for student admission»], *Upravlinnia proektamy, systemnyi analiz i lohistyka – Project Management, Systems Analysis and Logistics*, vol. 9, pp. 7-13, 2012. (Ukr.)
4. V. Makoyedova, «Modelyuvannya suprovodu vstupnoyi kampaniyi zakladu vyshchoyi osvity» [«Modeling the support of the admission campaign of a higher education institution»], *Informatsiini tekhnolohii ta suspilstvo – Information Technologies and Society*, vol. 1(7), pp. 44-49, 2023. **doi: 10.32689/maup.it.2023.1.6**. (Ukr.)
5. M.B. Kolomiiets, and R.F. Mirnyi, «Vstupna kampaniya zakladu vyshchoyi osvity yak systema» [«Admission campaign of a higher education institution as a system»], *Hlukhivskoho natsionalnoho pedahohichnoho universytetu imeni Oleksandra Dovzhenka. Seriia: Pedahohichni nauky – Bulletin of Oleksandr Dovzhenko Hlukhiv national pedagogical university*, vol. 3, pp. 105-111, 2017. (Ukr.)

Стаття надійшла 22.11.2024
Стаття прийнята 05.12.2024

УДК 004.932.4: 656.051 **doi: 10.31498/2225-6733.49.1.2024.321212**

© Марченко І.Ф.[1], Балалаєва О.Ю.[2], Коротенко Г.М.[3], Таразанов М.О.[4]

ДОСЛІДЖЕННЯ АЛГОРИТМІВ СТИСНЕННЯ ЗОБРАЖЕНЬ ІЗ ВИКОРИСТАННЯМ НЕЙРОННИХ МЕРЕЖ

У статті наведено результати дослідження алгоритмів стиснення зображень на основі нейронних мереж. Проаналізовано класичні методи стиснення, такі як JPEG, PNG, GIF, TIFF, а також виокремлено переваги нейромережевих методів, зокрема використання автокодувальника, варіаційного автокодувальника, генеративних змагальних мереж. Зроблено висновок, що основними перевагами нейромережевих методів є збереження високого рівня текстур та деталей при низьких бітрейтах, а також можливість роботи з високоякісними зображеннями, хоча це вимагає значних обчислювальних ресурсів. Проведено порівняльний аналіз класичних алгоритмів стиснення, таких як JPEG, з новими підходами на основі нейронних мереж на прикладі автокодувальника, а також оцінено перспективи нейронних мереж у вирішенні проблеми стиснення даних. Головний акцент зосереджено на аналізі якості відновлення зображень та рівня стиснення з використанням різних налаштувань нейронної мережі. Наведено математичну модель, яка описує принцип роботи автокодувальника та показує, як нейронна мережа кодує та відновлює зображення, використовуючи латентний простір. Для досягнення найкращої якості реконструкції використано гібридну функцію втрат, яка складається з трьох компонентів: перцептивної втрати на основі VGG16, SSIM-втрати та MSE-втрати. Для проведення експериментів розроблено модульну програмну систему за допомогою мови програмування Python. Програмне забезпечення включає в себе графічний інтерфейс, модуль стиснення для виконання операцій кодування та декодування зображення за допомогою моделі автокодувальника, а також модуль оцінки якості для розрахунку основних метрик якості (PSNR та SSIM). Встановлено, що традиційні методи стиснення зображень демонструють високу ефективність, але при цьому більше

[1] *канд. техн. наук, доцент, ДВНЗ «Приазовський державний технічний університет», м. Дніпро/Маріуполь, ORCID: 0000-0002-4566-3866, marchenko_i_f@pstu.edu*
[2] *канд. техн. наук, доцент, ДВНЗ «Приазовський державний технічний університет», м. Дніпро/Маріуполь, ORCID: 0000-0003-1461-4399, balalaeva_e_u@pstu.edu*
[3] *д-р техн. наук, доцент, НТУ «Дніпровська політехніка», м. Дніпро, ORCID: 0000-0003-3774-5260, korotenko.g.m@nmu.one*
[4] *магістр, ДВНЗ «Приазовський державний технічний університет», м. Дніпро/Маріуполь, tarazanov_m_o@pstu.edu*

схильні до генерації артефактів, особливо при високих рівнях стиснення, ніж нейромережеві методи. У результаті проведених досліджень встановлено, що модель автокодувальника може кодувати та декодувати зображення з мінімальною втратою якості, на рівні з JPEG, проте поступається класичним алгоритмам у швидкості (1,6 секунди на зображення проти 0,02 для JPEG) та ступені стиснення (модель забезпечує зменшення розміру файлу на 11-18%). Зроблено висновок, що без зменшення потреби в обчислювальних ресурсах нейромережеві методи стиснення не зможуть замінити класичні методи.
Ключові слова: *автокодувальник, стиснення зображень, нейронні мережі, JPEG, алгоритми стиснення.*

I. Marchenko, O. Balalaieva, H. Korotenko, M. Tarazanov. Research of image compression algorithms using neural networks. *The article presents the results of the study of image compression algorithms based on neural networks. Classical compression methods, such as JPEG, PNG, GIF, TIFF, are analyzed, and the advantages of neural network methods, in particular the use of an autoencoder, a variational autoencoder, and generative adversarial networks, are highlighted. It is concluded that the main advantages of neural network methods are the preservation of a high level of textures and details at low bitrates, as well as the ability to work with high-quality images, although this requires significant computing resources. A comparative analysis of classical compression algorithms, such as JPEG, with new approaches based on neural networks is carried out using the example of an autoencoder, and the prospects of neural networks in solving the problem of data compression are assessed. The main emphasis is placed on the analysis of the quality of image restoration and the level of compression using different neural network settings. A mathematical model is presented that describes the principle of operation of an autoencoder and shows how a neural network encodes and restores images using latent space. To achieve the best reconstruction quality, a hybrid loss function was used, which consists of three components: perceptual loss based on VGG16, SSIM loss, and MSE loss. A modular software system was developed using the Python programming language to conduct experiments. The software includes a graphical interface, a compression module for performing image encoding and decoding operations using an autoencoder model, and a quality assessment module for calculating the main quality metrics (PSNR and SSIM). It was found that traditional image compression methods demonstrate high efficiency, but are more prone to generating artifacts, especially at high compression levels, than neural network methods. As a result of the research, it was found that the autoencoder model can encode and decode images with minimal loss of quality, on a par with JPEG, but is inferior to classical algorithms in speed (1.6 seconds per image versus 0.02 for JPEG) and compression ratio (the model provides a reduction in file size by 11-18%). It is concluded that without reducing the need for computational resources, neural network compression methods will not be able to replace classical methods.*
Keywords: *autoencoder, image compression, neural networks, JPEG, compression algorithms.*

Постановка проблеми. Розвиток інформаційних технологій та постійне збільшення обсягу цифрового контенту призводять до зростаючої потреби в ефективних методах стиснення даних. Зображення займають значну частину всіх цифрових даних, а ефективне стиснення є критично важливим для їх зберігання та подальшої передачі. Більшість традиційних методів стиснення під час роботи втрачають частину інформації та генерують артефакти, що погіршують якість зображення. Використання нейронних мереж для стиснення зображень має великий потенціал щодо забезпечення високої якості зображення у порівнянні з класичними методами при однаковому ступені стиснення. Даний напрям потребує подальшого вивчення та удосконалення для широкого впровадження в промислові та комерційні системи.

Протягом останнього десятиліття з'явилося багато досліджень, що спрямовані на оптимізацію стиснення зображень за допомогою нейронних мереж. X. Li та S. Ji, у роботі «Neural Image Compression and Explanation» [1], показали, що нейронні мережі здатні не лише зменшувати

обсяг зображень, але й зберігати їх важливі семантичні деталі. Їхня методика дозволяє стискати зображення до 60% від початкового розміру без втрати значущої інформації. Дослідження, проведене J. Ballé та іншими авторами в роботі [2], продемонструвало переваги end-to-end оптимізованого стиснення, яке використовує нелінійні трансформації для покращення якості відновлених зображень у порівнянні з традиційними методами. Хоча ці роботи демонструють великий потенціал нейронних мереж у стисненні зображень, їх впровадження часто вимагає великих обчислювальних ресурсів, що робить ці методи недосяжними для багатьох комерційних додатків.

Актуальність теми дослідження полягає в необхідності детального вивчення сучасних підходів до стиснення зображень, що враховують обмеження традиційних методів. Використання нейронних мереж для стиснення зображень є інноваційним напрямком, який має потенціал покращити ефективність та якість стиснення. Дослідження в цій галузі є важливими для подальшого розвитку технологій, особливо в контексті зростаючих обсягів цифрових даних.

Метою даної роботи є дослідження ефективності алгоритмів стиснення зображень на основі нейронних мереж та порівнянні їх з традиційними методами.

Аналіз останніх досліджень і публікацій. Класичні методи стиснення зображень (JPEG, PNG, GIF, TIFF) мають як сильні, так і слабкі сторони. Наприклад, JPEG добре підходить для фотографій, але може створювати артефакти при сильному стисненні, тоді як PNG забезпечує високу якість, проте має більший розмір файлів. Вибір методу стиснення залежить від конкретних вимог до зображень, таких як якість, обсяг і тип контенту. Нейронні мережі можуть запропонувати нові підходи до стиснення зображень. Найбільша перевага нейронних мереж полягає у здатності навчатися на великих наборах даних, знаходити приховані патерни та виділяти головну інформацію на зображенні. Також нейронні мережі можуть бути адаптовані до специфічних типів зображень, таких як медичні чи супутникові знімки, що робить їх гнучким і універсальним інструментом.

Автокодувальник – різновид нейронної мережі, що використовується для кодування та декодування інформації [3]. Зазвичай використовується для зниження розмірності інформації або видалення шуму. Автокодувальник складається з двох частин: 1) кодувальник – відповідає за стиснення вхідних даних, під час стиснення мережа вивчає найважливіші характеристики даних, а також відкидає шуми та непотрібну інформацію; 2) декодувальник – бере стиснену інформацію та намагається відновити оригінальні дані якомога точніше. Перевагами використання автокодувальника є простота реалізації та налаштування, а також можливість адаптації до різних типів даних. В якості недоліків можна зазначити, що при використанні автокодувальника якість стиснення може бути нижчою порівняно з більш складними підходами, а латентний простір часто є лінійним, що обмежує можливості для роботи зі складними даними.

Варіаційний автокодувальник є розширенням класичного автокодувальника. Головною відмінністю є те, що замість визначеного вектора ознак у латентному просторі варіаційний автокодувальник використовує ймовірнісний розподіл для кодування вхідних даних [4]. Це дозволяє йому створювати більш узагальнені та гнучкі представлення даних. Варіаційний автокодувальник має гнучкий латентний простір, що дозволяє кодувати більш складні ознаки, може створювати нову інформацію на основі даних, яких мережа не бачила під час тренування, а також краще працює зі складними, нерівномірними розподілами даних. При цьому слід зауважити, що використання варіаційного кодувальника характеризується складнішим тренуванням порівняно з класичним автокодувальником та витратою більшого обсягу обчислювальних ресурсів.

Генеративні змагальні мережі (GAN) – це клас алгоритмів штучного інтелекту, що використовуються в некерованому навчанні, реалізовані системою двох штучних нейронних мереж, які змагаються одна з одною в рамках гри з нульовою сумою [5]. Одна мережа генерує кандидатів (генератор), а інша оцінює їх (дискримінатор). Як правило, генеративна мережа навчається будувати відповідності з латентного простору до певного розподілу даних, тоді як дискримінаційна мережа розрізняє представників справжнього розподілу даних та кандидатів, вироблених генератором. Метою навчання мережі є збільшення частоти помилок дискримінаційної мережі. Перевагами GAN є здатність працювати зі складними патернами та потенціал для генерації нових даних. Серед недоліків GAN можна виділити складне навчання, більший час та обсяг обчислювальних ресурсів для навчання, а також ризик генерації артефактів.

Зважаючи на переваги та недоліки методів, що були представлені вище, було прийнято рішення використовувати автокодувальник для проведення експериментів, що обумовлено простотою його реалізації та особливостями його роботи.

Розглянемо існуючі програмні засоби для стиснення зображень. У роботі [1] розглядається нова система, яка об'єднує пояснення згорткових нейронних мереж (CNN) та семантичне стиснення зображень в єдиний кінцевий процес. Автори розробляють структуру, яка пояснює передбачення CNN і водночас стискає вхідні зображення для ефективного зберігання або передачі. Цей метод пропонує інноваційне рішення, поєднуючи прозорість нейронних мереж та високу ефективність стиснення, що робить його особливо корисним для застосування в областях з обмеженими ресурсами для зберігання або передачі даних. В якості недоліків даного програмного забезпечення можна зазначити те, що маска, яку генерує алгоритм, може бути менш точною для зображень або класів, які не були використані для навчання, що призводить до неточностей у поясненнях або менш ефективного стиснення. Також вибір параметрів, таких як розмір блоків для стиснення, може впливати на компроміс між якістю зображення та рівнем стиснення.

Робота [2] представляє новий підхід до стиснення зображень за допомогою глибинних нейронних мереж, який оптимізований від початку до кінця з урахуванням компромісу між швидкістю передачі даних (rate) і спотвореннями (distortion). Автори використовують нелінійні трансформації, натхненні біологічними моделями нейронів, які значно покращують якість стиснених зображень у порівнянні зі стандартними методами JPEG та JPEG 2000. Метод демонструє суттєві покращення у стисненні зображень, особливо при низьких бітрейтах, що робить його перспективним для майбутнього використання в реальних додатках. Недоліками даного підходу є той факт, що оптимізація всіх параметрів моделі вимагає значних обчислювальних ресурсів і часу, а використання GDN та інших нелінійних трансформацій робить модель більш складною у впровадженні у порівнянні зі стандартними алгоритмами, такими як JPEG.

У роботі [6] представлено новий підхід до втратного стиснення зображень, який використовує GAN для забезпечення високої візуальної якості відновлених зображень при низьких бітрейтах. Основна ідея полягає у поєднанні генеративних моделей з методами стиснення, що дозволяє зберігати текстури та деталі зображень навіть при значному зменшенні їхнього розміру. GAN використовується для поліпшення якості відновлених зображень за допомогою дискримінатора, який «вчить» генератор генерувати реалістичні зображення. Введення втрат на основі перцептивної якості (Perceptual losses) дозволяє досягти високої схожості між відновленими та оригінальними зображеннями. Даний метод поєднує передові підходи у нейромережах та стисненні даних, забезпечуючи високу якість відновлених зображень. Але метод має наступні недоліки: зображення з маленькими деталями або текстом можуть втратити якість, особливо при дуже низьких бітрейтах; впровадження GAN для стиснення потребує значних обчислювальних ресурсів під час навчання, що може бути перешкодою для широкого використання на пристроях з обмеженими ресурсами.

Таким чином, серед основних переваг нейромережевих методів можна зазначити збереження високого рівня текстур та деталей при низьких бітрейтах, а також можливість роботи з високоякісними зображеннями. Проте дані методи вимагають значних обчислювальних ресурсів та можуть мати проблеми з точністю для дуже малих деталей або тексту. Можна зробити висновок, що нейронні мережі дозволяють досягнути ефективного стиснення зображень, покращуючи баланс між розміром файлу та візуальною якістю, проте перед широким впровадженням цей напрям потребує більш глибоких досліджень.

Виклад основного матеріалу. Модель автокодувальника являє собою складну систему функцій, що виконує кодування та декодування вхідного зображення через нейронні мережі. Модель шукає таке латентне представлення даних, яке мінімізує втрати інформації при відновленні зображення [7].

Автокодувальник можна представити як пару функції:

– кодувальник $E(X) = Z$, який зводить вхідне зображення $X \in \mathbb{R}^{H \times W \times C}$ до латентного простору $Z \in \mathbb{R}^{d}$, де $d \ll H \times W \times C$, H – висота, W – ширина, C – кількість каналів;

– декодувальник $D(Z) = \hat{X}$, який відновлює зображення $\hat{X}$, з латентного представлення Z.

Метою є знайти такі набори параметрів (ваг і зсувів) для кодувальника θ_E та декодувальника θ_D, які мінімізують різницю між вхідним зображенням X та відновленим $\hat{X}$.

Кодувальник виконує послідовні згорткові операції:

$$Z = f(X) = ReLU(Conv(X, W_1) + b_1), \quad (1)$$

де $Conv$ – операція згортки, W_1 – матриця ваг фільтрів, b_1 – зсув, $ReLU$ – функція активації.

Після кількох згорток отримуємо латентний вектор:

$$Z \in \mathbb{R}^d, \text{ де } d \ll H \times W \times C. \quad (2)$$

Декодувальник відновлює зображення з латентного простору за допомогою транспонованих згорткових шарів:

$$\hat{X} = g(Z) = Sigmoid(ConvTranspose(Z, W_2) + b_2), \quad (3)$$

де $ConvTranspose$ – операція транспонованої згортки, а $Sigmoid$ – функція активації, яка обмежує значення пікселів у діапазоні [0, 1].

Алгоритм навчання моделі:

1) вхідне зображення X проходить через кодувальник та декодувальник, отримуючи $\hat{X}$;

2) використовується різниця між X та $\hat{X}$ для оцінки якості відновлення;

3) градієнти обчислюються та використовуються для оновлення ваг мережі;

4) алгоритм стохастичного градієнтного спуску або його варіанти використовуються для мінімізації різниці між X та $\hat{X}$

Особливості латентного простору:

– латентний простір має меншу кількість параметрів, ніж вхідні дані, що дозволяє стискати інформацію;

– латентне представлення містить лише важливі для реконструкції ознаки, усуваючи зайві деталі;

– латентний простір передбачає лінійну структуру, що може обмежувати можливість моделі працювати з дуже складними даними.

Наведена модель описує принцип роботи автокодувальника та показує, як нейронна мережа кодує та відновлює зображення, використовуючи латентний простір. Під час цього процесу частина інформації втрачається, у результаті чого зменшується розмір вихідного зображення. За умови, що модель навчена правильно, ця інформаціє не є важливою для візуального сприйняття зображення, тому можна говорити, що відбувається стиснення з втратами (Lossy). Цей підхід є основою для подальших досліджень у даній роботі.

Для того щоб навчити модель правильно виділяти ознаки на зображенні та досягти найкращої якості реконструкції, використовується гібридна функція втрат, яка складається з трьох компонентів: перцептивної втрати на основі VGG16, SSIM-втрати та MSE-втрати.

Перцептивна втрата (Perceptual Loss) використовує проміжні ознаки з нейронної мережі VGG16, яка була попередньо навчена на датасеті ImageNet. Вона забезпечує оцінку схожості між істинними зображеннями y_{true} та передбаченим y_{pred} на рівні ознак (feature maps), а не лише на рівні пікселів.

Формалізація:

$$L_{perc}(y_{true}, y_{pred}) = \frac{1}{n}\sum_{i=1}^{n}(\phi(y_{true})_i - \phi(y_{pred})_i)^2, \quad (4)$$

де ϕ – функція, що представляє обчислення ознак за допомогою VGG16; n – кількість значень у ознаках (feature maps); $\phi(y_{true})_i$ та $\phi(y_{pred})_i$ – значення і-тої ознаки для істинного та передбаченого зображень.

Ця частина функції втрат допомагає мережі зберігати високорівневі семантичні деталі зображень.

SSIM-втрата (Structural Similarity Index Loss – Індекс структурної подібності) вимірює схожість між двома зображеннями, беручі до уваги структурні характеристики, такі як яскравість, контраст, текстуру [8].

Формула для розрахунку SSIM:

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}, \quad (5)$$

де x та y – області зображень, що порівнюються; μ_x та μ_y – середні значення інтенсивності пікселів для блоків x та y; σ_x^2 та σ_y^2 – дисперсії інтенсивності для блоків x та y; σ_{xy} – коваріація між x та y; C_1 та C_2 – малі сталі, які запобігають діленню на нуль, зазвичай визначаються як

p-ISSN: 2225-6733; e-ISSN: 2519-271X

$C_1 = (K_1 L)^2$ і $C_2 = (K_2 L)^2$, де L – динамічний діапазон пікселів (255 для 8-бітних зображень), K_1 і K_2 – невеликі сталі, зазвичай 0,01 та 0,03.

Оскільки SSIM є мірою схожості (значення ближче до 1 означає кращу подібність), функція втрат визначається як:

$$L_{SSIM}(y_{true}, y_{pred}) = 1 - SSIM(y_{true}, y_{pred}), \quad (6)$$

де $SSIM \in [0,1]$ – значення подібності між зображеннями.

Ця функція покращує структурну подібність між вхідним і відновленим ображеннями.

MSE-втрата (Mean Squared Error Loss) вимірює середнє квадратичне відхилення між пікселями істинного та передбаченого зображень:

$$L_{MSE}(y_{true}, y_{pred}) = \frac{1}{m}\sum_{i=1}^{m}(y_{true,i} - y_{pred,i})^2, \quad (7)$$

де m – кількість пікселів у зображенні; $y_{true,i}$ та $y_{pred,i}$ – значення інтенсивності і-го пікселя у відповідних зображеннях.

Ця частина функції втрат мінімізує числову різницю між оригінальним та відновленим зображеннями.

Гібридна функція втрат – це лінійна комбінація трьох описаних вище втрат:

$$L_{hybrid}(y_{true}, y_{pred}) = a \cdot L_{SSIM}(y_{true}, y_{pred}) + b \cdot L_{perc}(y_{true}, y_{pred}) + c \cdot L_{MSE}(y_{true}, y_{pred}), \quad (8)$$

де a, b, c – коефіцієнти значущості показника, $a + b + c = 1$.

Така функція втрат дозволяє балансувати між числовою точністю, структурною подібністю та високорівневими ознаками.

Для реалізації експерименту за допомогою мови програмування Python було розроблено модульну програмну систему (рисунок 1), яка включає наступні компоненти:

– графічний інтерфейс користувача – забезпечує взаємодію з користувачем, вибір моделі для стиснення та робочих папок, а також за відображення результатів та графіків.

– модуль стиснення – виконує операції кодування та декодування зображення за допомогою моделі автокодувальника;

– модуль оцінки якості – виконує розрахунки основних метрик якості, а саме PSNR та SSIM.

Рис. 1 – Графічний інтерфейс користувача

p-ISSN: 2225-6733; e-ISSN: 2519-271X

Під час розробки програми були використані наступні сторонні бібліотеки: tensorflow версії 2.17.0, keras, numpy, pillow, matplotlib, scikit-image, ttkbootstrap, opencv-python, scipy.

Для проведення експериментів було розроблено 4 конфігурації моделі автокодувальника: з 3 шарами, 5 шарами, 7 шарами та 10 шарами згортки.

Розглянемо результати роботи кожної з конфігурації та порівняємо їх з методом JPEG.

Результати роботи автокодувальника з 3 шарами приведено на рисунку 2.

Рис. 2 – Результати роботи моделі з 3 шарами

Модель досягла показника SSIM на рівні 0,97, що відповідає налаштуванням якості 95 зі 100 для JPEG (де 100 – найвища якість зображення). Показник SSIM близький до 1 свідчить про високу структурну схожість стиснутого зображення з оригіналом. Графік розподілу SSIM представлений на рисунку 3.

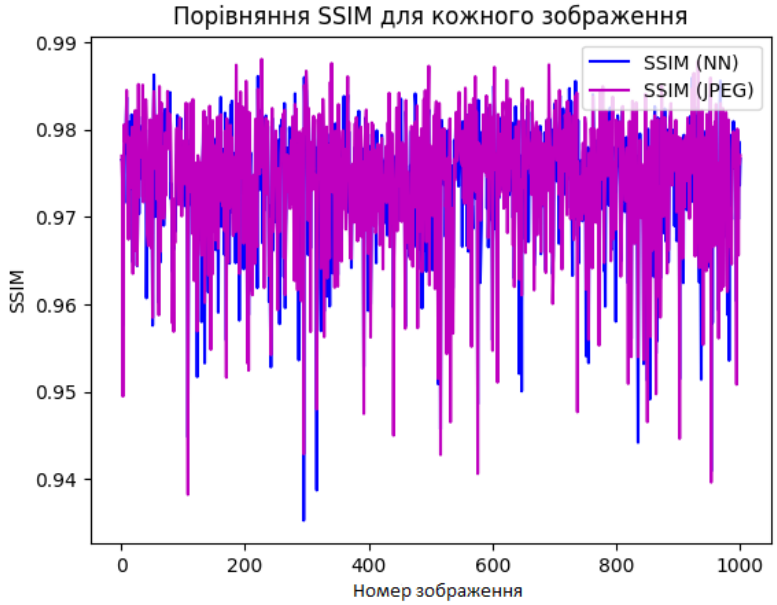

Рис. 3 – Розподіл SSIM для моделі з 3 шарами

PSNR для JPEG становить 41,11 dB, тоді як для автокодувальника – 27,09 dB. Більше значення PSNR у JPEG вказує на те, що з математичної точки зору зображення, яке було стиснуто JPEG, ближче до оригіналу, ніж зображення, яке було стиснуто моделлю. Попри те, що PSNR є важливим показником якості, він не завжди точно відображає візуальне сприйняття. Розподіл значень PSNR наведено на рисунку 4.

p-ISSN: 2225-6733; e-ISSN: 2519-271X

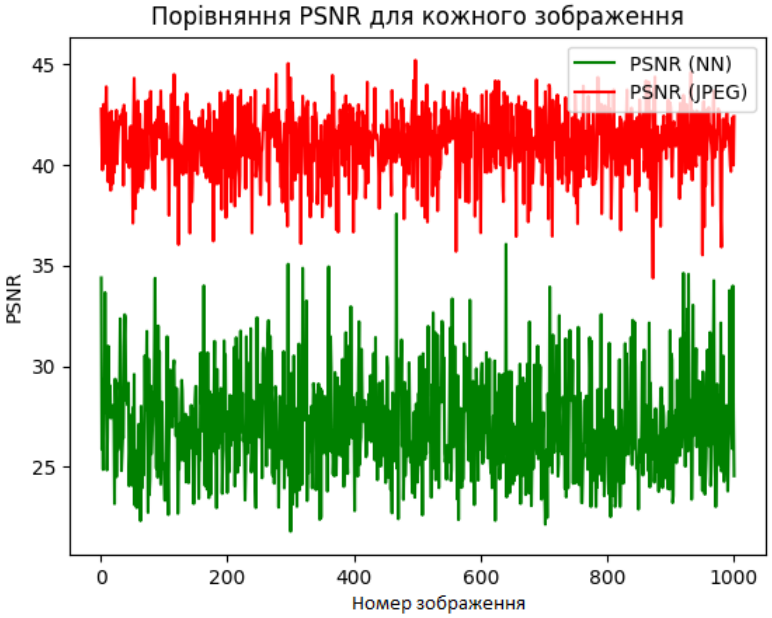

Рис. 4 – Графік розподілу PSNR для моделі з 3 шарами

Ступінь стиснення у JPEG (4,58) майже в 4 рази перевищує ступінь стиснення моделі (1,11). Отже, автокодувальник на даному етапі демонструє обмежену ефективність стиснення. Це говорить про те, що модель навчилась ефективно кодувати та декодувати зображення без суттєвої втрати інформації, що позитивно впливає на якість вихідних зображень, але негативно впливає на ступінь стиснення. Розподіл ступеня стиснення представлено на рисунку 5.

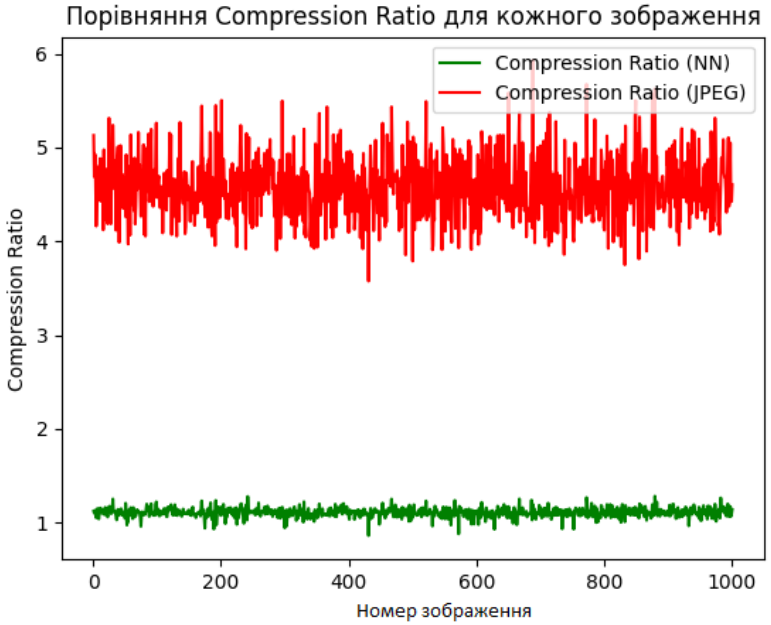

Рис. 5 – Розподіл Compression ratio для моделі з 3 шарами

JPEG значно перевершує автокодувальник за швидкодією: 92,07 секунд проти 1519,46 секунд. Це очікуваний результат, оскільки алгоритми на основі нейронних мереж зазвичай потребують більше обчислювальних ресурсів.

Візуальне порівняння вихідних зображень представлено на рисунку 6 та 7.

p-ISSN: 2225-6733; e-ISSN: 2519-271X

(а) (б) (в)

Рис. 6 – Порівняння деталей зображення для моделі з 3 шарами: оригінальне зображення (а); автокодувальник (б); JPEG (в)

(а) (б)

Рис. 7 – Порівняння деталей зображення для моделі з 3 шарами: модель (а); JPEG (б)

Візуально обидва зображення схожі на оригінал та не мають явних артефактів.

Ці результати вказують на те, що, хоча автокодувальник досягає високої якості зображення за SSIM, йому ще бракує ефективності в стисненні та швидкодії порівняно з JPEG.

Розглянемо роботу автокодувальника з 5 шарами та порівняємо з JPEG (рисунок 8).

Рис. 8 – Результати роботи моделі з 5 шарами

p-ISSN: 2225-6733; e-ISSN: 2519-271X

Модель показує SSIM 0,93, що приблизно відповідає налаштуванням 75/100 для JPEG. У порівнянні з моделлю з 3 шарами бачимо невелике падіння в якості (рисунок 9).

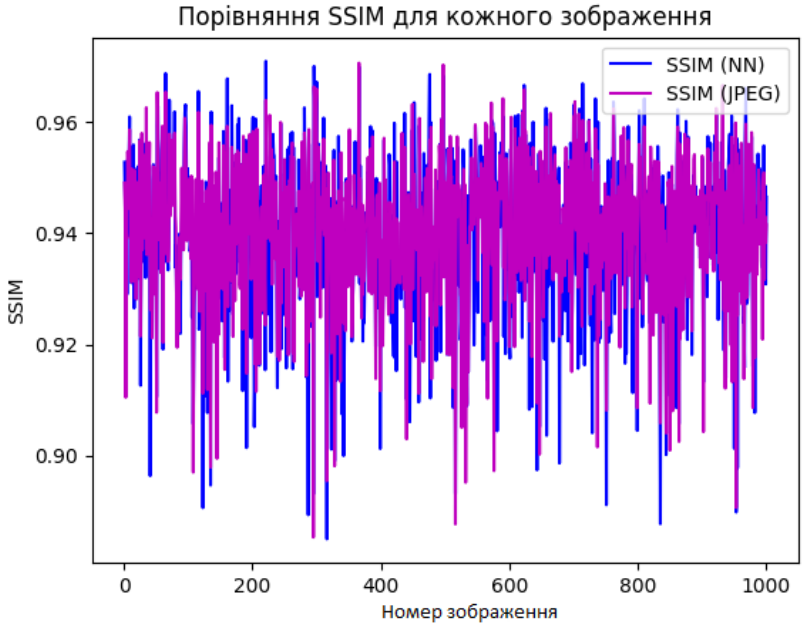

Рис. 9 – Розподіл SSIM для моделі з 5 шарами

Модель з 5 шарами також поступається JPEG за показником PSNR (25,78 проти 36,37), проте можна побачити що розрив стає меншим (рисунок 10).

Особливість JPEG в тому, що цей метод позволяє налаштовувати компроміс між якістю та стисненням. Тобто зі зменшенням SSIM буде збільшуватись ступінь стиснення, що можна побачити за результатами експериментів.

Модель автокодувальника з 5 шарами показує трохи кращу ступінь стиснення, ніж модель з 3 шарами (1,11 проти 1,18), проте у порівнянні з JPEG різниця незначуща. Графік розподілу ступеня стиснення представлений на рисунку 11.

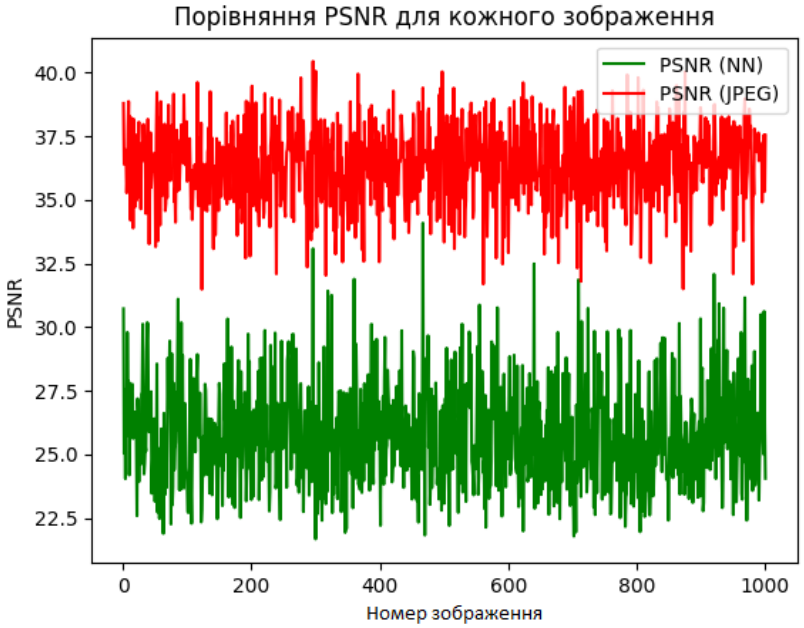

Рис. 10 – Розподіл PSNR для моделі з 5 шарами

p-ISSN: 2225-6733; e-ISSN: 2519-271X

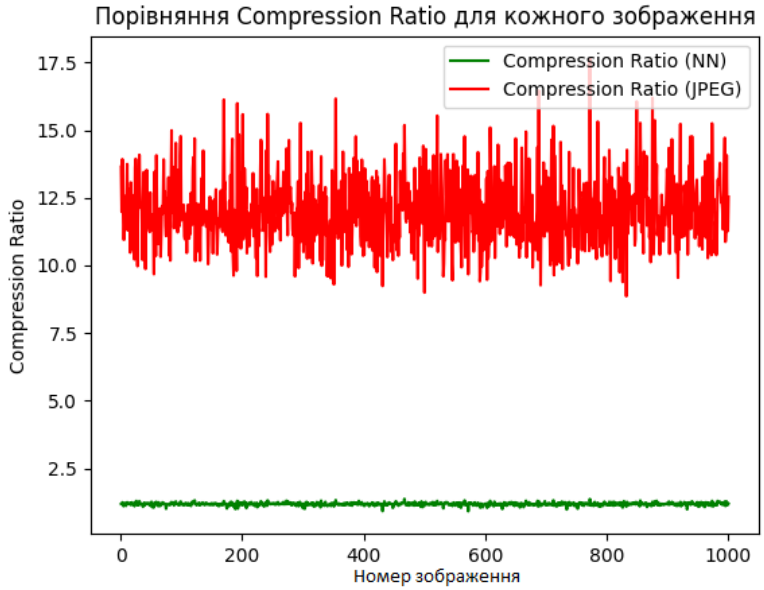

Рис. 11 – Графік Compression ratio для моделі з 5 шарами

Візуальне порівняння представлене на рисунках 12 та 13.

(а) (б) (в)

Рис. 12 – Порівняння деталей зображення для моделі з 5 шарами: оригінальне зображення (а); автокодувальник (б); JPEG (в)

(а) (б)

Рис. 13 – Порівняння деталей зображення для моделі з 5 шарами: модель (а); JPEG (б)

p-ISSN: 2225-6733; e-ISSN: 2519-271X

Зображення все ще дуже схожі на оригінал, проте вже можна побачити невелику різницю у кольорі.

У порівнянні з попередньою моделлю з 3 шарами, можна побачити невелике падіння таких показників, як SSIM та PSNR. При цьому наявний невеликий приріст ступеня стиснення. Отже, процес стиснення за допомогою моделі автокодувальника є стисненням з втратами, а ступінь стиснення залежить від якості вихідних зображень.

Розглянемо модель автокодувальника з 7 шарами (рисунок 14).

Рис. 14 – Результати роботи моделі з 7 шарами

SSIM 0,85 відповідає налаштуванням 20/100 для JPEG.

Можна побачити сильне падіння SSIM, що є першою ознакою перенавчання (overfitting) моделі. Вона має занадто багато параметрів, що дозволяє їй «запам’ятовувати» тренувальні дані замість того, щоб узагальнювати їх. У даному випадку вихідні зображення все ще мають чітку структуру, особливо при порівнянні з JPEG, де вже можна побачити «блокування» зображення. Проте на вихідних зображеннях майже повністю відсутній зелений колір (рисунок 15) та деякі дрібні деталі зображення починають розмиватись (рисунок 16).

(а) (б)

Рис. 15 – Результат роботи моделі з 7 шарами: модель (а); JPEG (б)

p-ISSN: 2225-6733; e-ISSN: 2519-271X

Рис. 16 – Розмиття дрібних деталей зображення при роботі моделі з 7 шарами

На основі цього можна сказати, що для зображень розміром 128x128 пікселів недоцільно використовувати модель, у якої більше 5 шарів згортання.

Роздивимось приклад дуже серйозного перенавчання на основі моделі з 10 шарами (рисунок 17). SSIM 0,80 приблизно дорівнює налаштуванням 13/100 для JPEG.

Рис. 17 – Результати роботи моделі з 10 шарами

Через сильне перенавчання модель генерує артефакти, як представлено на рисунках 18 та 19. Загалом, на вихідних зображеннях дуже часто можна побачити частини обличчя або навіть цілі обличчя там, де їх не повинно бути.

Рис. 18 – Результат роботи моделі з 10 шарами

p-ISSN: 2225-6733; e-ISSN: 2519-271X

Рис. 19 – Спотворене зображення в результаті стиснення при роботі моделі з 10 шарами

Підсумки проведеного дослідження демонструють, що на даному етапі нейронні мережі не можуть замінити класичні методи стиснення, і що ця тема потребує подальшого дослідження. На прикладі базового автокодувальника показано, що нейронні мережі здатні відтворювати зображення з достатньою якістю, однак наразі вони не досягають ефективності класичних методів у плані стиснення та затрат на час. Також було доведено, що занадто глибокі моделі гірше справляються з задачею. Отже, для стиснення однотипних зображень, розміром 128x128 пікселів, доцільно використовувати поверхневі (shallow) нейроні мережі з 2-3 шарами.

Основними викликами на даному етапі є недостатнє стиснення даних та дуже велика потреба у обчислювальних ресурсах. Перша проблема потребує подальшої оптимізації моделі та розробки функції втрат, яка б навчила модель компромісу між стисненням та якістю, тоді як друга потребує застосування сучасних методів компресії моделей, зокрема квантизації та видалення «мертвих» нейронів та ваг (pruning).

Висновки

Таким чином, було встановлено, що традиційні методи стиснення зображень демонструють високу ефективність, але при цьому генерують артефакти, особливо при високих рівнях стиснення, тоді як нейромережеві методи менше схильні до генерації артефактів.

У ході роботи було розроблено модель автокодувальника для стиснення зображень. Проведені експерименти засвідчили, що запропонована модель здатна зберігати ключові структурні елементи, але показала дуже низький ступінь стиснення у порівнянні з JPEG.

Експериментальним шляхом було виявлено максимальну допустиму кількість шарів згортання у моделі. Так, для зображень розміром 128x128 пікселів, максимально допустима кількість шарів згортання становить 5, оптимальним рішенням буде використовувати 2-3 шари згортання.

У результаті аналізу експериментальних даних було виявлено, що головною проблемою нейромережевого підходу є низький ступінь стиснення та висока ресурсомісткість. Подальша оптимізація архітектури моделі може підвищити ефективність стиснення. Для покращення результатів було запропоновано провести оптимізацію та компресію моделі. Проведені дослідження свідчать, що без зменшення потреби в обчислювальних ресурсах нейромережеві методи стиснення не зможуть замінити класичні методи.

Перелік використаних джерел:

1. Li X., Ji S. Neural Image Compression and Explanation. *IEEE Access*. 2020. Vol. 8. Pp. 214605-214615. DOI: https://doi.org/10.1109/ACCESS.2020.3041416.
2. Ballé J., Laparra V., Simoncelli E.P. End-to-end Optimized Image Compression. *ICLR 2017* : 5th International Conference on Learning Representations, Toulon, France, 24-26 April 2017. Pp. 1-27. DOI: https://doi.org/10.48550/arXiv.1611.01704.
3. Autoencoders and their applications in machine learning: a survey / K. Berahmand Fet al. *Artificial Intelligence Review*. 2024. Vol. 57(2). Pp. 1-52. DOI: https://doi.org/10.1007/s10462-023-10662-6.
4. Kingma D.P., Welling M. An introduction to variational autoencoders. *Foundations and trends in machine learning*. 2019. Vol. 12(4). Pp. 307-392. DOI: https://doi.org/10.48550/arXiv.1312.6114.
5. Generative Adversarial Nets / I. Goodfellow et al. *Proceedings of the International Conference on Neural Information Processing Systems*, Montreal, Canada, 8-13 December 2014. Vol. 3(11). Pp. 2672-2680. DOI: https://doi.org/10.1145/2969033.2969125.
6. High-Fidelity Generative Image Compression / Mentzer F., Toderici G., Tschannen M., Agustsson E. *NeurIPS 2020* : Proceedings of the 34th Conference on Neural Information Processing Systems, Vancouver, Canada, 6-12 December 2020. Pp. 11913-11924. DOI: https://doi.org/10.5555/3495724.3496723.
7. Bank D., Koenigstein N., Giryes R. Autoencoders. *Machine Learning for Data Science Handbook* / ed. by Rokach L., Maimon O., Shmueli E. Springer, Cham, 2023. Pp. 353-374. DOI: https://doi.org/10.1007/978-3-031-24628-9_16.
8. Image quality assessment: from error visibility to structural similarity / Wang Zh., Bovik A. C. , Sheikh H. R., Simoncelli E. P. *IEEE Transactions on Image Processing*. 2004. Vol.13. No. 4. Pp. 600-612. DOI: https://doi.org/10.1109/TIP.2003.819861.

References:

1. X. Li, and S. Ji, «Neural Image Compression and Explanation», *IEEE Access*, vol. 8, pp. 214605-214615, 2020. **doi: 10.1109/ACCESS.2020.3041416**.
2. J. Ballé, V. Laparra, and E.P. Simoncelli, «End-to-end Optimized Image Compression», in Proc. 5th Int. Conf. on Learning Representations (ICLR 2017), Toulon, France, 2017, pp. 1-27. **doi: 10.48550/arXiv.1611.01704**.
3. K. Berahmand, F. Daneshfar, E.S. Salehi, Yu.Li, and Yue Xu, «Autoencoders and their applications in machine learning: a survey», *Artificial Intelligence Review*, vol. 57(2), pp. 1-52, 2024. **doi: 10.1007/s10462-023-10662-6**.
4. D.P. Kingma, and M. Welling, «An introduction to variational autoencoders. foundations and trends in machine learning», *Foundations and trends in machine learning*, vol. 12(4), pp. 307-392, 2019. **doi: 10.48550/arXiv.1312.6114**.
5. I. Goodfellow et al., «Generative Adversarial Nets», in Proc. International Conference on Neural Information Processing Systems, Montreal, Canada, 2014, pp. 2672-2680. **doi: 10.1145/2969033.2969125**.
6. F. Mentzer, G. Toderici, M. Tschannen, and E. Agustsson, «High-Fidelity Generative Image Compression», in Proc. 34th Conf. on Neural Information Processing Systems, Vancouver, Canada, 2020, pp. 11913-11924. **doi: 10.5555/3495724.3496723**.
7. D. Bank, N. Koenigstein, and R. Giryes, «Autoencoders», in *Machine Learning for Data Science Handbook*, L. Rokach, O. Maimon, E. Shmueli, Eds. Springer, Cham, 2023. **doi: 10.1007/978-3-031-24628-9_16**.
8. Zh. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli «Image quality assessment: from error visibility to structural similarity», *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600-612, 2004. **doi: 10.1109/TIP.2003.819861**.

Стаття надійшла 10.09.2024
Стаття прийнята 29.09.2024