DOI 10.31891/2307-5732-2021-303-6-73-80
УДК 004.627

ПОРТЯНИЙ І. С.
НТУУ «Київський політехнічний інститут імені Ігоря Сікорського»
ORCID ID: 0000-0001-7477-3987
e-mail: iportianoy@gmail.com
ПОСПЄЛОВА К. І.
НТУУ «Київський політехнічний інститут імені Ігоря Сікорського»
ORCID ID: 0000-0001-7374-2812
ОЛІЙНИК Ю. О.
НТУУ «Київський політехнічний інститут імені Ігоря Сікорського»
ORCID ID: 0000-0002-7408-4927

КОДУВАННЯ РАСТРОВИХ ЗОБРАЖЕНЬ НА ОСНОВІ ПОДІБНОСТІ ФРАГМЕНТІВ

Робота присвячена кодуванню зображень на основі визначення подібності фрагментів шляхом використання нейронних мереж для виділення ознак фрагментів та алгоритмів машинного навчання для пошуку подібних фрагментів. У роботі використано згорткові нейронні мережі, а також класифікатор KNN (k-Nearest Neighbors) для кодування зображень, проведено порівняння розміру закодованого зображення з вхідним. Для того, щоб кодувати зображення для початку потрібно заповнити сховище даних ознаками фрагментів схожих зображень, після чого по кожному фрагменту отриманих зображень потрібно виділити ознаки та записати в сховище даних. Після того, як сформовано базу ознак фрагментів, можна виконувати кодування нових зображень за допомогою збережених фрагментів.

Ключові слова: кодування зображень, згорткові нейронні мережі, стиснення даних, декодування зображень.

IVAN PORTIANYI, KAROLINA POSPIELOVA, YURII OLIINYK
NTUU «Igor Sikorsky Kyiv Polytechnic Institute»

ENCODING RASTER IMAGES BASED ON FRAGMENT SIMILARITY

This paper is devoted to image encoding based on determining the similarity of fragments by using neural networks to extract the features of fragments and machine learning algorithms to find similar fragments. In the modern world, the problem of image storage is quite relevant. Graphic data takes up quite a lot of disk space, while Internet users upload more and more pictures. Also, every year there is a development of photography and image quality is improving, respectively, and the size of graphic data is growing. Data warehouses of social networks, messengers, file sharers and other Internet resources are filled with tens of thousands of new pictures every day. Therefore, the question arises about reducing the size of graphic data. In general, it should be noted that one of the most important and defining aspects of both storage and transmission of information is its compression. The problem described above is solved by encoding and compressing images. With the help of coding, the size of graphic information is reduced, which saves storage space and, accordingly, the money spent. In view of this, it is important to develop a method and means of image coding. Many methods exist for compressing graphic information. For example, jpeg, webp, png and others. These methods usually use the removal of redundant information in the photo and work purely with the image itself, but none of the methods uses fragments of similar images. The article uses convolutional neural networks and KNN (k-Nearest Neighbors) classifier for image encoding. and compares the size of the encoded image with the input. In order to encode the image, you first need to fill the data warehouse with features of fragments of similar images, then for each fragment of the obtained images you need to select the features and write to the data warehouse. Once the snippet feature database is formed, you can encode new images using saved snippets.

Keywords: image encoding; convolutional neural networks; data compression; image decoding.

Постановка проблеми

Сучасний світ постійно змінюється, у наш час відбувається стрімкий розвиток технологій та застосування їх у будь-яких сферах життя. За останні роки значних успіхів було досягнуто в сфері комп'ютерного зору та, відповідно, нейронних мереж, що працюють з зображеннями. Розвиток цих сфер розширює можливості для роботи з картинками.

Достатньо актуальною є проблема зберігання зображень. Адже графічні дані займають достатньо велику кількість дискового простору, при цьому користувачі інтернету завантажують все більше й більше картинок. Також з кожним роком відбувається розвиток фототехніки й якість зображень покращується, відповідно й розмір графічних даних зростає. Сховища даних соціальних мереж, месенджерів, файлообмінників та інших інтернет ресурсів кожного дня наповнюються десятками тисяч нових картинок. Тому постає питання щодо зменшення розміру графічних даних. Слід зазначити, що одним з найбільш важливих і визначальних аспектів як для зберігання, так і для передачі є стиснення вихідної інформації. Графічні дані займають достатньо велику кількість дискового простору, тому актуальною є задача кодування та стиснення зображень.

Головною ідеєю даної роботи є розробка методу та засобів для кодування зображень на основі визначення подібних фрагментів інших зображень. Розроблюваний інструмент дозволить зменшити розмір графічного файлу за допомогою кодування.

Аналіз останніх досліджень і публікацій

Алгоритми кодування та стиснення зображень можна розділити на два класи: стиснення без втрат і стиснення з втратами. У першому випадку після компресії інформація про зображення зберігається в повному обсязі, а в другому – частково втрачається.

Під час стиснення без втрат стиснуте зображення є таким самим як і оригінальне, після декодування якість зображення не погіршується. Прикладами методів стиснення без втрат є: метод Хафмана, групове стиснення (RLE – Run Length Encoding), метод LZW.

Методи стиснення з втратами, як зрозуміло з назви, призводять до невеликої втрати інформації й погіршення якості зображення. Прикладами методів стиснення з втратами є: JPEG, стиснення на основі вейвлет-перетворень, фрактальне стиснення [1].

Метод LZW. Цей метод винайдено у 1978 році [2]. Він отримав назву на честь його трьох розробників Lempel, Ziv, Welch. Алгоритм кодує будь-які сигнали. Він подібний до метода Хафмана, але на відміну від нього, для кодування сигналів використовуються коди однакових довжин, також додатково використовуються коди для послідовностей елементів, що часто зустрічаються.

Спочатку послідовно зчитуються символи вхідного потоку послідовності та відбувається перевірка на існування у новоствореній таблиці рядків такого рядка. Якщо умова дійсна, то зчитується наступний сигнал, інакше – в потік заноситься код попередньо знайденого рядка, цей рядок заноситься в таблицю й пошук відбувається знову.

Для кодування зображення спочатку створюється таблиця кольорів, що наявні у стиснутому зображенні. Це потрібно для того, щоб замість значення кольору пікселя використовувати індекс з таблиці. Аналогічно алгоритму Хафмана, найбільш часто зустрічають кольори з найменшими індексами, а рідші – розміщуються в кінці таблиці. Ця палітра кольорів з індексами повинна розміщуватись між заголовком та зображенням.

Алгоритм JPEG. JPEG – це зображення з досить дрібним регулярним малюнком. Характер JPEG спотворень, що вносяться алгоритмом такий, що зменшення або збільшення зображення може дати неприємні ефекти [3].

Практично він є стандартом де-факто для повнокольорових зображень. Оперує алгоритм областями 8x8, на яких яскравість і колір змінюються порівняно плавно. Внаслідок цього, при розкладанні матриці такої області в подвійний ряд косинусів значущими виявляються тільки перші коефіцієнти. Таким чином, стиснення в JPEG здійснюється за рахунок плавності зміни кольорів у зображенні.

Метод фрактального стиснення. Новий алгоритм фрактального стиснення [4] було запропоновано Майклом Барнслі та Аланом Слоуном на основі системи доменних та рангових блоків зображення, а також квадратних блоків, що покривали все зображення. Цей метод є одним з найкращих, бо має досить хороше співвідношення між якістю відновленим зображенням і глибиною стиснення.

Ідея методу стиснення заснована на пошуку подібних ділянок зображення, які дуже близькі між собою за розподілом інтенсивності пікселів та розрахунку коефіцієнтів. Результатом такого алгоритму є набір параметрів системи ітерованих функцій (Iterated Function System).

Перевагами методу є те, що в порівнянні з іншими – цей алгоритм позбавляє таких недоліків, як втрата важливих деталей зображення. На сьогодні є багато удосконалень та оптимізацій методу.

Недоліком є складність, бо досить довго відбувається процес пошуку доменних блоків, адже потрібний повний перебір (потрібно порівняти два масиви).

Проаналізувавши існуючі методи, враховуючи сучасні дослідження, не вдалося знайти подібного методу, до того, що розглядається в даній статті.

Цілі та задачі дослідження

Предметною областю даної роботи є будь-яка система, яка працює із великою кількістю зображень й має потребу в зменшенні їх розміру для зберігання.

Метою роботи є зменшення об’єму даних для збереження зображень за рахунок використання методу кодування на основі наявної бази даних сегментів схожих зображень.

Для цього необхідно мати базу картинок, які потрібно розбити на фрагменти і з кожного фрагменту за допомогою нейронної мережі дістати їх ознаки, після чого записати отримані ознаки в сховище даних. Коли система отримуватиме нове зображення для кодування, то його також потрібно розділити на фрагменти й дістати з них ознаки. Маючи ознаки вхідного зображення за допомогою алгоритму машинного навчання к-найближчих сусідів потрібно визначити набір фрагментів, якими потім закодувати картинку.

На вхід програмі подається файл зображення з розширенням *.bmp, *.jpg, *.jpeg, *.png.

Вихідними даними є набір байтів. Цей набір являється стиснутим масивом ідентифікаторів фрагментів з бази, якими закодовано вхідне зображення. Також програма дозволяє переглянути декодоване зображення.

Програмне забезпечення повинно реалізовувати такі функції:

- зчитування та обробка вхідного зображення;
- розділення зображення на фрагменти;
- відновлення суцільного зображення з набору фрагментів;
- виділення ознак з фрагменту;
- пошук подібного фрагменту;
- стиснення набору ідентифікаторів фрагментів;
- кодування зображення;
- декодування зображення;
- зчитування та запис даних про фрагменти в базу даних.

Обрані засоби та технології

Згорткові нейронні мережі. Згорткова нейронна мережа за рахунок застосування спеціальної операції – згортки – дозволяє зменшити кількість інформації в пам'яті (зменшити розмірність даних) й за рахунок цього краще справляється з картинками більш високого розширення [5].

По суті кожен шар нейронної мережі використовує власне перетворення. Якщо на перших шарах мережа оперує такими поняттями, як «ребра», «межі», то далі використовуються поняття «текстури», «частини об'єктів». У результаті такої обробки ми можемо правильно класифікувати картинку або отримати на кінцевому кроці певний об'єкт на фото.

Типова архітектура згорткової нейронної мережі містить наступні базові шари: Convolutional Layer, Pooling Layer, ReLU(Rectified Linear Unit) Layer та Fully Connected Layer.

Згортковий шар (Convolutional layer) є одним з основних шарів згорткової мережі, який виконує основну масу обчислювальної роботи. Головне призначення цього шару – це витягнути ознаки з вхідних даних, у нашому випадку вхідними даними є картинки. За допомогою згортки зберігаються просторові пропорції між пікселями та визначаються особливості зображення використовуючи маленькі квадрати вхідного зображення.

Pooling layer зменшує розмірність кожної активаційної карти, але зберігає найбільш важливу інформацію. Цей шар забезпечує найкраще узагальнення та найшвидшу збіжність. Він стійкий до спотворень та зазвичай розміщується між згортковими шарами.

ReLU являється нелінійною операцією – це поелементна операція, що означає, що вона застосовується до кожного елементу активаційної матриці. Застосування ReLU замінює всі від'ємні значення в активаційній карті на нуль.

Останнім шаром є Fully Connected Layer (FCL). Термін Fully Connected Layer означає, що кожен фільтр на попередньому рівні пов'язаний з кожним фільтром на наступному рівні. Цей шар отримує оброблені попередніми шарами ознаки картинки, та на основі отриманих даних класифікує вхідне зображення по різних класах на основі навчального набору даних.

***KNN* (*k-Nearest Neighbours*).** Цей метод визначає поділяючі межі локально. У варіанті 1NN кожна ознака відноситься до визначеного класу в залежності від інформації про його найближчого сусіда. У варіанті kNN кожна ознака відноситься до переважного класу найближчих сусідів, де k – параметр методу. В основі методу kNN лежить факт, що відповідно до гіпотези компактності ми очікуємо, що тестова ознака d буде мати таку ж мітку, як і навчальні ознаки в локальній області, що оточує ознаку d.

Важливо правильно вибрати параметр k. З одного боку, k має бути достатньо великий, щоб серед найближчих сусідів виявилися представники різних класів, з іншого – достатньо малий, щоб не стерти нюанси межі, що розділяє класи. До переваг методу k найближчих сусідів слід віднести простоту реалізації, легку інтерпретацію результатів, а також стійкість до шуму порівняно з методом найближчого сусіда. Їх недоліки подібні: необхідність зберігати всю навчальну вибірку; значні витрати на пошук найближчих сусідів; мінімальні можливості з настроювання алгоритму, оскільки має місце лише один параметр k [6].

TensorFlow. TensorFlow – відкрита програмна бібліотека для машинного навчання, розроблена компанією Google для вирішення завдань побудови і тренування нейронної мережі з метою автоматичного знаходження та класифікації образів, досягаючи якості людського сприйняття [7].

Кожне обчислення в TensorFlow представляється як граф потоку даних. У нього є два елементи: який представляє одиниці обчислень та той, що представляє одиниці даних.

TensorFlow дає змогу випускати ML-орієнтовані рішення на пристрої користувачів дуже швидко та з невеликим двійковим розміром, але має підтримку обмеженого набору систем.

Сильні сторони:

1. Найпопулярніший інструмент DL, з відкритим кодом, швидким залученням, який добре підтримується сильною промисловою компанією.
2. Потужна чисельна бібліотека для програмування потоків даних, яка є основою для досліджень та розробок DL.
3. Ефективно працює з математичними виразами, що включають багатовимірні масиви.
4. Дуже добре задокументована.
5. Обчислення на GPU/CPU, мобільні обчислення, висока масштабованість обчислень на різних машинах та величезні набори даних.

Індекс структурної подібності SSIM. Індекс структурної подібності є одним з методів вимірювання схожості двох зображень. SSIM-індекс це метод повного зіставлення, іншими словами, він проводить вимірювання якості на основі вихідного зображення (не стисло, без спотворень). SSIM-індекс є розвитком традиційних методів, таких як PSNR (peak signal-to-noise ratio) і метод середньоквадратичної помилки MSE, які виявилися несумісні з фізіологією людського сприйняття.

Відмінною особливістю методу є те, що метод враховує "сприйняття помилки", завдяки врахуванню структурних змін інформації. Ідея полягає в тому що пікселі мають сильний взаємозв'язок, особливо коли вони близькі просторово. Дані залежності несуть важливу інформацію про структуру об'єктів і про зображення в цілому [8].

LZMA. Для стиснення набору ідентифікаторів подібних фрагментів було обрано алгоритм стиснення даних без втрат LZMA (Lempel-Ziv-Markov chain-Algorithm). LZMA використовує алгоритм словникового стиснення, вихідні дані якого закодовані інтервальним кодуванням, що використовує складну модель обчислення ймовірності появи кожного біта. Система стиснення знаходить відповідності, використовуючи

словникову структуру даних, і створює потік символів і посилань фраз, які вже наявні у словнику, який закодовано 1 бітом інтервальним кодувальником [9].

Опис методу. Метод кодування Frasim (скорочення від fragment similarity) зображень на основі подібності фрагментів містить в собі два основних процеси:

1. Процес наповнення бази даних фрагментів.

2. Процес кодування зображення.

Схеми для обох процесів наведено на рис. 1 та 2.

Рис. 1. Схема процесу наповнення бази даних фрагментів

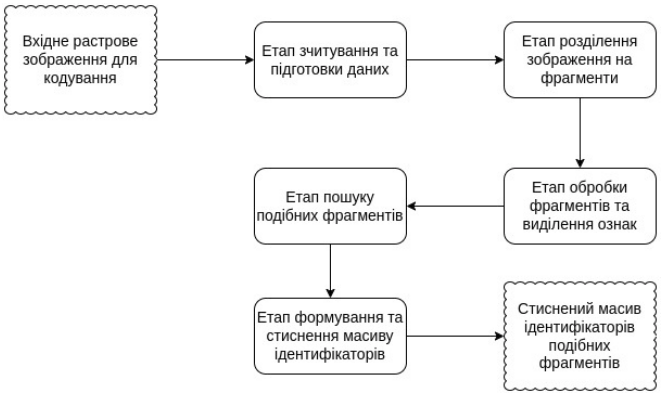

Рис. 2. Схема процесу кодування зображення

Кожен з процесів складається з окремих етапів. Деякі етапи будуть унікальними для процесу, а інші будуть зустрічатися в обох. Кодування зображення буде доступним тільки після наповнення бази даних фрагментів зображень.

Етап зчитування та підготовки даних

На цьому етапі отримується вхідне зображення, яке при зчитуванні перетворюється у масив, кожним елементом якого є цифрове представлення пікселя у форматі RGB. Далі, за потреби, виконується зміна розширення (висоти та/або ширини) зображення.

Етап розділення зображення на фрагменти

Під час цього етапу зображення розділяється на фрагменти. Для початку визначаються розміри фрагментів і, в залежності від цього, їх кількість. Після чого масив пікселів, отриманий при зчитуванні зображення, розділяється на задану кількість фрагментів відповідного розміру.

Під час процесу наповнення бази вхідне зображення розділяється на фрагменти декілька разів. Кожного разу задаються різні розміри фрагментів, від відносно невеликих (3×4, 5×6) до більших (16×24, 24×32). Це дозволяє отримати більшу кількість різноманітних фрагментів з одного зображення, що при кодуванні дасть можливість вибирати бажану деталізацію.

Під час процесу кодування зображення на цьому етапі є можливість обрати розміри та число фрагментів. Чим менший розмір фрагментів та більша їх кількість, тим кращою буде якість декодованого зображення, але й водночас більший розмір вихідних закодованих даних.

Етап обробки фрагментів та виділення ознак

На цьому етапі відбувається підготовка отриманих фрагментів, так як виділення ознак виконується за допомогою нейронної мережі, яка потребує відповідний формат вхідних даних. Кожен з фрагментів перетворюється в тензор, потім змінюється розмір тензора на 224×224×3 і дані конвертуються в формат float32. Після цього підготовлені дані фрагменту подаються на вхід нейронній мережі, яка виділяє ознаки фрагменту й на виході повертає масив чисел-ознак.

Етап збереження фрагментів у базу

На цьому етапі ми маємо набір фрагментів, їх характеристики та виділені ознаки. Ці дані зберігаються в базу даних. Запис містить наступний формат:

- ідентифікатор фрагменту в базі даних у цілочисельному форматі;
- фрагмент у вигляді масиву пікселів, які представлені у форматі RGB;
- розмір фрагменту;
- масив виділених ознак.

За допомогою ідентифікатору буде відбуватись кодування нових зображень. Сам фрагмент потрібно зберігати для того, щоб у подальшому декодувати зображення. Розмір знадобиться при фільтрації й відкиданні зайвих фрагментів. А за допомогою масиву виділених ознак буде відбуватись порівняння й визначення рівня подібності.

Етап пошуку подібних фрагментів

Цей етап є одним з головних в розробленому методі. Передбачається, що при переході на даний етап база даних фрагментів заповнена, а вхідне зображення розділене на фрагменти, з яких отримано ознаки.

Отже, вхідними даними етапу є масив з наборами ознак кожного фрагменту, отриманого із зображення, яке потрібно закодувати, та самі фрагменти.

На цьому етапі для фрагментів вхідного зображення знаходиться найбільш схожий фрагмент з бази, шляхом порівняння ознак. Спочатку визначається розмір фрагменту, після чого з бази вибираються всі

фрагменти з таким же розміром. Далі для кожного вхідного фрагменту виконується пошук найбільш схожого й зберігається його ідентифікатор.

Схожість фрагментів визначається за допомогою порівняння ознак. У нашому випадку між наборами ознак знаходиться евклідова відстань й фрагмент з найменшим значенням відстані вважається найбільш схожим.

На виході з даного етапу отримується масив з ідентифікаторами схожих зображень.

Етап формування та стиснення масиву ідентифікаторів

Зазвичай розмір даних масиву ідентифікаторів уже буде набагато меншим за розмір даних вхідного зображення. Але на останньому етапі масив також стискається для того, щоб ще більше зменшити розмір вихідних даних закодованого зображення. Існує багато методів для стиснення масиву цілих чисел, вони будуть описані в наступному розділі дисертації.

Кінцевим результатом даного етапу є набір байтів – стиснений методом lzma (Lempel-Ziv-Markov chain-Algorithm) масив ідентифікаторів подібних фрагментів.

Експерименти та результати

Для проведення експериментів було обрано набір зображень квітів 102 Flowers Diff Species DataSet [10]. Цей набір містить зображення квітів, що відносяться до 102 різних категорій. Зображення були отримані шляхом пошуку в Інтернеті і фотографування. Для кожної категорії є мінімум 40 зображень з різних ракурсів та з різним освітленням. Дані узято з ресурсу Kaggle [10].

Перед тим, як почати дослідження ефективності, було сформовано базу ознак фрагментів з картинок квітів однієї категорії. На рис. 3 показано декілька картинок з набору даних.

Рис. 3. Приклади зображень

Залежність якості та часу кодування від кількості фрагментів

На рис. 4 показано вхідне зображення для кодування. Розмір вхідного зображення 180 128 байт.

Рис. 4. Вхідне зображення для кодування

У роботі було проведено дослідження залежності якості декодованого зображення та часу кодування від кількості фрагментів, серед яких виконується пошук подібних. У таблиці 1 наведено результати дослідження.

Таблиця 1

Результати експериментів

Кількість фрагментів	Час кодування, секунд	Вихідний розмір, байт	Відсоток стиснення, %	Подібність, SSIM
20 000	15,55	3733	97,93	0,84
50 000	27,56	4069	97,74	0,86
100 000	44,67	4677	97,40	0,88
150 000	63,36	4905	97,28	0,89
200 000	82,83	5005	97,22	0,89

З результатів експерименту бачимо, що зі збільшенням кількості фрагментів у базі час кодування стрімко зростає, відсоток стиснення незначно зменшується, а подібність зображення поступово збільшується. Варто зазначити, що після 150 000 фрагментів значення відсотку стиснення та подібності майже не змінюються, а от час кодування зростає.

Отже, при збільшенні кількості фрагментів у базі даних покращується якість декодованого зображення та в незначній мірі погіршується відсоток стиснення. Збільшення кількості фрагментів є ефективним до певного числа, після якого всі метрики не сильно змінюються, а час роботи методу кодування зростає.

На рис. 5 можемо побачити результат декодування вхідного зображення при кількості фрагментів 200 000.

Рис. 5. Декодоване вхідне зображення

Порівняння розробленого методу з JPG і JPEG. Виконаємо порівняння результатів розробленого методу з найбільш популярними методами стиснення зображень JPG та JPEG. У таблиці 2 наведено результати порівняння. Вхідне зображення у форматі PNG показано на рис. 6. Зображення має розмір 300×300 пікселі, а розмір даних 175 248 байт.

Рис. 6. Вхідне зображення у форматі PNG

Таблиця 2

Порівняння з JPG і JPEG

Метод	Вихідний розмір, байт	Відсоток стиснення, %	Подібність, SSIM
Розроблений метод Frasim	11 613	93,37	0,85
JPG	40 890	76,67	0,98
JPEG	41 832	76,13	0,98

На рис. 7 наведено зображення, яке закодоване за допомогою методу, який запропоновано в даній роботі.

Рис. 7. Зображення, закодоване запропонованим методом Frasim

В результаті експериментів ми дійшли висновку, що розроблений метод має високий коефіцієнт подібності, хоча він і є меншим ніж у JPG та JPEG. Але запропонований у даній роботі метод має більший відсоток стиснення. Абсолютне значення розміру декодованого зображення є в 3.5 рази меншим за вихідний розмір даних JPG та в 3.6 рази меншим за вихідний розмір даних JPEG.

Висновки

Дослідження присвячене пошуку нових шляхів зменшення розміру зображень для їх збереження, зокрема за допомогою методів кодування. Було запропоновано новий метод для кодування растрових зображень на основі подібності фрагментів. Розроблений метод працює за наявності бази фрагментів схожих зображень, з яких було виділено ознаки за допомогою нейронної мережі.

Наведено детальний опис підготовки зображення до кодування, процес розбиття на фрагменти і виділення їх ознак, а також пошук подібних фрагментів.

Було проведено ряд експериментів для перевірки ефективності методу та порівняння з популярними методами кодування JPG і JPEG. В результаті експериментів було визначено залежність якості та розміру декодованого зображення від кількості наявних фрагментів у базі даних. У порівнянні з стандартами JPG і JPEG розроблений метод стискає зображення у більше ніж 3 рази більше при незначному зменшенні метрики SSIM.

Література

1. Various Image Compression Techniques: Lossy and Lossless. 2016. URL: https://www./ researchgate.net/publication/303319054_Various_Image_Compression_Techniques _Lossy_and_Lossless.

2. LZW Data Compression. American Journal of Engineering Research (AJER). 2014. URL: https://www.ajer.org/papers/v3(2)/C0322226.pdf.

3. Milano J. Compressed Image File Formats: JPEG, PNG, GIF, XBM, BMP. 1999. 288 p.

4. Ситник А. Ю. Обробка графічних зображень засобами фрактальної геометрії [Електронний ресурс] / Аким Юрійович Ситник. – 2021. – Режим доступу : https://ela.kpi.ua/bitstream/123456789/43996/1/ Sytnyk_bacalavr.pdf.

5. Rohan T. Convolutional Networks for everyone. 2018. URL: https://medium.com/@rohanthomas.me/ convolutional-networks-for-everyone-1d0699de1a9d

6. Antony C. K-Nearest Neighbor. 2021. URL: https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4.

7. Erdem Isbilen. Image Similarity Detection in Action with Tensorflow 2.0. 2019. URL: https://towardsdatascience.com/image-similarity-detection-in- action-with-tensorflow-2-0-b8d9a78b2509.

8. Datta P. All about Structural Similarity Index (SSIM). 2020. URL: https://medium.com/srm-mic/all-about-structural-similarity-index-ssim-theory-code- in-pytorch-6551b455541e.

9. Heinz M. All The Ways to Compress and Archive Files in Python. 2021. URL: https://towardsdatascience.com/all-the-ways-to-compress-and-archive-files-in- python-e8076ccedb4b.

10. 102 Flowers Diff Species DataSet. 2016. URL: https://www.kaggle.com/lenine/flower-102diffspecies-dataset.

11. Burger W. Principles of Digital Image Processing: Advanced Methods (Undergraduate Topics in Computer Science. London: Springer, 2013. 382 p.

12. Thakar V. Deep Learning with Python and OpenCV: A beginner's guide to perform smart image processing techniques using TensorFlow and Keras.

References

1. Various Image Compression Techniques: Lossy and Lossless. 2016. URL: https://www.researchgate.net/publication/ 303319054_Various_Image_Compression_Techniques _Lossy_and_Lossless.

2. LZW Data Compression. American Journal of Engineering Research (AJER). 2014. URL: https://www.ajer.org/papers/v3(2)/C0322226.pdf.

3. Milano J. Compressed Image File Formats: JPEG, PNG, GIF, XBM, BMP. 1999. 288 p.

4. Sytnyk A. Yu. Obrobka hrafichnykh zobrazhen zasobamy fraktalnoi heometrii [Elektronnyi resurs] / Akym Yuriiovych Sytnyk. – 2021. – Rezhym dostupu : https://ela.kpi.ua/bitstream/123456789/43996/1/ Sytnyk_bacalavr.pdf.

5. Rohan T. Convolutional Networks for everyone. 2018. URL: https://medium.com/@rohanthomas.me/convolutional-networks-for-everyone-1d0699de1a9d

6. Antony C. K-Nearest Neighbor. 2021. URL: https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4.

7. Erdem Isbilen. Image Similarity Detection in Action with Tensorflow 2.0. 2019. URL: https://towardsdatascience.com/image-similarity-detection-in- action-with-tensorflow-2-0-b8d9a78b2509.

8. Datta P. All about Structural Similarity Index (SSIM). 2020. URL: https://medium.com/srm-mic/all-about-structural-similarity-index-ssim-theory-code- in-pytorch-6551b455541e.

9. Heinz M. All The Ways to Compress and Archive Files in Python. 2021. URL: https://towardsdatascience.com/all-the-ways-to-compress-and-archive-files-in- python-e8076ccedb4b.

10. 102 Flowers Diff Species DataSet. 2016. URL: https://www.kaggle.com/lenine/flower-102diffspecies-dataset.

11. Burger W. Principles of Digital Image Processing: Advanced Methods (Undergraduate Topics in Computer Science. London: Springer, 2013. 382 p.

12. Thakar V. Deep Learning with Python and OpenCV: A beginners guide to perform smart image processing techniques using TensorFlow and Keras.

Рецензія/Peer review : 20.11.2021 Надрукована/Printed :30.12.2021