метрики для оцінювання результатів алгоритмів збільшення роздільної здатності, такі як Mean Squared Error (MSE), Peak Signal-to-Noise Ratio (PSNR), та Structural Similarity Index (SSIM).

Порівнюючи результати за допомогою об’єктивних метрик, можна відзначити, що генеративно-змагальна модель проявляє ефективність у обробці зображень порівняно зі згортковою нейронною мережею, але вимагає значно більше часу для навчання. Використання готової моделі SRGAN для обробки тестових зображень також займає більше часу, ніж SRCNN, але отримані результати значно перевершують методи інтерполяції.

Важливо відзначити, що нейронні мережі характеризуються великим розміром програмної реалізації. Розмір файлу, що містить стан нейронної мережі, для обох моделей перевищує 400 мегабайт, що є недоліком у відношенні до алгоритмів інтерполяції, чий розмір може становити кілька кілобайт. Це може ускладнити включення нейронних мереж у програмні реалізації для кінцевих користувачів. Однак отримані результати переважають недоліки, представлені нейронними мережами.

Результати дослідження апробовано у вигляді тез доповідей під час VI Міжнародної науково-практичної конференції «Methodical and practical methods of creating inventions» [40].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Kim, J., Lee, J. K., & Lee, K. M. (2016). Accurate image super-resolution using very deep convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1646-1654).

2. Zhang, K., Zuo, W., Chen, Y., Meng, D., & Zhang, L. (2017). Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. IEEE transactions on image processing, 26(7), 3142-3155.

3. Tvoroshenko I., Gorokhovatskyi V., Kobylin O., and Tvoroshenko A. (2023) Application of deep learning methods for recognizing and classifying culinary dishes in images, International Journal of Academic and Applied Research, 7(9), pp. 57-70.

4. Timofte, R., Agustsson, E., Van Gool, L., Yang, M. H., & Zhang, L. (2017). Ntire 2017 challenge on single image super-resolution: Methods and results. In Proceedings of the IEEE conference on computer vision and pattern recognition workshops (pp. 114-125).

5. Кобилін, О. А., & Творошенко, І. С. (2021). Методи цифрової обробки зображень.

6. Gorokhovatskyi V., Tvoroshenko I. (2023) Identification of visual objects by the search request. International scientific symposium «INTELLIGENT SOLUTIONS-S». Computational intelligence (results, problems and perspectives). Decision making theory: proceedings of the international symposium, September 28, 2023, Kyiv-Uzhorod, Ukraine, pp. 25-27.

7. Gorokhovatskyi V., Tvoroshenko I., Kobylin O., and Vlasenko N. (2023) Search for visual objects by request in the form of a cluster representation for the structural image description, Advances in Electrical and Electronic Engineering, 21(1), pp. 19-27.

8. Гороховатський В.О., Творошенко І.С., Чмутов Ю.В. (2022) Застосування систем ортогональних функцій для формування простору ознак у методах класифікації зображень, Сучасні інформаційні системи, 6(3), С. 5-12.

9. Гороховатський В., Передрій О., Творошенко І., Марков Т. (2023) Матриця відстаней для множини компонентів структурного опису як інструмент для створення класифікатора зображень, Сучасні інформаційні системи, 7(1), С. 5-13.

10. Mashtalir, V., Ruban, I., & Levashenko, V. (Eds.). (2020). Advances in Spatio-Temporal Segmentation of Visual Data. Springer.

11. Mashtalir, S., Mashtalir, V., & Stolbovyi, M. (2018, August). Representative based clustering of long multivariate sequences with different lengths. In 2018 IEEE second international conference on Data Stream Mining & Processing (DSMP) (pp. 545-548). IEEE.

12. Kattenborn, T., Leitloff, J., Schiefer, F., & Hinz, S. (2021). Review on Convolutional Neural Networks (CNN) in vegetation remote sensing. ISPRS journal of photogrammetry and remote sensing, 173, 24-49.

13. Tvoroshenko, I., & Babochkin, O. (2021). Object identification method based on image keypoint descriptors.

14. Путятін, Є. П., Гороховатський, В. О., & Матат, О. О. (2006). Методи та алгоритми комп'ютерного зору: навч. посібник.

15. Reyad, M., Sarhan, A. M., & Arafa, M. (2023). A modified Adam algorithm for deep neural network optimization. Neural Computing and Applications, 1-18.

16. Pomazan V., Tvoroshenko I., and Gorokhovatskyi V. (2023) Handwritten character recognition models based on convolutional neural networks, International Journal of Academic Engineering Research, 7(9), pp. 64-72.

17. Dong, C., Loy, C. C., He, K., & Tang, X. (2014). Learning a deep convolutional network for image super-resolution. In Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part IV 13 (pp. 184-199). Springer International Publishing.

18. Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., ... & Shi, W. (2017). Photo-realistic single image super-resolution using a generative adversarial network. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4681-4690).

19. Zhang, K., Gool, L. V., & Timofte, R. (2020). Deep unfolding network for image super-resolution. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 3217-3226).

20. Dong, C., Loy, C. C., & Tang, X. (2016). Accelerating the super-resolution convolutional neural network. In Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14 (pp. 391-407). Springer International Publishing.

21. Timofte, R., De Smet, V., & Van Gool, L. (2015). A+: Adjusted anchored neighborhood regression for fast super-resolution. In Computer Vision--ACCV 2014: 12th Asian Conference on Computer Vision, Singapore, Singapore, November 1-5, 2014, Revised Selected Papers, Part IV 12 (pp. 111-126). Springer International Publishing.

22. Zhang, Y., Tian, Y., Kong, Y., Zhong, B., & Fu, Y. (2018). Residual dense network for image super-resolution. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2472-2481).

23. Tai, Y., Yang, J., Liu, X., & Xu, C. (2017). Memnet: A persistent memory network for image restoration. In Proceedings of the IEEE international conference on computer vision (pp. 4539-4547).

24. Lai, W. S., Huang, J. B., Ahuja, N., & Yang, M. H. (2017). Deep laplacian pyramid networks for fast and accurate super-resolution. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 624-632).

25. Mao, X. J., Shen, C., & Yang, Y. B. (2016). Image restoration using convolutional auto-encoders with symmetric skip connections. arXiv preprint arXiv:1606.08921.

26. Zhang, K., Tao, D., Gao, X., Li, X., & Xiong, Z. (2015). Learning multiple linear mappings for efficient single image super-resolution. IEEE Transactions on Image Processing, 24(3), 846-861.

27. Bhardwaj, A. & Singh, R. (2020) Handwritten devanagari character recognition usingdeep learning – Convolutional Neural Network (CNN) model, PalArch's Journal ofArchaeology of Egypt/Egyptology, 17(6), pp. 7965–7984.

28. Lai, W. S., Huang, J. B., Ahuja, N., & Yang, M. H. (2018). Fast and accurate image super-resolution with deep laplacian pyramid networks. IEEE transactions on pattern analysis and machine intelligence, 41(11), 2599-2613.

29. Ahn, N., Kang, B., & Sohn, K. A. (2018). Fast, accurate, and lightweight super-resolution with cascading residual network. In Proceedings of the European conference on computer vision (ECCV) (pp. 252-268).

30. Lim, B., Son, S., Kim, H., Nah, S., & Mu Lee, K. (2017). Enhanced deep residual networks for single image super-resolution. In Proceedings of the IEEE conference on computer vision and pattern recognition workshops (pp. 136-144).

31. Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A. P., Bishop, R., ... & Wang, Z. (2016). Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1874-1883).

32. Wronski, B., Garcia-Dorado, I., Ernst, M., Kelly, D., Krainin, M., Liang, C. K., ... & Milanfar, P. (2019). Handheld multi-frame super-resolution. ACM Transactions on Graphics (ToG), 38(4), 1-18.

33. Hu, Y., Gao, X., Li, J., Huang, Y., & Wang, H. (2018). Single image super-resolution via cascaded multi-scale cross network. arXiv preprint arXiv:1802.08808.

34. Gerchberg, R. W. (1974). Super-resolution through error energy reduction. Optica Acta: International Journal of Optics, 21(9), 709-720.

35. Yang, C. Y., & Yang, M. H. (2013). Fast direct super-resolution by simple functions. In Proceedings of the IEEE international conference on computer vision (pp. 561-568).

36. Rosebrock, A. (2017). Deep learning for computer vision with python: Starter bundle. PyImageSearch.

37. Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). The elements of statistical learning: data mining, inference, and prediction (Vol. 2, pp. 1-758). New York: springer.

38. Howse J., Minichino J. (2020). Learning OpenCV 4 Computer Vision with Python 3: Get to grips with tools, techniques, and algorithms for computer vision and machine learning. Packt Publishing.

39. Wang X.Y. (2023). TensorFlow: 100 Interview Questions (Advanced Topics in Machine Learning Book 3).

40. Лещенко, Н. К. (2023, October). ДОСЛІДЖЕННЯ МЕТОДІВ ОТРИМАННЯ ЗОБРАЖЕНЬ З ВИСОКОЮ РОЗДІЛЬНОЮ ЗДАТНІСТЮ. In The 6th International scientific and practical conference «Methodical and practical methods of creating inventions»(October 24–27, 2023) Sofia, Bulgaria. International Science Group. 2023. 282 p. (p. 252).