Міністерство освіти і науки України
Західноукраїнський національний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерної інженерії

Кухарук Віталій Романович

«Алгоритми генерування імуногістохімічних зображень / Algorithms for immunohistochemical images generating»

Студент групи КІм – 21
Кухарук Віталій Романович
_____

Науковий керівник
д.т.н., проф. О.М. Березький
_____

Тернопіль – 2020

РЕЗЮМЕ

Кваліфікаційна робота на тему «Алгоритми генерування імуногістохімічних зображень» зі спеціальності 123 «Комп’ютерна інженерія» написана обсягом 97 сторінок і містить 31 рисунок, 4 таблиці, 2 додатки та 62 джерела за переліком посилань.

Метою роботи є розробка алгоритмів генерування імуногістохімічних зображень і створення на цій основі генератора імуногістохімічних зображень.

Методи досліджень. Для розв’язання поставлених задач у випускній кваліфікаційній роботі використано методи: теорію штучних нейронних мереж, теорію ймовірностей, теорію алгоритмів, об’єктно-орієнтоване програмування.

Результати дослідження: на основі запропонованих алгоритмів розроблено генератор імуногістохімічних зображень, який реалізовано в на мові програмування Python 3.6.

Орієнтовні напрямки розвитку досліджень: у подальшому розвитку даної програмної системи передбачено розглянути питання нормалізації зображень з іншими геометричними перетвореннями, а також збільшення кількості зображень для навчання генеративно-змагальної мережі зі 100 до 10000.

КЛЮЧОВІ СЛОВА: НЕЙРОННІ МЕРЕЖІ, ГЕНЕРАТИВНО-ЗМАГАЛЬНІ МЕРЕЖІ, ГЕНЕРУВАННЯ, ІМУНОГІСТОХІМІЧНЕ ЗОБРАЖЕННЯ, ШТУЧНИЙ ІНТЕЛЕКТ.

RESUME

The qualification graduation thesis on «Algorithms for immunohistochemical images generating» from the specialty 123 «Computer engineering» is 97 pages long and contains 31 illustrations, 4 tables, 2 appendices and 62 references.

The aim of the work is to develop algorithms for generating immunohistochemical images and create an immunohistochemical images generator on this basis.

Research methods. To solve tasks in the final qualification work, the following methods were used: the theory of artificial neural networks, probability theory, the theory of algorithms, object-oriented programming.

Research results: on the basis of the proposed algorithms, a generator of immunohistochemical images was developed, which is implemented in the Python 3.6 programming language.

Directions of research development: in the further development of this software system, it is envisaged to consider the issue of image normalization with other geometric transformations, as well as an increase in the number of images for training a generative adversarial network from 100 to 10,000.

KEY WORDS: NEURAL NETWORKS, GENERATIVE ADVERSARIAL NETWORK, GENERATION, IMMUNOHISTOCHEMISTRY IMAGES, ARTIFICIAL INTELLIGENCE.

ЗМІСТ

ВСТУП

Актуальність роботи. У наш час у штучному інтелекті зростає зацікавленість вирішення більш складних задач генерування зображень. Тому удосконалення реалізації генерування зображень комп’ютерними системами є актуальною. Перспективним напрямком вирішення даної проблеми є застосування штучних нейронних мереж і нейрокомп'ютерів. В останні роки запропонована велика кількість архітектур нейромеж для застосування у генеруванні зображень. Перспективу в удосконаленні архітектур вбачають у генеративно-змагальних нейронних мережах (Generative adversarial network, GAN) [1]. Це алгоритм машинного навчання, який побудований на комбінації двох нейронних мереж – генеративної і дискримінативної. Перша – генерує зразки, друга – намагається відрізнити справжні зразки від згенерованих.

Генеративні моделі застосовуються для створення контенту і даних. Завдяки роботі генеративно-змагальних нейронних мереж і генеративних моделей виникає синтез даних, на яких потім будуть навчатися інші системи. Генеративні моделі застосовуються і при опрацюванні біомедичних зображень.

Питанням опрацювання біомедичних зображень присвячені наукові праці викладачів кафедри комп’ютерної інженерії Західноукраїнського національного університету. В роботах [2-4] описані інформаційно-аналітичні системи аналізу гістологічних та цитологічних зображень, нечіткі системи діагностики в роботах [5-6]. Інтелектуальним системам автоматизованої мікроскопії аналізу гістологічних та цитологічних зображень присвячені роботи [7-9]. База даних цифрових гістологічних та цитологічних зображень різних форм раку молочної залози «CIFDB» описана в роботі [10]. Синтез біомедичних зображень на основі генеративно-змагальних мереж присвячена робота [11].

Мета і завдання дослідження. У цій випускній кваліфікаційній роботі буде розглянуто методи та алгоритми генерування зображення на основі нейронних мереж. Метою випускної кваліфікаційної роботи є розробка алгоритмів генерування імуногістохімічних зображень.

У відповідності із поставленою метою випускна кваліфікаційна робота включає розв’язки таких задач:

проведення аналізу області застосування нейронних мереж – імуногістохімії та імуногістохімічних зображень;

аналіз методів та алгоритмів генерування зображень;

аналіз програмних засобів генерування зображень;

розробка алгоритмів та методів роботи генератора мережі;

розробка алгоритмів та методів роботи дискримінатора мережі;

розробка алгоритмів та методів роботи генеративно-змагальної мережі;

проектування та програмна реалізація генератора імуногістохімічних зображень на основі генеративно-змагальної мережі;

тестування розробленого програмного забезпечення.

Об’єктом досліджень є імуногістохімічні зображення.

Предметом досліджень є алгоритми генерування імуногістохімічних зображень.

Методи досліджень. У кваліфікаційній роботі використано теорію ймовірностей і математичну статистку, теорію алгоритмів, об’єктно-орієнтоване програмування.

Наукова новизна одержаних результатів полягає в розробці алгоритмів генерування імуногістохімічних зображень на основі генеративно-змагальної мережі.

Практичне значення одержаних результатів полягає в розробці програмної системи генерування імуногістохімічних зображень на основі генеративно-змагальної мережі.

Публікації результатів досліджень. За результатами досліджень опубліковані двоє тез доповідей ІІІ науково-практичної конференції молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі» (26 листопада 2020 р., м. Тернопіль, Західноукраїнський національний університет) [12-14]:

Гарматюк В.Р. Кухарук В.Р. Алгоритми оптимізації структур згорткових нейронних мереж: ІІІ Наук.-практ. конф. молодих вчених і студентів

«Інтелектуальні комп’ютерні системи та мережі». 26 листопада 2020 р. Тернопіль, 2020. С. 49.

Кухарук В.Р., Гарматюк В. Р. Алгоритми генерування зображень на основі нейронних мереж: III Наук.-практ. конф. молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі». 26 листопада 2020 р. Тернопіль, 2020. С. 50.

Випускна кваліфікаційна робота складається із трьох розділів, висновків, списку використаної літератури та додатків [15].

У першому розділі здійснено огляд області імуногістохімії та імуногістохімічних зображень, проаналізовано методи та алгоритми генерування зображень, описано програмні засоби генерування зображень, здійснено постановку задач на кваліфікаційну роботу.

У другому розділі здійснено аналіз алгоритмів і методів роботи генератора мережі: метод зворотного перетворення, вибірка з відхиленням, алгоритм Метрополіса — Гастингса; алгоритмів і методів роботи дискримінатора мережі як згорткової мережі; алгоритмів і методів роботи генеративно-змагальної мережі.

У третьому розділі спроектована і реалізована програмна система.

У додатках приведено світлокопії виданих публікацій та код програми.

1 АНАЛІЗ МЕТОДІВ ТА АЛГОРИТМІВ ГЕНЕРУВАННЯ ІМУНОГІСТОХІМІЧНИХ ЗОБРАЖЕНЬ

1.1 Аналіз імуногістохімічних зображень

Імуногістохімія (ІГХ) – це аналітичний метод визначення протеїнів (антигенів) у клітинах біологічних тканин на основі реакції антиген-антитіло [16, 17]. Заснували імуногістохімічний метод дослідники, які працювали під керівництвом Альберта Кунса (Albert Coons). Група вчених в 1941 р. вперше отримала мічені флюоресцеїном антитіла та використала їх на практиці [18]. У 1975 році дослідники Кохлер і Мілстейн розробили гібридомну технологію, яка стала кардинальною подією в імуногістохімії. Дослідження дозволило отримувати високоспецифічні антитіла у великих обсягах, та дало можливість використовувати імуногістохімію у рутинній клінічній практиці.

Імуногістохімія є невід'ємною методикою багатьох медичних лабораторій для діагностичних та дослідницьких цілей [17]. За останнє десятиліття здатність виявляти антигени (Ags) у зрізах тканин значно покращилася, головним чином, протидіючи шкідливим ефектам формальдегіду за допомогою вилучення антигенів (AR) та підвищуючи чутливість систем виявлення.

Важливою можливістю є те, що ІГХ можна проводити на свіжозаморожених зразках чи на фіксованих в формаліні та залитих у парафінові блоки тканинах. Далі, ІГХ нарізують у тонкий шар (4-5 мкм) мікротом, а потім зафіксовують його на скельці.

Існують різні методи виявлення антигенів:

– імуноферментні методи подвійного маркування. Ці методи широко використовуються і стали повсякденними для виявлення специфічних антигенів та антитіл у біологічних рідинах (сироватці крові, сечі, слині, спинномозковій рідині, молоці тощо). Проте вони забирають багато часу і дуже схильні до хибнопозитивного фарбування. Крім того, їх використання для виявлення різних антигенів в одному клітинному типі не підходить [17];

– подвійне маркування імунофлюоресценції. Метод виявляє більше ніж один антиген як у звичайно оброблюваних парафінових, так і в зрізах тканин кріостату з використанням комерційно доступних реагентів. Подвійне маркування імунофлюоресценції виконується швидше і є методом вибору за допомогою одновалентних фрагментів Fab для виявлення двох антигенів, які, як очікується, локалізуються в одному клітинному типі.

Основним реагентом, імуногістохімічних методів, є антитіло. Поява нових антисироваток, їх фракцій, імуноглобулінів та моноклональних антитіл збільшила кількість і якість імуногістологічних методів.

Антитіла належать до групи білків, які називаються імуноглобулінами (Ig). Ці білки є присутніми в крові імунізованих тварин. Існує п’ять основних класів імуноглобулінів: IgG, IgA, IgM, IgD і IgE. Їх кількість зменшується в плазмі крові або сироватці у вказаному порядку. Для збору фракції сироватки (антисироватки), з крові видаляються клітини і фібрин. На рисунку 1.1 схематично зображено імуноглобулін. Він складається з двох однакових важких ланцюгів (H) і двох однакових легких ланцюгів (L). H-ланцюги відрізняються антигенними властивостями і структурою, і визначають клас і підклас молекули. Два L-ланцюги мають тип каппа (κ) або лямбда (λ). Розподіл κ і λ ланцюгів відрізняється у всіх класах і підкласах імуноглобулінів, а також між різними видами. Ковалентні міжланцюгові дисульфідні зв’язки з’єднують L-ланцюг з H-ланцюгом і H-ланцюг з H-ланцюгом. Вони надають молекулі імуноглобуліну велику стабільність.

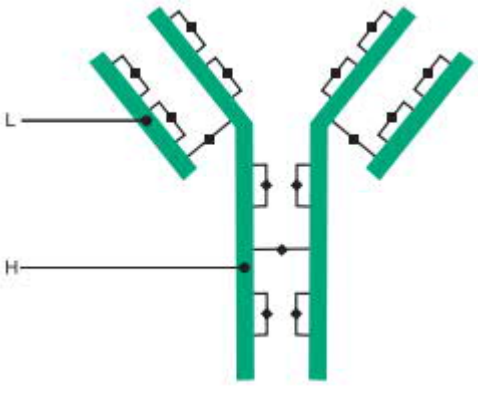

Рисунок 1.1 – Схематичне зображення структури молекули імуноглобуліну

Найчастіше в імуногістохімії використовуються імуноглобуліни IgG і IgM [19, 20].

Розглянемо структуру IgG. Позначимо важкі ланцюги IgG – гамма (γ) ланцюгами. IgG має загальну формулу $\gamma_2 \cdot \kappa_2$ або $\gamma_2 \cdot \lambda_2$. З формули випливає, що одна молекула IgG (молекулярна маса 150 кДа) складається з двох важких γ ланцюгів і двох легких ланцюгів, що належать до типу κ або λ. На рисунку 1.2 показана структура молекули IgG (визначена частково за допомогою протеолітичного розщеплення і редуктивного розщеплення молекули). Розщеплення папаїном призводить до руйнування зв'язку, який є чутливий до дії ферменту, і розташований з боку N-кінцевої ділянки від дисульфідних зв'язків, що з'єднують важкі ланцюги. Це призводить до утворення двох моновалентних антигензв'язуючих фрагментів (Fab) і одного кристалічного фрагмента (Fc). Пепсин розщеплює γ-ланцюги з боку С-кінцевої ділянки від дисульфідних зв'язків, що з'єднують важкі ланцюги, в результаті чого утворюється один бівалентний антигензв'язуючий фрагмент $F(ab')_2$, а Fc-фрагменти руйнуються. Редуктивне розщеплення молекули IgG руйнує міжланцюгові дисульфідні зв'язки і призводить до утворення двох Н-ланцюгів (молекулярна маса кожного становить 50 кДа) і двох L-ланцюгів (по 25 кДа кожен).

Молекулу IgG можна розділити на домени: варіабельні (V) і константні (С). Кожен домен, у свою чергу, складається з 110-120 амінокислот і одного внутрішньоланцюгового дисульфідного зв'язку. На варіабельному домені легкого ланцюга (V_L), і на варіабельному домені важкого ланцюга (V_H) розташовуються аміно-кінцеві ділянки молекули імуноглобуліну. V_L і V_H разом утворюють антигензв'язуючий центр. В межах V_L і V_H доменів антитіла розташовуються кілька гіперваріабельних (HV) ділянок. Під час реакції з антигенами HV ділянки зближуються з антигенною детермінантою (епітопом). Відстань між антигеном і HV ділянками антитіла складає близько 0,2-0,3 нм.

У цій області локалізуються унікальні структури – ідіотипові детермінанти. Кожен клон антитіла видає свій власний ідіотип. Кожен L-ланцюг до V_L домену містить константний домен (C_L). Н-ланцюг теж має три константних домени (C_H1, C_H2 і C_H3) і несе карбоксильну кінцеву ділянку

імуноглобуліну. В домені C_H2 є вуглеводна частина молекули IgG і деякі високо гідрофобні нейтральні ароматичні амінокислоти. Між доменами C_H1 і C_H2 H-ланцюгів локалізуються шарнірні області.

Також існують підкласи IgG: IgG_1, IgG_{2a}, IgG_{2b}, IgG_3 і IgG_4. Вони відрізняються незначними відмінностями в шарнірних областях.

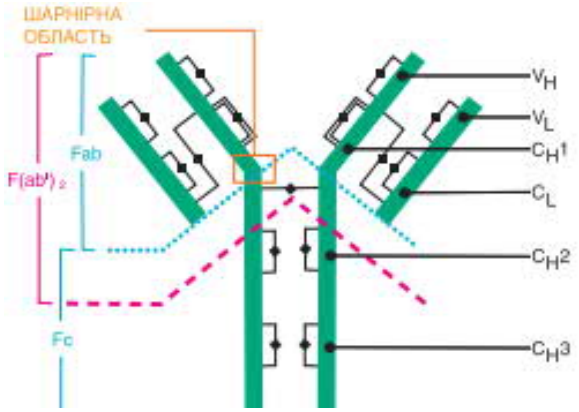

Рисунок 1. 2 – Структура IgG кролика

Множинні антигени у ділянках тканин можна виявляти імунофлюоресценцією з подвійним маркуванням. Вона є хорошою альтернативою і надійною методикою.

Fluorescence in situ hybridization, FISH) — це цитогенетичний метод, який використовується для визначення і локалізації певної послідовності ДНК на хромосомі.

Зібраний матеріал біопсій, ексцизій і резекцій повинен оброблятися швидко, щоб зберегти тканину для FISH. Спочатку зразки тканини розміром 3-4 мм фіксують в 10% нейтральному забуференому парафіні протягом 18-24 годин. Далі проходить дегідратація і заливка в парафін. Потім іде виготовлення зрізів товщиною 4-6 мкм, монтування їх на позитивно заряджені скла (наприклад, SuperFrost Plus, Mentel-Glaser, Termo Scientific), просушування при температурі 60°C протягом 1 години.

Завдяки такій формаліновій фіксації сформовані зв’язки між білками і нуклеїновими кислотами ефективно фіксують тканину, зберігаючи відносно інтактною її структуру. Однак, макромолекулярні зв’язки, які створені

формаліном, значно ускладнюють доступ зондів FISH до ДНК. Отже, перші кроки FISH фарбування повинні бути спрямовані на руйнування цих зв'язків.

Універсальної системи оцінки імунозафарбовування немає, і в ряді випадків дослідники обмежуються тільки якісною оцінкою реакції і висловлюють її як негативна (-), слабо позитивна (+), помірно позитивна (++) і сильно позитивна (+++) реакції .

Результати імуногістохімічної реакції (ІГР) можуть оцінюватися напівкількісним способом. Існують кілька систем напівкількісної оцінки результатів ІГР, такі як H. score, Allred score, Quick score, які успішно застосовуються при оцінці рецепторів естрогену і прогестерону при раку молочної залози [21].

Система гістохімічного підрахунку Histochemical score (Hscore.) була розроблена McCalny і співавт. (1985) для оцінки експресії рецепторів естрогенів і прогестерону. Дана система підрахунку включає інтенсивність імуногістохімічного зафарбовування, що оцінюється за 4-бальною шкалою і частку (%) зафарбованих клітин, і являє собою суму добутків відсотків, що відображають частку клітин з різною інтенсивністю фарбування на бал, відповідний інтенсивності реакції. Інтенсивність фарбування оцінюється від 0 до 4 балів: 0 – немає фарбування, 1 – слабке фарбування, 2 – помірне фарбування, 3 – сильне, 4 – дуже сильне фарбування. Формула підрахунку наступна [21]:

$$\text{Histochemical score} = I \times P,$$

де I – інтенсивність фарбування, виражена в балах від 0-4,

P – відсоток клітин, забарвлених з різною інтенсивністю.

Максимальне значення HScore відповідно повинно бути 300. Підрахунок проводиться в трьох когортах по 100 пухлинних клітин в різних полях зору (об. ×40; ок. ×10).

Алгоритм підрахунку наступний: наприклад: 20% нефарбованих клітин, 10% слабофарбованих клітин, 30% помірної інтенсивності фарбування і 40%

сильнофарбованих. Перемножуємо відсоток клітин на бал інтенсивності і, підсумовуючи, отримуємо [21]:

$$\text{HScore } (20 \times 0 + 10 \times 1 + 30 \times 2 + 40 \times 3) = 190.$$

Результат HScore трактується наступним чином.

Від 0 до 10 – негативний (рисунок 1.3),

від 10 до 100 – слабопозитивний (рисунок 1.4),

від 100 до 300 – позитивний (рисунок 1.5).

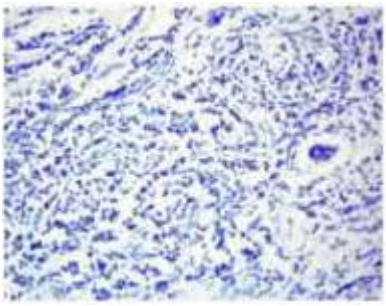

Рисунок 1.3 – Рецептори естрогенів в клітинах раку молочної залози, що виявлені імуногістохімічним методом

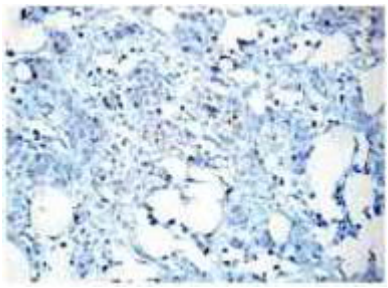

Рисунок 1.4 – Рецептори естрогенів в клітинах раку молочної залози, що виявлені імуногістохімічним методом (45 балів – за шкалою HScore, 5 балів – за шкалою Allred). Зб. 200

Система оцінки рецепторів естрогенів і прогестерону в пухлинних клітинах раку молочної залози (за D. C. Allred). Рецептори естрогенів і прогестерону вважаються основними критеріями чутливості раку молочної

залози до гормонотерапії. Вони є першими тканинними маркерами, визначення яких увійшло в практику лікування хворих на рак молочної залози.

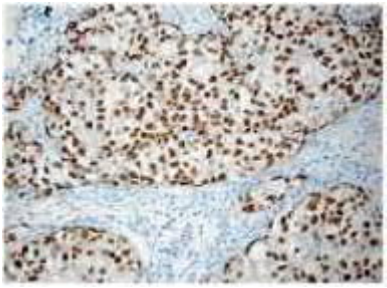

Рисунок 1.5 – Рецептори естрогенів в клітинах раку молочної залози, що виявлені імуногістохімічним методом (220 балів – за шкалою HScore, 8 балів – за шкалою Allred). Зб. 200

Система оцінки за D. C. Allred полягає в підрахунку суми балів [бал частки (%) зафарбованих клітин (1) і бал ступеня інтенсивності фарбування) пухлинних клітин, що містять рецептори естрогенів і прогестерону (2) (таблиці 1.1, 1.2) (рисунок 1.6)].

Таблиця 1.1 – Оцінка частки (%) і інтенсивності зафарбованих пухлинних клітин, що містять рецептори естрогенів і прогестерону

Оцінка частки (%) зафарбованих клітин		Оцінка інтенсивності фарбування	
Бал	Частка (%) з зафарбованих клітин	Бал	Інтенсивність зафарбованих клітин
0	0	1	Слабка
1	> 0 < 1/100(<1%)	2	Помірна
2	≥ 1/100 і < 1/10 (від 1 до 10%)	3	Виражена
3	≥ 1/10 і < 1/3 (від 10 до 33%)		
4	≥ 1/3 і < 2/3 (від 33 до 66%)		
5	≥ 2/3 і ≤ 1 (> 66%)		

Таблиця 1.2 – Результат інтерпретації отриманої суми балів за D. C. Allred

Сума балів	Результат
0 – 2	Негативний
3 – 8	Позитивний

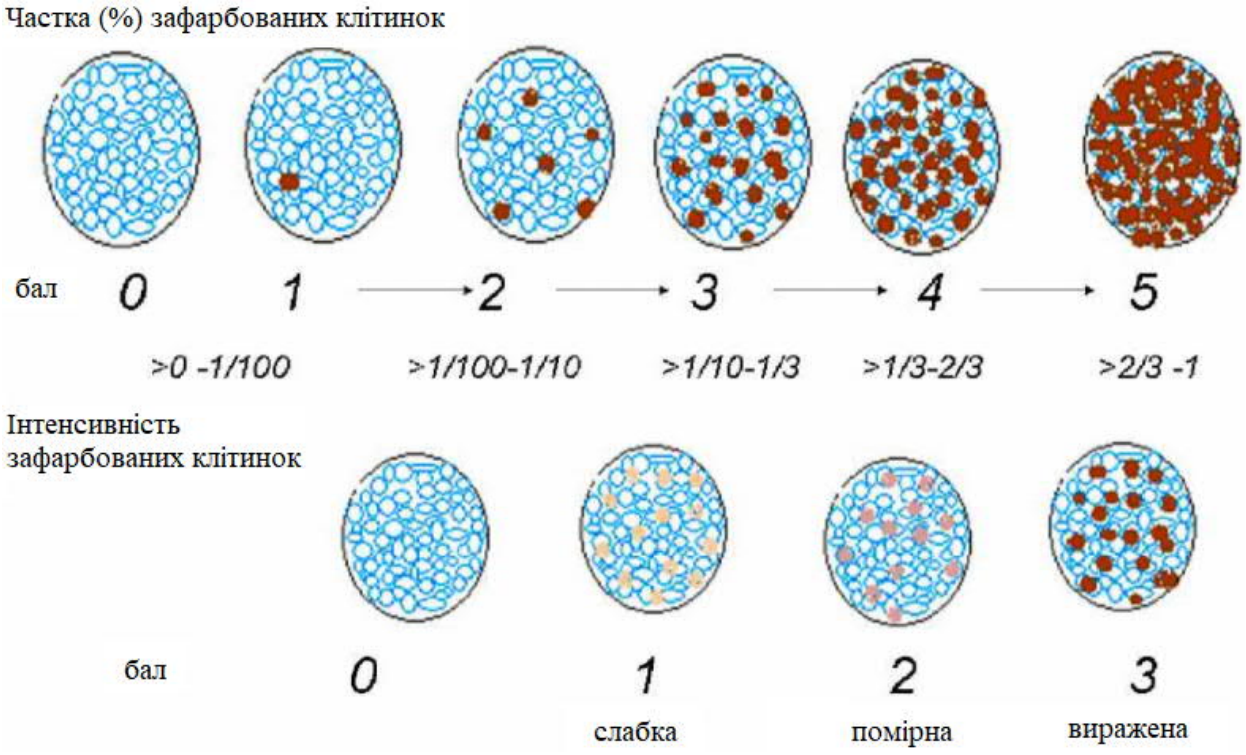

Рисунок 1.6 – Схема системи підрахунку за D. C. Allred

Система оцінки ІГХ реакції онкобілка HER 2. Ген c-erbB2 (HER-2/neu) є гомологом v-erbB онкогена вірусу еритробластоми птахів. Розмір його мРНК складає 4,5 т.п.н .; в ній закодована інформація для синтезу глікопротеїду p185 (1225 амінокислот). Цей білок має 44% гомології з рецептором епідермального фактору росту [21].

HER-2 рецептор є унікальним представником сімейства трансмембранних тирозинкіназ, так як, не маючи власного ліганда і не взаємодіючи ні з одним з відомих факторів росту, що активують родинні рецептори, він є, тим не менше, ключовою ланкою передачі мітогенних сигналів всіх пептидів, подібних епідермальному фактору росту, і необхідний для успішного функціонування всієї системи. Встановлено, що у жінок, які страждають на рак молочної залози,

відбувається активація гена HER-2/neu, в результаті чого на поверхні пухлинних клітин експресуються рецептори HER-2.

Рецептори HER-2, зв'язані з факторами росту, є найважливішою ланкою регуляторної системи передачі мітогенного сигналу. Вважається, що їх наявність в клітинах раку молочної залози забезпечує її здатність до саморегуляції росту, тобто порушують нормальний клітинний цикл і змушують клітини безконтрольно ділитися.

Ампліфікація протоонкогена HER-2/neu (визначається при FISH) призводить до гіперекспрессії відповідного онкогена (від 2 до 40 разів перевищує норму), знайдену приблизно в 30% випадків раку молочної залози і асоційовану з несприятливим прогнозом перебігу захворювання (рисунок 1.7)

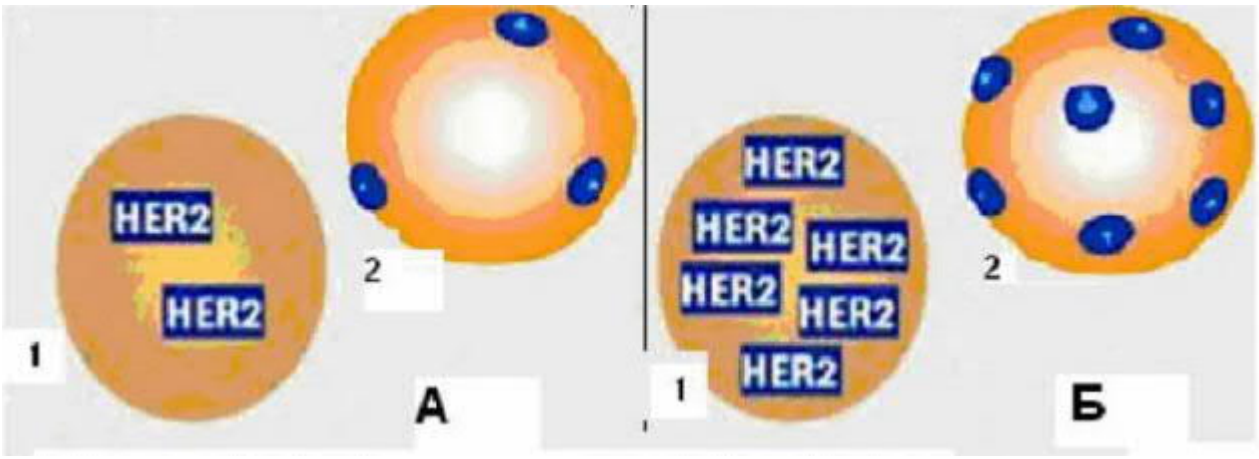

Рисунок 1.7 – Вміст HER-2/neu гена (1) и онкобілка (2) в нормальній клітині (А) і в пухлинній клітині (Б)

Результат оцінки фарбування HER-2 онкобілка представлений в таблиці 1.3.

Результати імуногістохімічної реакції на онкобілок HER-2 представлені на рисунках 1.8-1.11.

Таблиця 1.3 – Система імуногістохімічної оцінки HER-2/neu онкобілка в пухлинних клітинах

Бал	Інтенсивність і відсоток зафарбованих пухлинних клітин
0	Немає фарбування або фарбування <10% пухлинних клітин
1	Неповне фарбування мембрани, забарвлено > 10% пухлинних клітин
2	Повне фарбування мембрани, слабке або помірне забарвлення > 10% пухлинних клітин
3	Повне фарбування мембрани, сильне фарбування > 30% пухлинних клітин

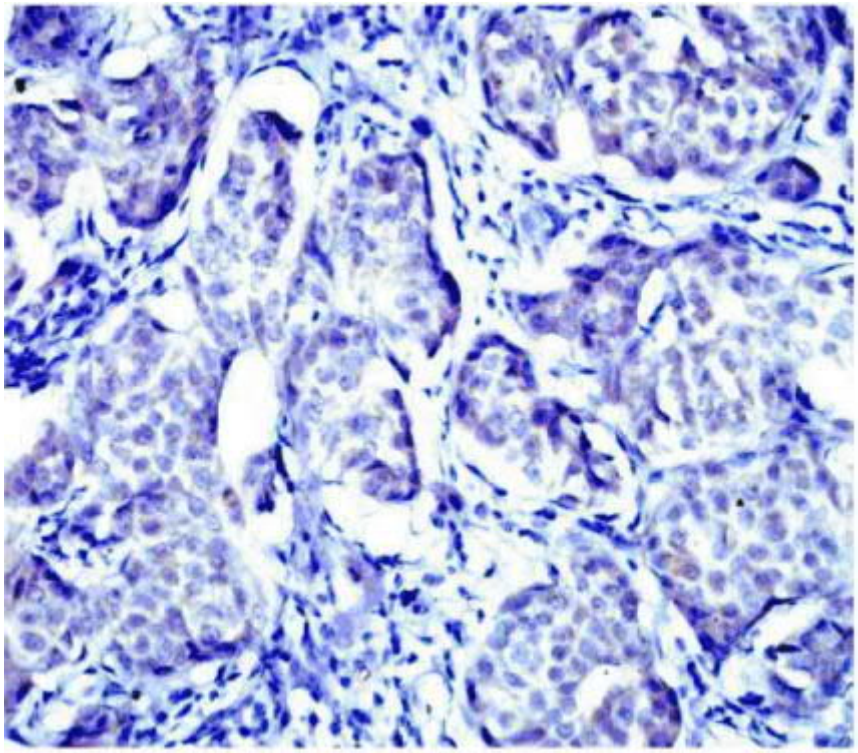

Рисунок 1.8 – Відсутність експресії онкобілка HER-2 в клітинах раку молочної залози. [(HER-2 (0)-фенотип)]. Зб. 200

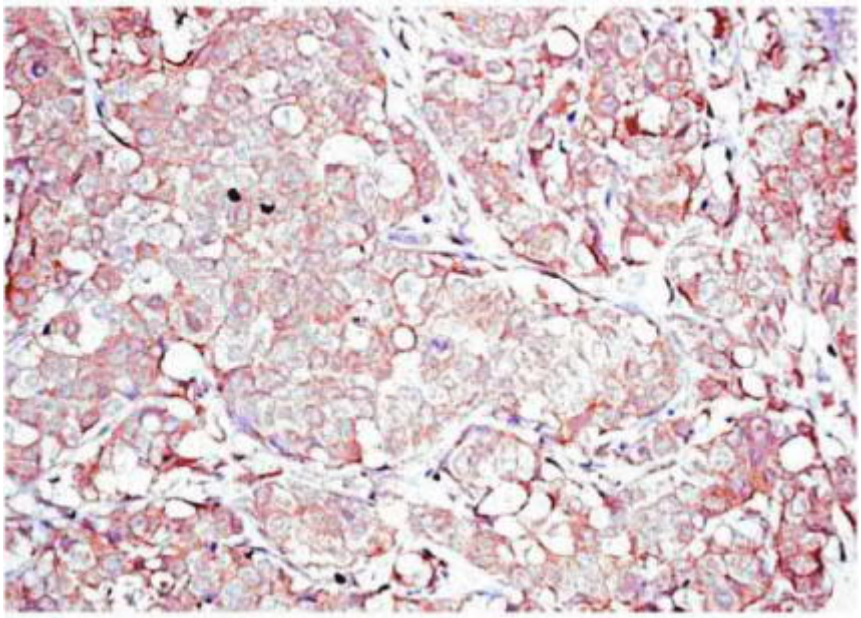

Рисунок 1.9 – Експресія онкобілка HER-2 в клітинах раку молочної залози, виявлена імуногістохімічним методом. [(HER-2 (1+)-фенотип)]. Зб. 200

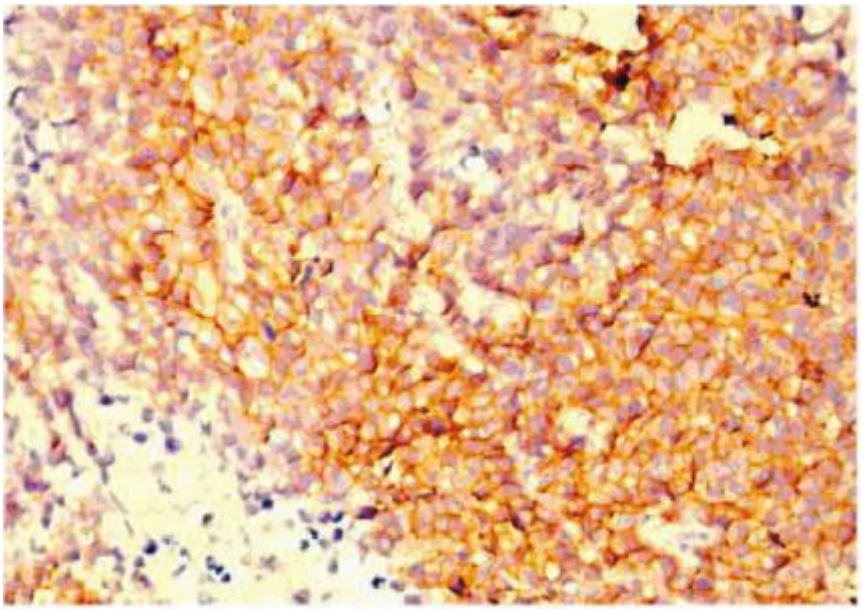

Рисунок 1.10 – Надекспресія онкобілка HER-2 в клітинах раку молочної залози, виявлена імуногістохімічним методом. [(HER-2 (2+)-фенотип)]. Зб. 200

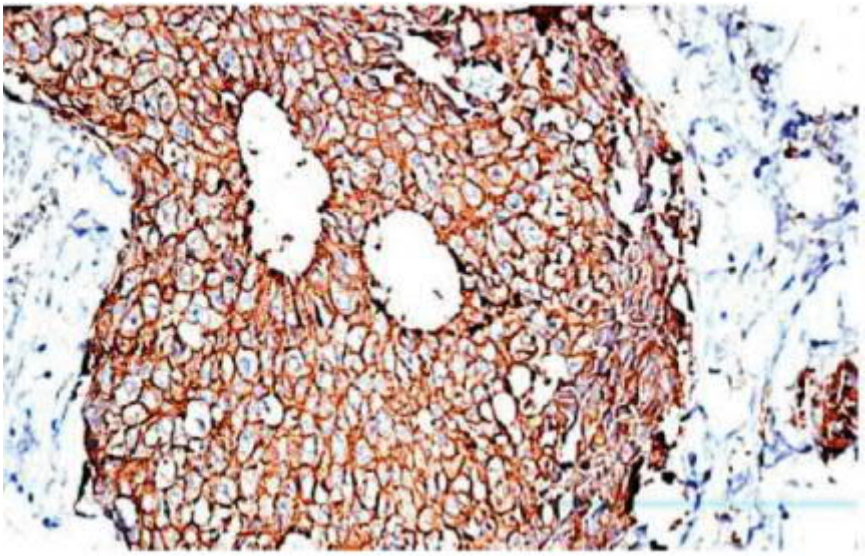

Рисунок 1.11 – Надекспресія онкобілка HER-2 в клітинах раку молочної залози, виявлена імуногістохімічним методом. [(HER-2 (3+)-фенотип)]. Зб. 200

1.2 Методи та алгоритми генерування зображень

Задача генерування зображень виникла на початку 60-х років 20 ст. Спричинена вона була бурхливим розвитком ЕОМ [22, 23]. На цей час проводилися дослідження, які, були спрямовані на збільшення об’єму інформації, що обробляється. Розвиток синтезу зображень можна розділити на три періоди:

- початковий період (початок 60-х – 70-х рр.);
- період навчання (початок 70-х – 80-х рр.);
- сучасний період (початок 80-х. рр. і до теперішнього часу).

Початковий період галузі синтезу зображень характеризується побудовою двомірних (2D), а потім і тривимірних (3D) об’єктів за допомогою відрізків ліній. Були розроблені методи та алгоритми невидимих частин відрізків, що розвинуло ефект двозначності сприйняття синтезованого об’єкта. З 1965 р. появилися алгоритми, які дозволили створювати рисунки на цифровому графопобудувачі; з 1967 р. – алгоритми побудови складних поверхонь. Також в цей період введено

поняття освітлення і затінення, розроблено методи опису та моделювання складних об’єктів.

Період навчання характеризується двома напрямами. Перший – покращення та подальший розвиток базових технічних пристроїв. Другий – виділення основних концепцій галузі. В рамках першого напрямку відбулося покращення візуальної якості зображень. Появилися методи: «згладжування» поверхонь Гуро, моделювання ефекту освітлення, генерування текстур, рельєфів, багаточисленні методи апроксимації та представлення складних поверхонь. В рамках другого напрямку: класифікація алгоритмів усунення невидимих ліній (розробник Сазерленд), підхід до визначення базового графічного математичного забезпечення, яке не залежить від апаратури (розробник Ньюмен-Спрулл).

В сучасний період галузі синтезу зображень була проведена класифікація зображень на:

− абстрактні (такі, які не містять реальної інформації, а створюють або викликають певні враження, асоціації);

− символічні (такі, як діаграми, гістограми, графіки, карти шляхів (погоди і т. п.), тобто ті, що відображають кількісну, топологічну, структурну та ін. інформацію);

− спрощено-фігурні (такі, що в спрощеному вигляді представляють за допомогою рисунка або ескізу предмети реального світу);

− реалістичні (такі, що надають зображуваним об’єктам вигляд реальних об’єктів).

Часто цей поділ є суб’єктивним.

Перейдемо до підходів синтезу складних зображень. На сучасному етапі існує багато методів синтезу. Більшість методів синтезу використовує множину елементарних об'єктів та множину операцій [24-26], які застосовуються до елементарних об'єктів з метою створення більш складних, плавно поєднаних об'єктів. Тому формування множини операцій, перетворень та відношень між об'єктами відіграє важливу роль в процесі синтезу складних зображень. Тому існує цілий ряд теоретико-множинних операцій, перетворень та відношень, які

підвищують ефективність процесу синтезу. Найбільше використовуються: логічні (булеві) операції, а також операції змішування, офсетінгу, проектування, відношення належності.

Характерною рисою майже всіх відношень є те, що для більшої наочності виконуваних дій вони використовують деревовидну (графову) структуру. Причому вершинами такої графової структури виступають операції та відношення.

Основні теоретико-множинні операції та інші перетворення, використовуються для створення складних об'єктів з функціонально визначених більш простих примітивів. Множина геометричних операцій Φ описується наступним чином:

$$\Phi : M^1 + M^2 + ... + M^n \to M,$$

де n – кількість операндів в операції.

При застосуванні логічних операцій до множини примітивів з метою створення більш складних конструкцій використовуються булеві операції, такі як перетин та об'єднання двох примітивів, які описуються функціями f_1 та f_2. Останні мають вигляд:

$$f_1 \cap f_2 = \max(f_1, f_2), \qquad f_1 \cup f_2 = \max(f_1, f_2).$$

R-функції. Рвачов сформулював свій підхід до цієї задачі в термінах *R*-функцій [27, 28]. Одним з можливих аналітичних описів *R*-функцій є:

$$f_1 \cup f_2 = \frac{1}{1+\alpha}(f_1 + f_2 + \sqrt{f_1 + f - 2 \cdot \alpha \cdot f_1 \cdot f_2}),$$

$$f_1 \cap f_2 = \frac{1}{1+\alpha}(f_1 + f_2 - \sqrt{f_1 + f - 2 \cdot \alpha \cdot f_1 \cdot f_2}),$$

де $\alpha = \alpha(f_1, f_2)$ – довільна неперервна функція, яка задовольняє наступні умови:

$$-1 < \alpha(f_1, f_2) \le 1, \quad \alpha(f_1, f_2) = \alpha(f_2, f_1) = \alpha(-f_1, f_2) = \alpha(f_1, -f_2).$$

Аналітичний вираз для віднімання описується за допомогою заперечення та перетину:

$$f_1 \setminus f_2 = f_1 \cap (-f_2).$$

Для забезпечення неперервності використовують наступну модифікацію *R*-функцій:

$$f_1 \cup f_2 = \frac{1}{2}(f_1 + f_2 + \sqrt{f_1^2 + f_2^2}) \cdot (\sqrt{f_1^2 + f_2^2})^{m/2},$$

$$f_1 \cap f_2 = \frac{1}{2}(f_1 + f_2 - \sqrt{f_1^2 + f_2^2}) \cdot (\sqrt{f_1^2 + f_2^2})^{m/2}.$$

Ще одним із аналітичних виразів псевдо-булівських операцій є операції степені *n*, які носять назву модифікованих операцій Ріккі:

$$f_1 \cap f_2 = f_1^n + f_2^n,$$
$$f_1 \cup f_2 = (f_1^{-n} + f_2^{-n})^{-1},$$
$$f_1 \setminus f_2 = f_1^n + f_2^{-n}.$$

На рисунках 1.12, 1.13 показані зображення, які побудовані за допомогою *R*-функцій [29].

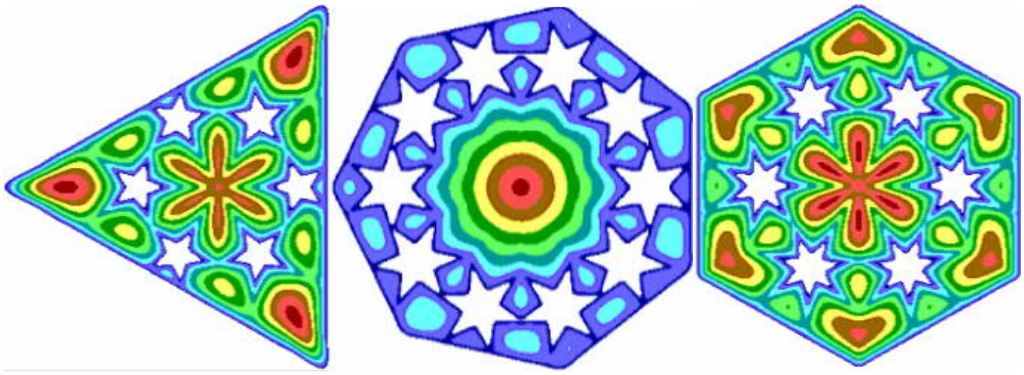

Рисунок 1.12 – Зразки зображень, які побудовані за допомогою *R*-функцій

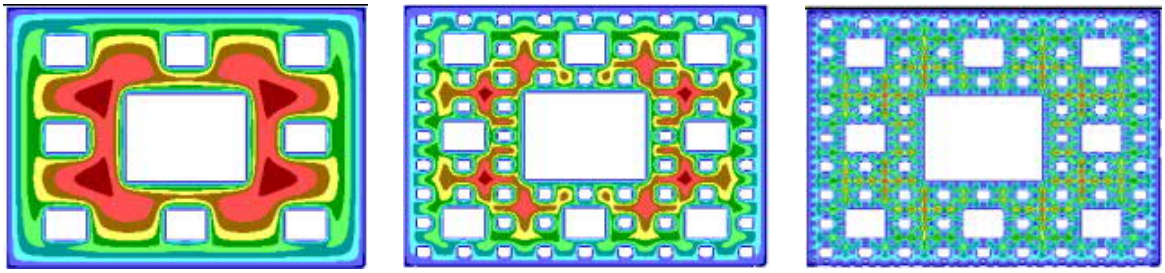

Рисунок 1.13 – Килими Серпінського, які побудовані за допомогою *R*-функцій

Фрактальна графіка. Опишемо алгоритм синтезу цього виду зображень. Відрізок ділиться на N рівних частин, кожну з яких можна рахувати копією вихідного відрізка, зменшеною в $1/r$ раз. N і r зв'язані співвідношенням $N \cdot r = 1$. Якщо квадрат розбити на N рівних квадратів площею в $1/r^2$ меншою площі вихідного квадрата, то співвідношення прийме наступний вигляд $N \cdot r^2 = 1$. Аналогічно, якщо з кубом зробити таке розбиття, то отримаємо співвідношення $N \cdot r^3 = 1$. В загальному, якщо розмірність об'єкта d, то отримуємо співвідношення між N рівними підоб'єктами і коефіцієнтами подібності r: $N \cdot r^d = 1$. Звідки:

$$d = \frac{\log N}{\log \frac{1}{r}}.$$

Найпростішим прикладом фракталів є сніжинка Коха (розробник Гельг Кох, 1904 р.). Границя сніжинки описується кривою, що складена з трьох однакових фракталів, які мають розмірність d=1,2618. Кожна третина сніжинки будується

ітеративно, починаючи з однієї зі сторін рівностороннього трикутника. Нехай K_0 початковий відрізок. Забираємо середню третину і додаємо два нових відрізки такої ж довжини. Отримана множина – K_1. Дана процедура повторюється багаторазово, на кожному кроці, заміняючи середню третину двома новими відрізками. Після *n*-го кроку одержимо фігуру K_n.

Ще одна важлива властивість, якою володіє границя сніжинки Коха — її нескінченна довжина [30-32].

Особливої популярності для генерування зображень набирають штучні нейронні мережі. На рисунку 1.14 приведена класифікація ШНМ.

Генеративно-змагальні мережі складаються з двох, одна з яких генерує зразки, інша намагається відрізнити справжні зразки від згенерованих.

Згорткові нейронні мережі являють собою деякі архітектури: складні функції з великою кількістю параметрів, елементарні операції, що відносяться до певних типів: узагальнені згортки, зменшення розмірності зображення (пулінг), поелементна нелінійність, що застосовується до індивідуальних вимірів, і множення на матрицю. Така згорткова нейромережа бере послідовність цих операцій, бере деяке зображення і перетворює подібний набір з трьох зображень в набір зі ста зображень, де кожне є значущим [33].

Рисунок 1.14 – Види основних нейронних мереж

Далі застосовуються деякі нелінійності, і за допомогою операцій узагальненої згортки утворюється новий набір зображень. Відбувається кілька ітерацій, і в якийсь момент такий набір зображень перетворюється в багатовимірний вектор. Далі ще ряд множень на матрицю і поелементних нелінійностей і цей вектор перетворюється в вектор властивостей вхідного зображення, яке ми будували. Наприклад, цей вектор може відповідати нормованим можливостям або ненормованим величинам, де великі значення свідчать про те, що той чи інший клас об'єктів присутній на зображенні.

На рисунку 1.15 показані зображення текстур, які побудовані за допомогою згорткових нейронних мереж. Зліва на рисунку оригінал, а справа – синтезовані.

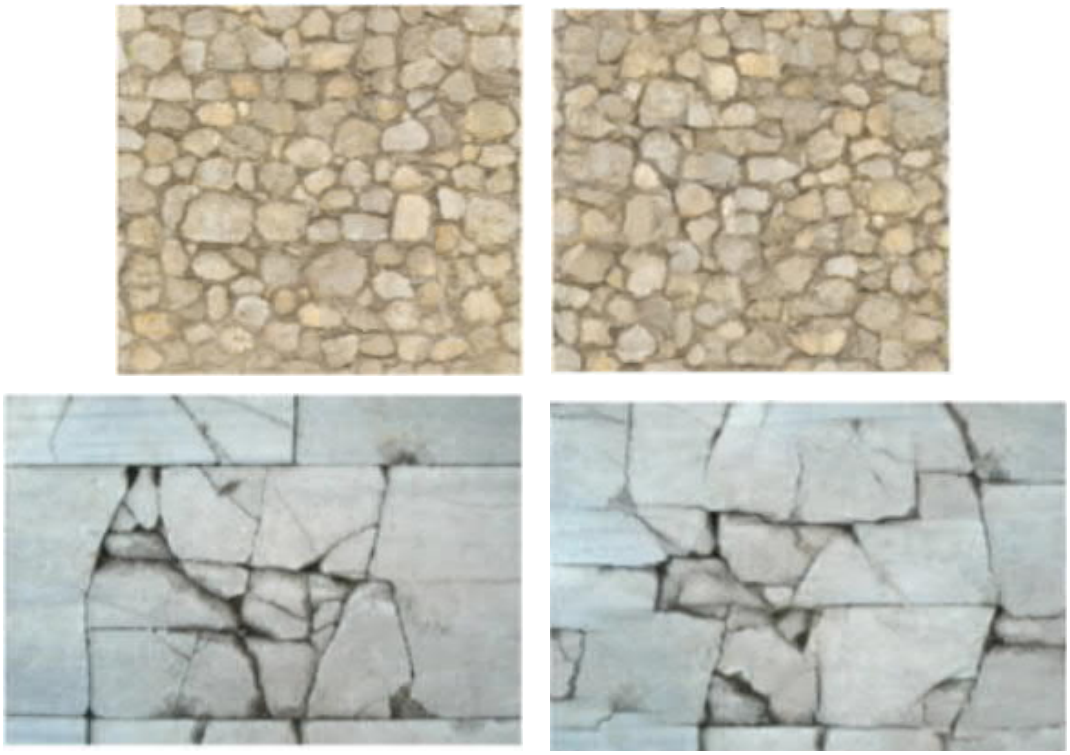

Рисунок 1.15 – Текстури. Зліва оригінал, справа – синтезовані за допомогою нейромереж

Як видно з рисунку 1.15 дана модель синтезу працює дуже добре.

1.3 Програмні засоби генерування зображень

Представлення даних на моніторі у графічному виді вперше відбулося всередині 1950-х років для великих ЕОМ. Графічна інформація, яка була відображена, стосувалася наукових і військових досліджень.

Комп'ютерна графіка – це спеціальна ділянка інформатики, що вивчає методи і засоби створення та обробки зображень за допомогою програмно-апаратних обчислювальних комплексів [22, 34].

На сьогоднішній день майже усі доповіді у сфері бізнесу, науки та ін. використовують комп'ютерні презентації. Популярними є комп'ютерні ігри, наукова графіка, фільми.

Більшість комп'ютерних систем володіють графічним способом відображення даних. Програмне забезпечення різного класу має графічний інтерфейс. Комп'ютерна графіка охопила всі види та форми представлення зображень, як на екрані монітора, так і на зовнішньому носії (папір, плівка, тощо).

Комп'ютерна графіка стала складним комплексом створення зображень і умовно поділилася на ряд напрямів:

– двомірна графіка (2D-графіка);
– поліграфія;
– мультимедіа;
– 3D-графіка;
– відео монтаж;
– програми САПР (чи CAD – computer-aided design).

2D-графіка є основою всієї комп’ютерної графіки [35, 36]. Кожен фахівець комп’ютерного дизайну повинен знати базові положення двохмірної графіки і володіти ними. Зображення в 2D-графіці має два виміри – ширину і висоту. Тому воно завжди виглядає плоским. 2D-графіка використовується для створення логотипів, карт, сайтів, рекламних банерів, в іграх і інтерфейсах додатків, мультфільмах і відеофільмах. Незважаючи на те, що 2D графіка виглядає як плоске

зображення, за рахунок тіней можна домогтися ефекту об'ємних об'єктів (але не фотореалістичності).

Напрям поліграфія – це робота у періодичних виданнях, створення візиток, бланків, рекламних листівок, буклетів, плакатів тощо. Працівники цього напряму працюють в програмах верстки сторінок. Найбільш популярними програмами є Adobe PageMaker і Quark Press. Програми верстки сторінок дають можливість з’єднувати разом текстову і графічну інформацію для створення інформаційних бюлетенів, журналів, брошур і рекламної продукції.

Adobe PageMaker має високоякісні зрозумілі дизайнерські інструменти для професійних дизайнерів і інших фахівців, в чиї обов'язки входить верстка і підготовка до різних публікацій, наприклад, брошур або офіційних бланків [37-39]. Додаток містить колекцію шаблонів і готових зображень. Шаблони бувають трьох різних типів: beginning, intermediate і advanced. Adobe PageMaker можна інтегрувати з іншими додатками Adobe, що робить роботу більш продуктивною. Блоки тексту і зображень можна вставляти в готові макети. В Adobe PageMaker можна імпортувати файли з найбільш популярних додатків Adobe.

В даний додаток можна розміщувати файли Illustrator, можна додавати в макет зображення Photoshop, розташовувати документи Adobe PDF або використовувати засоби малювання для проведення ліній, створення прямокутників, кіл і овалів, а також багатокутників, до яких може бути застосоване штрихування і заливка. Ці форми можуть являти собою самостійні графічні елементи, а також містити текст або імпортовані зображення. Аналогічно з текстом. Можна вводити текст безпосередньо в програмі PageMaker або імпортувати його з інших додатків. З текстовими блоками можна виконувати ті ж операції, що і з іншими об'єктами, (наприклад, вибирати і переміщати їх, а також змінювати їх розміри).

PageMaker підтримує всі провідні стандарти друку, а це означає, що матеріали можуть виводитися на будь-яких типах друкованих пристроїв, включаючи високотехнологічні принтери для друку комерційної продукції.

Файл Adobe PageMaker має декілька шарів. Можна зберігати кілька версій публікації в одному файлі, додавати коментарі, а також експериментувати з

розміщенням об'єктів в різних шарах. Помістивши текст в окремий шар, можна контролювати видимість об'єктів, а можливість відключення шару з графікою дозволить працювати швидше. Використання шарів забезпечує численні переваги.

Кожен пристрій, що використовується в процесі створення публікацій (включаючи сканери, монітори, настільні та друкарські принтери) здатний відтворювати або відображати обмежений набір кольорів. Система управління кольором витягує колірні характеристики, характерні для кожного пристрою і впорядковує зібрану інформацію. Ефективне управління кольором дозволяє забезпечити максимально високу якість друку.

Додаток має додаткові функції друку. Плагін Save for Service Provider допомагає зібрати в окремому файлі всю інформацію, необхідну для виведення публікації на друк. Плагін Build Booklet дозволяє формувати багатосторінкові публікації і виводити їх на друк з дотриманням нумерації сторінок.

Мультимедіа – це область комп'ютерної графіки, яка зв'язана зі створенням інтерактивних енциклопедій, довідкових систем, навчальних програм і інтерфейсів до них. Найбільш відомими програмами створення мультимедії є Macromedia Director і MS Power Point.

З допомогою MS PowerPoint створюються яскраві і наочні презентації, що складаються з набору слайдів і включають текст, зображення, таблиці, графіки, діаграми, блок-схеми, 3D-моделі, аудіо, відео [40-42]. Презентації можна відтворювати на великих екранах, в тому числі за допомогою кінопроектора.

Програма має безліч шаблонів оформлення презентацій в залежності від тематики: бізнес, освіта, маркетинг і ін. Також є можливість створення власних шаблонів PowerPoint, а також завантаження свіжих тем з сайту Microsoft.

MS PowerPoint включає інструменти для форматування тексту, побудови різних видів діаграм і графіків, роботи з зображеннями, фігурами і мультимедіа; можливе застосування анімованих ефектів, наприклад, плавне наближення, розчинення, виліт і т.д. Програма також дозволяє налаштовувати події, які будуть відбуватися при кліці або наведенні миші на вибрані об'єкти.

PowerPoint має версії для Microsoft Windows і Mac OS, а також мобільних пристроїв. Основним форматом збереження файлів в PowerPoint є PPTX

(PowerPoint Open XML Presentation). Програма також підтримує інші формати, в тому числі: PPT (використовувався в старих версіях PowerPoint), PPSX, PPSM, POTX, POTM, PDF, GIF, JPG, PNG, XML.

3D-графіка сьогодні знайшла своє використання в індустрії комп'ютерних ігор, архітектурі, телевізійній рекламі, оформленні телевізійних каналів тощо [33, 34]. З допомогою 3D-графіки створюються штучні предмети і персонажі, їх анімація і сполучення з реальними предметами й інтер'єрами. Найбільш популярні програмні продукти для 3D графіки: 3D Max; Maya; Softimage; Lightwave 3D; Houdini; Modo; Rhinoceros 3D; Cinema 4d.

Тривимірну графіку також використовують в науці, промисловості, медицині і архітектурі. В промисловості її використовують разом з технологією швидкого прототипування – створенням дослідних зразків і моделей.

3D графіка тісно пов'язана з 3D моделюванням – процесом створення 3D моделей. За допомогою 3D графіки і 3D моделювання створюються точні або зменшені чи збільшені копії конкретного об'єкта. Можна також створити модель неіснуючого або невидимого для людини об'єкта (предметне моделювання).

В 3D графіці виокремлюють такий напрямок, як CGI графіка (computer-generated imagery) – це створення нерухомих або рухомих зображень, створених за допомогою тривимірної графіки, і які використовують в кінематографі, образотворчому мистецтві, друкованій продукції, на телебаченні і т.п. (крім комп'ютерних ігор). Застосування комп'ютерної графіки CGI дозволяє в кіно замінити роботу каскадерів, масовки і навіть декорації.

Напрям комп'ютерної графіки відеомонтаж маніпулює «живими» картинками. Відеомонтаж використовує власну технологію роботи і тим відрізняється від інших напрямів комп'ютерної графіки. Однією з найбільш популярних програм, що використовується у цій області, є Adobe Premier. Це – програмне забезпечення для нелінійного монтажу (миттєвого доступу до довільного місця будь-якого з монтажних кадрів, що зберігаються в цифровій пам'яті), яке призначене для кінематографістів, творців телевізійних передач, журналістів, студентів і відеооператорів. Adobe Premier підтримує різні комп'ютерні платформи: Windows і macOS.

За час існування кінематографа, телебачення і радіо технології монтажу вдосконалювалися одночасно з тими галузями, в яких застосовувалися. І, якщо на зорі кіно існував тільки механічний монтаж, сьогодні використовується, головним чином, електронний. Електронний монтаж здійснюється перезаписом вихідних монтажних кадрів на іншу магнітну стрічку в потрібному порядку [45].

Програми САПР застосовуються у різних сферах людської діяльності: інженерно-конструкторській діяльності, архітектурі, медицині тощо. З допомогою САПР проходить проектування мікросхем до створення літаків, мікросхем до комп'ютерів, проектування імплантів, архітектурне проектування. Однією з найбільш популярних програм, що використовується у цій області, є AutoCAD фірми AutoDesk. Це могутня система машинного проектування, яку іноді розглядають як електронний кульман, така програма навіть здатна допомогти сформувати бюджет великих архітектурних і інженерних проектів [34, 44].

З допомогою AutoCAD створюють текстову документацію, машинобудівні креслення, графічну документацію для випуску радіоелектронної апаратури, архітектурно-будівельні креслення, креслення для суднобудівної та авіаційної промисловості, картографічну документацію, технічні та художні ілюстрації. Система дозволяє виконувати креслення у кольорі, будувати аксонометричні і тривимірні зображення, створювати бібліотеки та архіви, якими можна користуватися при розробці нових документів та інше [46, 47].

Технологія побудови креслення, що застосовується в системі AutoCAD, дозволяє вводити в креслення раніше заготовленні варіанти креслень деталей, проектувати варіанти розташування обладнання та ін. Висока продуктивність системи досягається як її власними програмними засобами, так і конструкціями виконавця креслення. Їх можна описувати спеціальною мовою AutoLISP, на якій складені програми для системи AutoCAD.

Розвиток інформаційних технологій, проникнення їх в усі сторони життя, надзвичайно підвищують затребуваність комп'ютерної графіки, так як її результатами користується не обмежене коло фахівців, як було раніше, а практично всі без винятку люди взаємодіють з комп'ютерами, мобільними

пристроями, цікавляться комп’ютерними іграми і сучасними творами кіноіндустрії.

1.4 Постановка задач дослідження

Метою кваліфікаційної роботи є програмна реалізація алгоритмів генерування імуногістохімічних зображень.

Для досягнення поставленої мети необхідно:

- проаналізувати об’єкт досліджень – імуногістохімічні зображення;
- проаналізувати методи та алгоритми синтезу;
- проаналізувати програмні засоби синтезу зображень;
- проаналізувати алгоритми і методи роботи генератора та дискримінатора мережі;
- проаналізувати алгоритми і методи роботи генеративно-змагальної мережі її архітектуру;
- програмно реалізувати генератор імуногістохімічних зображень;
- провести навчання генеративно-змагальної мережі;
- провести тестування реалізованих алгоритмів.

2.4 Висновки до розділу 1

Результатами першого розділу є:

- проведений аналіз об’єкту досліджень – імуногістохімічні зображення;
- проведений аналіз методи та алгоритми синтезу;
- проведений аналіз програмних засобів синтезу зображень.

Таким чином, на основі проведеного аналізу показано актуальність проведення наступних досліджень та здійснено постановку задач на наступні розділи.

2 АЛГОРИТМИ ТА МЕТОДИ РОБОТИ ГЕНЕРАТИВНО-ЗМАГАЛЬНИХ НЕЙРОННИХ МЕРЕЖ

2.1 Алгоритми і методи роботи генератора мережі

Останнім часом особливої популярності набирають алгоритми, що базуються на використанні нейромереж, які дозволяють генерувати нові зображення на основі вивчених зображень. Генеративно-змагальні мережі (ГЗМ) поєднують дві штучні нейронні мережі (ШНМ): генератор, який вчиться виробляти вихідний результат, з дискримінатором, який вчиться відрізняти справжні дані від виходу генератора. Ці мережі налаштовані на роботу один проти одного, тому й звідси пішла така назва.

В підрозділі 1.2 при аналізі методів синтезу ми переконалися (рисунок 1.14), що модель синтезу зображень з допомогою нейронних мереж працює дуже добре.

Розглянемо роботу генеруючої нейронної мережі. Спочатку генератор формує зображення. Для цього він генерує довільний шум, з часом на ньому проглядають фрагменти необхідного зображення. І так до тих пір, поки ці фрагменти не стануть більш явними, схожими на реальні. Нейромережа запам’ятовує яким саме чином вона генерувала шум для цього результату, і наступного разу відтворює свої дії. Цей алгоритм описано просто, насправді процес роботи є доволі складним. В якості генератора досить часто використовуються мережі FFNN – нейронні мережі прямого поширення (feed forward neural networks).

Під час тренування генеративна мережа вивчає, які області зображення необхідно змінити або поліпшити, щоб дискримінатору знадобилося більше часу для визначення автентичності згенерованого зображення. Після навчання генеративну модель можна використовувати для створення нових правдоподібних зразків на побажання.

Для кращого розуміння роботи генератора розглянемо генерацію ймовірнісних розподілів випадкових змінних. Існує багато методів. Розглянемо наступні:

- метод зворотного перетворення;
- вибірка з відхиленням;
- алгоритм Metropolis-Hasting.

2.1.1 Метод зворотного перетворення

Метод зворотного перетворення дозволяє генерувати складні випадкові змінні з простих рівномірних випадкових змінних [48]. Метод зворотного перетворення є способом генерації випадкових величин з заданою функцією розподілу. Він проходить шляхом модифікації роботи генератора чисел рівномірного розподілу. Рівномірну випадкову змінну ми знаємо, як генерувати. Метод використовує кумулятивну функцію розподілу.

Опишемо алгоритм.

Нехай $F(x)$ є функцією довільного розподілу. Необхідно, маючи генератор вибірки з стандартного неперервного рівномірного розподілу, отримати вибірку з розподілу, що задається функцією розподілу $F(x)$.

Нехай функція $F:\mathrm{R}\to[0;1]$ строго зростає на всій області визначення, то вона бієктивна, а, отже, має зворотну функцію $F^{-1}:[0;1]\to\mathrm{R}$.

Якщо $U_1,...,U_n\sim U[0;1]$ – вибірка з стандартного неперервного рівномірного розподілу. Тоді $X_1,...,X_n$, де $X_i=F^{-1}(U_i)$, $i=\overline{1,n}$ – вибірка шуканого розподілу (рисунок 2.1).

Як приклад приведемо генерацію вибірки з експоненційного розподілу з параметром $\lambda>0$. Функція цього розподілу $F(x)=1-e^{-\lambda x}$ строго зростає, і її зворотна функція має вигляд $F^{-1}(x)=-\frac{1}{\lambda}\ln(1-x)$. Таким чином, якщо $U_1,...,U_n$ – вибірка з стандартного неперервного рівномірного розподілу, то $X_1,...,X_n$, де $X_i=-\frac{1}{\lambda}\ln(1-U_i)$, $i=\overline{1,n}$ – шукана вибірка з експоненційного розподілу.

Якщо функція $F:\mathrm{R}\to[0;1]$ лише не зменшується, то її зворотна функція може не існувати, тоді треба користуватися модифікованим алгоритмом.

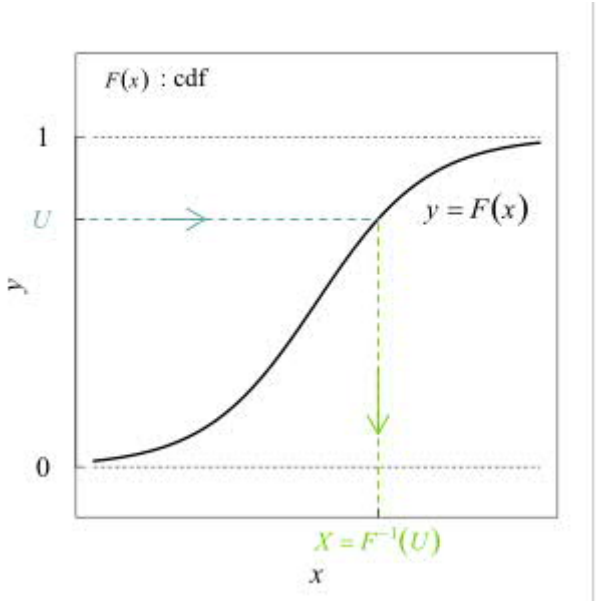

Рисунок 2.1 – Метод зворотного перетворення

Нехай $U_1,...,U_n \sim U[0;1]$ – вибірка з стандартного неперервного рівномірного розподілу. Тоді $X_1,...,X_n$, де $X_i = \inf\{x \mid F(x) = U_i\} = \min\{x \mid F(x) = U_i\}$, $i = \overline{1,n}$ – вибірка шуканого розподілу. Рівність точної нижньої грані мінімуму виконується через неперервність функції розподілу справа, що означає, що точна нижня грань досягається.

Враховуючи це визначення, можна переписати алгоритм для функції, яка строго зростає. Якщо $F(x)$ строго зростає, то $F^{-1}(u) = \inf\{x \mid F(x) = u\}$. Таким чином, модифікований алгоритм для довільної функції розподілу включає в себе окремо розібраний випадок строго зростаючої функції розподілу.

Незважаючи на універсальність, даний алгоритм має серйозні практичні обмеження. Навіть якщо функція розподілу строго зростає, обчислити її зворотню функцію не завжди просто. Для функції розподілу загального вигляду найчастіше необхідно чисельно знаходити точну нижню грань, що є трудомісткою роботою.

Поняття «методу зворотного перетворення» може, поширюватися на поняття «метод перетворення», яке, в загальному, полягає в генеруванні випадкових змінних як функції деяких більш простих випадкових змінних (не обов'язково рівномірних, і тоді функція перетворення не є зворотною).

На рисунку 2.2 приведено приклад методу зворотного перетворення. Синім кольором позначено рівномірний розподіл на [0,1]. Помаранчевим кольором – стандартний гауссовий розподіл. Сірим кольором – відображення від рівномірного до гауссового розподілу.

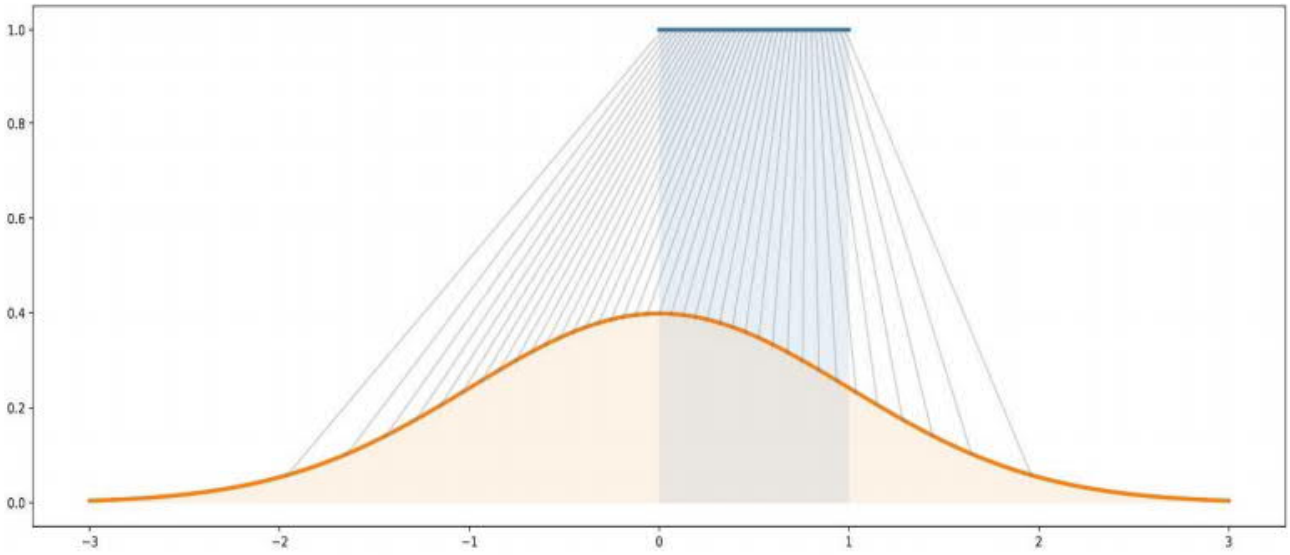

Рисунок 2.2 – Приклад методу зворотного перетворення

2.1.2 Вибірка з відхиленням

Вибірка з відхиленням – метод, який використовується для семплування (управління створенням вибірки) складними імовірнісними розподілами [49].

Метод «вибірка з відхиленням» використовується тоді, коли функція густини розподілу $f(x)$ робить управління складним.

Тоді генерація вибірки за $f(x)$ відбувається за допомогою більш простого допоміжного розподілу $g(x)$, який є простішим, і який задовольняє наступній умові:

$$\forall x \; f(x) < cg(x), \text{ де } c > 1.$$

Алгоритм:

1) взяти вибірку x з розподілу $g(x)$;

2) вибрати випадкове число u рівномірно з відрізка $[0, c\,g(x)]$;

3) обчислити $f(x)$:

– якщо $u \le f(x)$, то x додається до вибірки;

– якщо $u > f(x)$, то x відхиляється.

Алгоритм вибирає точки (x, u) рівномірно з області під графіком $f(x)$, а це і означає що виходять значення вибірки $f(x)$.

Наприклад. Припустимо, ми хочемо вибрати випадкову точку всередині кола одиничного радіуса.

Генеруємо точку (x, y), x і y – незалежні довільні числа з відрізка [–1, 1]. Якщо $x^2 + y^2 \le 1$, то – точка лежить всередині кола, і вона приймається у вибірку. В іншому випадку точка відхиляється, і генерується наступна.

Ефективність цього методу стрімко падає зі збільшенням кількості змінних. Тому потрібно організувати отримання репрезентативних вибірок з вибраного розподілу.

2.1.3 Алгоритм Метрополіса — Гастингса

Розробники цього алгоритму – Ніколас Метрополіс (опублікував в 1953 р.) і К. Гастінгс (узагальнив в 1970 р.) [50]. Алгоритм Метрополіса — Гастингса (Metropolis-Hasting) в основному використовується для складних функцій розподілу. Він частково схожий на алгоритм вибірки з відхиленням, але тут допоміжна функція розподілу змінюється з часом. Окремим випадком даного алгоритму є семплування за Гіббсом. Це простий і швидкий алгоритм, але рідко застосовується.

За алгоритмом Метрополіса — Гастингса можна семплювати будь-яку функцію розподілу. Він ґрунтується на створенні ланцюга Маркова, тобто на кожному кроці алгоритму нове вибране значення x^{t+1} залежить тільки від попереднього x^t. Алгоритм використовує допоміжну функцію розподілу $Q(x' \mid x^t)$, де x^t – змінна, для якої генерувати вибірку просто (наприклад, вона нормально розподілена). На кожному кроці для цієї функції генерується випадкове значення x'. Потім з ймовірністю

$$u = \frac{P(x')Q(x^t \mid x')}{P(x^t)Q(x' \mid x^t)},$$

(якщо $u > 1$, то з ймовірністю 1), x^{t+1} приймається як нове: $x^{t+1} = x'$, інакше залишається старе: $x^{t+1} = x^t$. У випадку, якщо допоміжною функцією є нормальна функція розподілу, то $Q(x' \mid x^t) \sim N(x^t, \sigma^2 I)$. Така функція видає нове значення в залежності від значення на попередньому кроці. Спочатку (до 1970 р.), алгоритм Метрополіса вимагав, щоб допоміжна функція була симетрична: $Q(x' \mid x^t) = Q(x^t, x')$, однак узагальнення Гастингса зняло це обмеження.

Розглянемо алгоритм методу.

Вибираємо випадкове значення x^t. Знаходимо випадкове значення x' для функції $Q(x' \mid x^t)$. Обчислюємо добуток $a = a_1 \cdot a_2$, де

$$a_1 = \frac{P(x')}{P(x^t)},\ a_2 = \frac{Q(x^t \mid x')}{Q(x' \mid x^t)}.$$

Випадкове значення на новому етапі вибирається за правилом:

– якщо $a \geq 1$, то $x^{t+1} = x'$,

інакше,

– якщо $a < 1$, то $x^{t+1} = \begin{cases} x' \text{ з імовірністю } a, \\ x^t \text{ з імовірністю } 1 - a. \end{cases}$

Алгоритм стартує з випадкового значення x^0, і спочатку проганяється «вхолосту» кілька кроків, щоб «забути» початкове значення. Найкраще алгоритм працює тоді, коли форма допоміжної функції близька до форми цільової функції P. Для вирішення цієї проблеми допоміжну функцію налаштовують в ході підготовчої стадії роботи алгоритму. Наприклад, для нормального розподілу налаштовують дисперсію σ^2 так, щоб частка тих випадкових значень, для яких $x^{t+1} = x'$ була близькою до 60%. Якщо дисперсія σ^2 занадто мала, то значення

будуть виходити занадто близькими і частка прийнятих значень буде висока. Якщо дисперсія σ^2 занадто велика, то з великою ймовірністю нові значення будуть знаходитися в зоні малої ймовірності *P*, тому частка прийнятих значень виявиться низькою.

Основна альтернатива всьому цьому полягає в тому, щоб використовувати неявні (implicit) генеративні моделі, в яких ми не намагаємося отримати функцію, підраховуючи щільність потрібного розподілу в кожній точці, а просто моделюємо те, що нам від цієї моделі потрібно. Необхідно вміти породжувати нові точки з цього складного розподілу.

2.2 Алгоритми і методи роботи дискримінатора мережі

Дискримінаційні алгоритми намагаються класифікувати вхідні дані. З огляду на особливості отриманих даних, вони намагаються визначити категорію, до якої вони належать.

Наприклад, дискримінаційний алгоритм може передбачити, перевіривши набір слів в листі, чи повідомлення є спамом чи не спамом. В даному випадку, спам – це категорія, а набір слів, який алгоритм зібрав з електронної пошти – образи, які є вхідними даними. Математично категорії позначають *y*, а образи позначають *x*. Запис $p(y \mid x)$ використовується для позначення «ймовірності *y* при заданому *x*», яка позначає «ймовірність того, що електронний лист є спамом при наявному наборі слів». Дискримінаційні функції зіставляють образи з категорією. Вони зайняті тільки цією кореляцією. Мережа дискримінаторів є стандартною згортковою мережею, яка може класифікувати зображення, що подаються на неї за допомогою бінарного класифікатора, що розпізнає зображення як реальні або як підроблені.

Згорткові нейронні мережі – це окремий вид глибоких нейронних мереж в яких присутні усі базові концепції нейронних мереж. Нововведеннями стали види прихованих шарів (hidden layers). Розглянемо їх.

Шар згортки (Convolutional Layer). Шари згортки отримують на вхід інформацію у матричному вигляді, застосовують фільтри, що змінюються протягом навчання. Фільтри працюють так: підсилюють сигнали, що мають більший вплив, та послаблюють (відрізають) сигнали, які не впливають на дані. Отже, проходячи шари згортки, вхідні дані поступово зменшуються у розмірі та формі після кожного шару, і залишається тільки важлива інформація, яка поступає на наступне "відсіювання". Мовою математики: операція згортки – це поелементне перемноження матриці, та формування за певними правилами нової матриці. Нова матриця містить отриману від перемноження суму. Перша матриця – це вхідна інформація, друга матриця – це фільтр, що "накладається" на інформацію. У кожному шарі нейронної мережі можливо виконати перемноження вхідної інформації на велику кількість різних фільтрів. Усі результуючі матриці після перемноження з'єднуються між собою [51].

Накладання фільтрів здійснюється за алгоритмом:

- заданий фільтр починає рух по матриці вхідних даних з лівого верхнього кута;
- виконується перемноження частини вхідної матриці, що співпадає за розміром з фільтром, на фільтр;
- отримані значення записуються у нову матрицю;
- фільтр пересувається на значення stride (відступ) у праву сторону. Якщо рухатись далі нікуди, то фільтр пересувається на значення відступу вниз і починає цикл по новому з лівої сторони.

У шарах згортки часто задають параметри stride і padding: stride – це відступ, padding – штучне створення навколо вхідної матриці додаткових рядків та колонок з нулями. Приклад роботи операції згортки приведено на рисунку 2.3.

За рахунок проведеної операції, числа, що розташовані по краях вхідної матриці, більше впливають при операції згортки, і не є проігнорованими.

Наступний вид шарів, що існують майже у будь-якій архітектурі згорткових нейронних мереж, це – шари об'єднання (Pooling Layer) [51]. Призначення цих шарів полягає в тому, щоб узагальнити отриману після шарів згортки інформацію та зменшити її розмір (залишити тільки найкориснішу інформацію).

Операція об'єднання відбувається за алгоритмом:

– визначається розмір “вікна”, що рухається по вхідній матриці;

– зліва направо, та згори вниз “вікно” рухається з заданим відступом, та в залежності від типу об’єднання (max pooling чи average pooling) записує дані у результуючу матрицю.

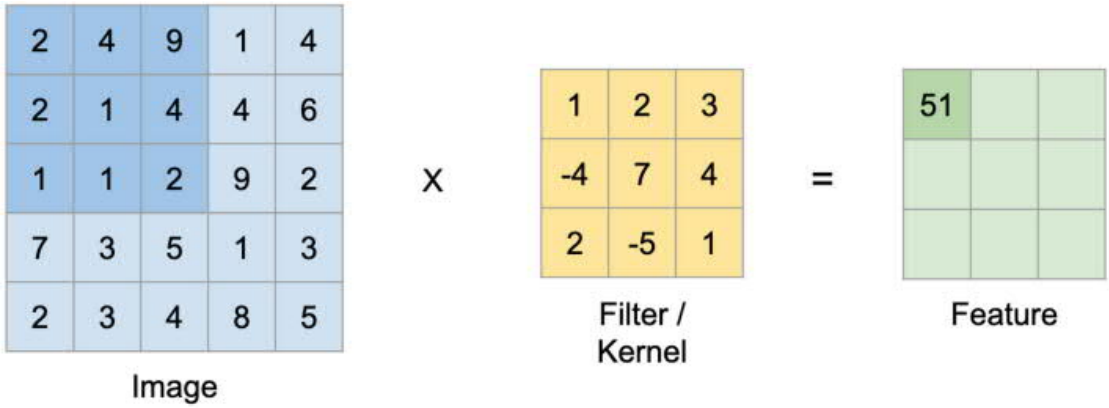

Рисунок 2.3 – Приклад роботи операції згортки

Вид об’єднання шарів max pooling з обраного алгоритмом “вікна” вхідної матриці знаходить максимальний елемент і записує у результуючу матрицю. Таким чином метод підкреслює тільки найбільш високорівневі особливості інформації. Вид об’єднання шарів average pooling рахує середнє арифметичне елементів, та записує у результуючу матрицю. При використанні average pooling є небезпека провести окрім важливої інформації ще й зайві низькорівневі особливості та шуми. Найчастіше на практиці використовується max pooling. Для більшості задач він дає оптимальний результат. Average pooling та інші види об’єднальних шарів використовуються рідше. На рисунку 2.4 приведено приклад роботи обох варіантів.

Після блоків шарів згортки та об’єднання у згорткових мережах використовується операція – повністю з’єднаний шар (Fully Connected Layer) [51]. Для обробки цим шаром, вхідна інформація компресується у вигляді одномірного вектора. Цей вектор є набором даних для одно- чи багатошарового персептрона.

Отриманий вектор представляє собою певні особливості зображення різних рівнів і їх комбінація подається на вхід до повністю з’єднаного шару, який утворює вектор ймовірностей певних подій. Повністю з’єднаний шар використовується у згорткових мережах для знаходження взаємозв’язку між отриманими після згорток та об’єднань набору особливостей, та певною величиною, що репрезентує самі

умови задачі. Для задачі класифікації, необхідно навчити мережу встановлювати відповідності між зображенням, та об’єктом, що на ньому наявний; визначати реальне чи згенероване зображення. Результатом буде вектор з числами від 0 до 1, що позначають ймовірність того, чи той чи інший об’єкт наявний на зображенні або чи зображення є реальним чи згенерованим). Для задачі регресії, окрім класифікації об’єкту, необхідно ще апроксимувати його координати на зображенні і т.д.).

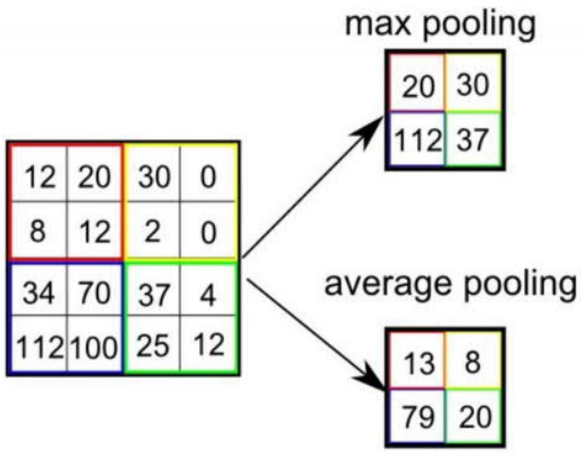

Рисунок 2.4 – Приклад роботи операцій об’єднання шарів max pooling і average pooling

На рисунку 2.5 наведена класична згорткова нейронна мережа.

Для задач сегментації, обробки чи генерації даних, крім наведених операцій під’єднують розгорткову мережу (deconvolutional Neural Network). Вона складається з шарів роз’єднання (unpooling layer) та розгортки (deconvolution layer). Операції у цих шарах за дією є інверсними до операцій об’єднання та згортки.

Шар розгортки, працює обернено до шару згортки. Для отримання матриці більшої розмірності, вхідну матрицю розширюють до певного необхідного розміру, заповнюючи невідому інформацію одиницями. Дальше вступають в дію фільтри. Вони навчаються протягом тренування, завдяки чому стає можливо якісно реконструювати скомпресовану інформацію [52].

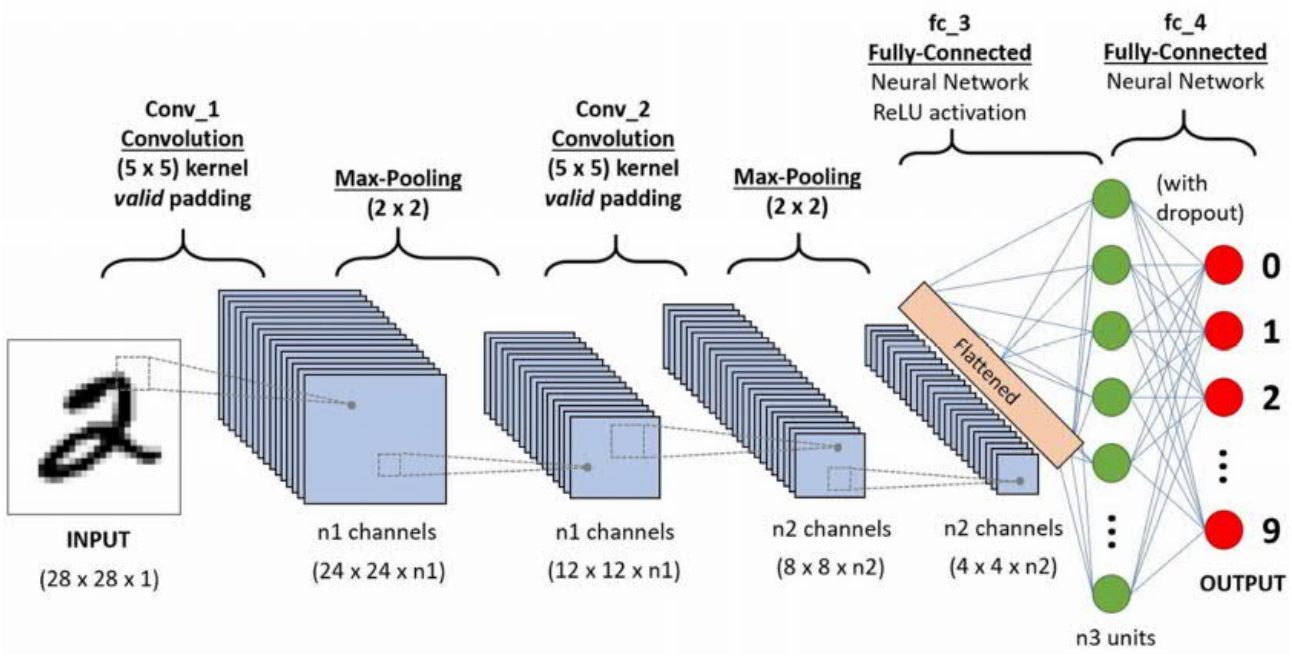

Рисунок 2.5 – Приклад згорткової нейронної мережі

Шар роз’єднання працює обернено до шару об’єднання. На вхід шару поступає скомпресована інформація, даний шар копіює її на карту особливостей більшого розміру. Нагадаємо, що при виконанні операції об’єднання, у нейронній мережі зберігається карта розташування максимальних елементів (утворена за допомогою max pooling), яка передається до шарів роз’єднання. Завдяки цьому проходить якісна реконструкція інформації [52]. На рисунку 2.6 приведено приклад операції роз’єднання.

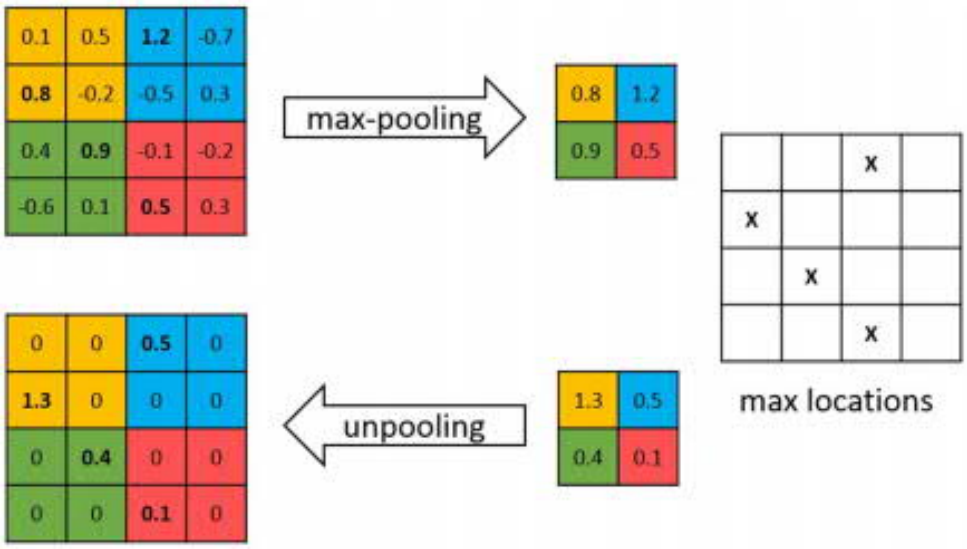

Рисунок 2.6 – Приклад операції роз’єднання

Отже, повна згорткова мережа виконує як згортку так і розгортку вхідних даних, завдяки чому змінює ці дані з середини. Таким прикладом є image2image translation алгоритми. Зміст цих алгоритмів у тому, що при навчанні з учителем (коли присутні як вхідні дані, так і відповідні до них, сподівані результати), ставиться задача перетворити зображення з однієї категорії на зображення з іншої, зі збереженням контексту [52].

В даному підрозділі описано алгоритми роботи дискримінатора, який представляє собою згорткову нейронну мережу.

2.3 Алгоритми і методи роботи генеративно-змагальної мережі

Математично, різницю між дискримінаційними і генеративними моделями можна пояснити так:

– дискримінаційні моделі навчають функцію, яка відображає вхід x в деяку мітку класу y; в імовірнісних термінах це означає, що вони навчають умовний розподіл $p(y \mid x)$;

– генеративні моделі навчають спільний розподіл даних $p(x, y)$; це можна використовувати для того, щоб отримати $p(y \mid x)$, адже для фіксованого x ми отримаємо просто $p(y \mid x) = \frac{p(x, y)}{p(x)}$, але спільний розподіл дає більше інформації, його можна використовувати, наприклад, для породження нових даних.

Важливою відмінністю є також те, що дискримінаційні моделі вирішують тільки задачі навчання з учителем, а генеративні моделі можуть намагатися вирішувати і задачі навчання без учителя, коли міток ніяких немає, і потрібно просто змоделювати розподіл даних $p(x)$. По цій причині в багатьох практичних завданнях часто бувають потрібні саме генеративні моделі.

Основна мета генеративної моделі зазвичай полягає в максимізації функції правдоподібності: для набору даних $D = \{x_i\}_{i=1}^{N}$ максимізувати $\prod_{i=1}^{N} p(x_i; \theta)$ за параметрами моделі, тобто після логарифмування шукати:

$$\theta^* = \arg\max_{\theta} \sum_{i=1}^{N} \log p(x_i; \theta).$$

Максимізація правдоподібності еквівалентна мінімізації відстані Кульбака – Лейблера між розподілом *p*, який виходить з нашої моделі, і розподілом $\hat{p}_{\text{data}}$ – емпіричним розподілом даних. Цей емпіричний розподіл просто повністю зосереджений в точках з dataset і рівномірно розподілений по них, так що:

$$\text{KL}(\hat{p}_{\text{data}}(x), p(x;\theta)) = \int \hat{p}_{\text{data}}(x) \log p(x;\theta) dx = \sum_{i=1}^{N} \hat{p}_{\text{data}}(x_i) \log p(x_i;\theta) dx$$

і мінімізація цього виразу еквівалентна максимізації того, що вище. Інакше кажучи, точки даних, в яких зосереджений розподіл $\hat{p}_{\text{data}}$, «тягнуть вгору» розподіл *p*.

Dataset для машинного навчання – це оброблена і структурована інформація в табличному вигляді. Рядки такої таблиці називаються об'єктами, а стовпці – ознаками.

Генеративні моделі розрізняються тим, як саме вони будують розподіл $p(x;\theta)$. Можна будувати цей розподіл явно, роблячи імовірнісні припущення, які зазвичай зводяться до того, що загальний розподіл $p(x;\theta)$ виражається у вигляді добутку тих чи інших «маленьких» розподілів.

Наприклад, байєсівські мережі довіри будують розподіл з умовних розподілів виду $p(x_i \mid x_{j_1},...,x_{j_k})$ для кожного *i*:

$$p(x) = \prod_{i=1}^{n} p(x_i \mid X_i).$$

У цих моделях імовірнісні припущення про умовні залежності і незалежності між змінними виражаються у вигляді того, які змінні входять в X_i.

Інший підхід до породження складних розподілів, що використовується в

ГЗМ, полягає в тому, щоб почати з простого розподілу $p(z)$ на скриті фактори z і потім застосувати до нього складне перетворення, яке і буде містити всю складність різноманіть.

Деякі методи генерації ймовірнісних розподілів випадкових змінних такого типу ми розглянули в підрозділі 2.1.

В назві ГЗМ закладено, що щось і з чимось змагається. Множина в назві моделі насправді неспроста: у базовому варіанті модель ГЗМ складається з двох штучних нейронних мереж, які змагаються одна з одною. Одна з них, генератор (все пов'язане з генератором позначатимемо літерою g або G), породжує об’єкти в просторі даних, а друга, дискримінатор (про нього говоримо з індексами d або D), вчиться відрізняти породжені генератором об'єкти від справжніх прикладів з навчальної вибірки. Таким чином, виходить, що модель ГЗМ складається з двох частин з протилежними цілями:

– мета дискримінатора – довести, що генератор творить якусь дурницю, навчившись надійно відрізняти породжені генератором приклади від справжніх; інакше кажучи, дискримінатор вирішує найзвичайнішу задачу бінарної класифікації: за заданим прикладом, що виглядає як елемент простору даних, вирішити, чи був він «справжнім» або був породжений генератором;

– мета генератора – «обдурити» дискримінатор, зробити так, що дискримінатор не зможе розрізнити розподіл даних p_{data} і розподіл p_{gen}, який породжує генератор; якби дискримінатор працював ідеально, то ця мета збігалася б з метою навчитися породжувати дані з в точності такого розподілу, як у вхідній вибірці; на практиці виходить не настільки ідеально, але все одно непогано.

Якщо стосовно зображень, то генератор створює нові зображення, які він передає дискримінатору. Він робить це в надії, що вони будуть прийняті справжніми, хоча є підробленими. Мета дискримінатора – визначити, чи є зображення справжнім.

ГЗМ класифікуються в групу «генеративних» моделей. Це означає, що вони здатні створювати, тобто генерувати абсолютно нові, «дійсні» дані. Під

достовірними даними ми маємо на увазі, що вихід мережі повинен бути таким, що ми вважатимемо прийнятним для нашої цілі.

Перед тим як почати будувати дискримінатор і генератор, потрібно зібрати дані і зробити їх попередню обробку.

Яким чином нейронна мережа може працювати генератором, як вона може породжувати об'єкти із заданого розподілу? Тому замість того, щоб генерувати з нуля, нейронна мережа буде перетворювати випадкові входи, згенеровані з деякого «посівного» розподілу, який може бути просто стандартним нормальним розподілом $N(0,1)$ (багатовимірним), чи рівномірним розподілом на [0,1]. Тобто мережа не породжує приклади, а перетворює просту функцію (стандартний нормальний розподіл, рівномірний розподіл) в складну (розподіл даних).

Формально генератор можна записати так:

$$G = G(z;\theta_g) : Z \to X ,$$

де Z – деякий простір прихованих (латентних) факторів, на якому заданий апріорний розподіл $p_z(z)$. А дискримінатор, в свою чергу, виглядає так:

$$D = D(x;\theta_d) : X \to [0;1],$$

Він відображає об'єкти з простору даних в відрізок [0, 1], який інтерпретується як імовірність того, що приклад був дійсно «справжній», з p_{data}, а не згенерований з p_{gen}. Мета дискримінатора полягає в тому, щоб на навчальній вибірці видавати максимальний результат, а на породжених генератором прикладах – мінімальний.

Цільова функція для дискримінатора: видавати свою відповідь на прикладах з p_{data} якомога більшу, а на прикладах з p_{gen} – якомога меншу. Тобто дискримінатору треба максимізувати наступну величину:

$$E_{x\sim p_{\text{data}}(x)}[\log D(x)] + E_{x\sim p_{\text{gen}}(x)}[\log(1-D(x))],$$

де $p_{\text{gen}}(x)$ – це породжений генератором розподіл, $p_{\text{gen}}(x) = G_{z\sim p_z(z)}$.

З іншого боку, генератор повинен навчитися обманювати дискримінатор, тобто мінімізувати по p_{gen} наступну величину:

$$E_{x\sim p_{\text{gen}}(x)}[\log(1-D(x))] = E_{z\sim p_z(z)}[\log(1-D(G(z)))].$$

Якщо тепер об'єднати ці функції в одну, то видно, що дискримінатор і генератор грають між собою в гру, яку в теорії ігор називають мінімаксної грою, вирішуючи таку задачу оптимізації:

$$\min_G \max_D V(G,D) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim p_z(z)}[\log(1-D(G(z)))],$$

де G – генератор, D – дискримінатор, E – математичне сподівання.

Звідси і походить назва моделі. Схематично це можна зобразити так, як показано на рисунку 2.7.

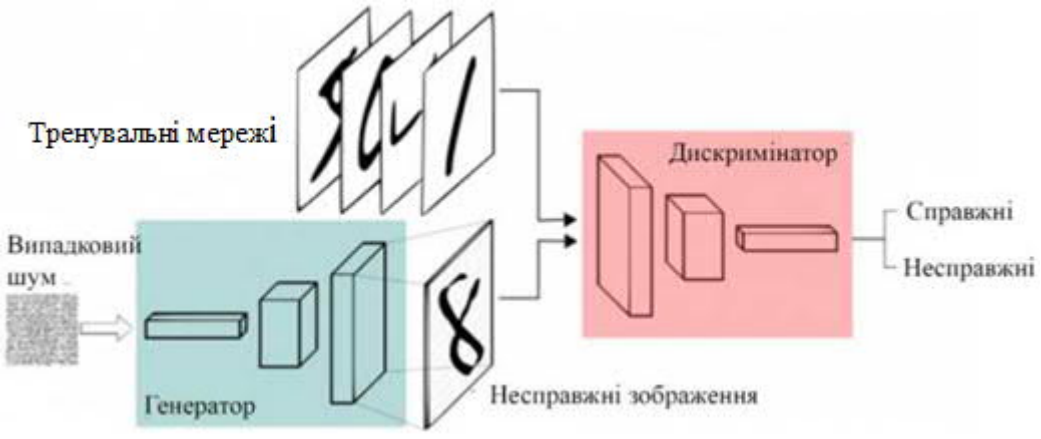

Рисунок 2.7 – Схема роботи ГЗМ

Алгоритм навчання ГЗМ такий: по черзі оновлюються то ваги генератора, то ваги дискримінатора, щоразу рахуючи «противника» фіксованим.

Опишемо алгоритм роботи ГЗМ по кроках:

1) проводиться підготовка реальних даних. Це вихідні зображення, до яких не було застосовано масштабування;

2) генерується шум, на базі якого генератор генерує дані;

3) формується набір даних для навчання дискримінатора. Справжні зображення, які підготовлені на кроці 1, мають імовірність 1 (мітка 1). Зображення, які створені генератором і отримані на кроці 2, мають імовірність 0 (мітка 0);

4) проводиться навчання дискримінатора (навчання генератора відключене), в процесі якого на вхід подаються реальні дані і створені генератором. Дискримінатор видає ймовірності, числа від 0 до 1, причому 1 являє собою справжнє зображення і 0 представляє фальшиве;

5) проводиться навчання генератора (навчання дискримінатора відключене). На вхід подається шум, на виході очікується отримання позначки 1. При навчанні генератора не використовується реальне зображення, а використовується тільки відмітка дискримінатора.

Таким чином, між цими двома ШНМ існує подвійний цикл зворотного зв'язку:

− дискримінатор знаходиться в циклі з справжніми зображеннями;

− генератор знаходиться в циклі разом з дискримінатором.

На початку навчання, генератор не виконує задачу якісно, і візуально згенеровані дані зовсім не є схожими на реальні приклади. Однак за достатнього обсягу даних та добре підібраної функції помилок, за допомогою методу зворотного поширення, шари мережі навчаються перетворювати вхідний шум на інформацію, подібну до реальних даних.

Сформулюємо алгоритм навчання за n ітерацій, за заданих k кроків навчання дискримінатора:

− генеруються нові дані з розподілу − p_{gen};

− випадково вибирається частина даних з p_{data};

− переобчислюються градієнти для дискримінатора;

− генеруються нові дані з розподілу − p_{gen};

− переобчислюються градієнти для генератора.

Глобальною метою навчання таких мереж є прирівняти розподіл p_{gen}, утворений генератором, до розподілу реальних даних p_{data}.

Генератор є оптимальним тоді, коли дискримінатор є максимально «заплутаним» і не може відрізнити реальних і згенерованих зображень.

Теоретично, дискримінатор навчається до оптимального значення, для конкретного генератора, тоді оновлюються параметри генератора. Проте на практиці, дискримінатор може навчатися не до оптимального значення, а протягом деякої кількості ітерацій, тоді генератор оновлюється разом із дискримінатором. Критерієм для дискримінатора може бути рівняння $\max_D \left[\log D(x) + \log(1 - D(G(z)))\right]$, для генератора – $\min_G \log(1 - D(G(z)))$.

Мережа дискримінаторів є стандартною згортковою мережею, яка може класифікувати зображення, що подаються на неї за допомогою бінарного класифікатора, що розпізнає зображення як реальні або як підроблені.

Генератор в певному сенсі є зворотною згортковою мережею: хоча стандартний згортковий класифікатор приймає зображення і зменшує його дозвіл, щоб отримати можливість, генератор приймає вектор випадкового шуму і перетворює його в зображення. Перший відсіває дані за допомогою методів зниження дискретизації, таких як maxpooling, а другий генерує нові дані.

З розвитком штучного інтелекту, кількість мереж для вирішення більш складних задач збільшується (додаються додатково мережі: автокодери (автоенкодери) та декодери, використовуються декілька дискримінаторів для оцінки різних особливостей згенерованих даних, додається генератор для реконструкції даних і т.д.).

Автокодери призначені для кодування вхідних даних в вектори. Вони створюють приховане або стисле представлення необроблених даних. Вони досить корисні при необхідності зменшення розмірності [53].

Автокодери часто пов'язані з декодером. Останній дозволяє відновлювати вхідні дані на основі їх прихованого представлення. Схема їх роботи представлена на рисунку 2.8.

Окрім автокодерів існують ще варіаційні автокодери. Останній є генеративним алгоритмом, який додає додаткове обмеження для кодування вхідних даних, а саме те, що приховані представлення нормалізуються. Варіаційні автокодери дозволяють стискати дані як автокодери і синтезувати дані подібно як робить ГЗМ. Проте, в той час як ГЗМ генерують дані деталізовано, зображення, створені варіаційним автокодером часто є більш розмиті.

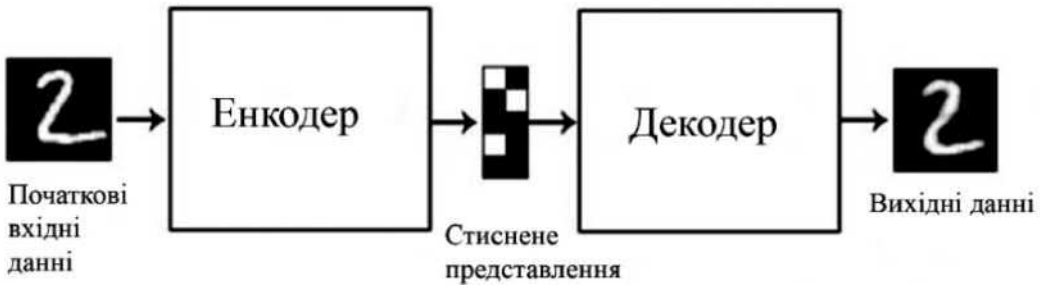

Рисунок 2.8 – Схема роботи енкодера та декодера

Технологія проектування ГЗМ значно розвинулася за допомогою різноманітної оптимізації, додавання нових видів функцій помилок, додаткових шарів у архітектуру, створення абсолютно нових архітектур, та комбінації багатьох генеративних моделей для сумісного тренування. Як відзначалося вище, навчання ГЗМ зводиться до прирівнювання розподілу p_{gen} до p_{data}. Якраз за такими принципами навчаються ГЗМ. За такого підходу є можливість підлаштовувати багато параметрів, та більше фокусувати мережу на усуненні конкретних недоліків. Для ГЗМ властива висока якість навчання, чого нема в звичайних згорткових мережах. Останні застосовують більш прості критерії оцінки, внаслідок чого без внесення власних модифікацій мережі можуть зациклитись на вибраному наборі даних, і не вивчати весь можливий внутрішній розподіл даних. Дискримінатор є кращим та глибшим засобом для оцінки якості роботи мережі, аніж прості функції помилок, що присутні у згорткових мережах. Саме тому, за умови достатньо місткої архітектури для генератора та дискримінатора, правильно налаштованих параметрів, внесення необхідних модифікацій, стає можливим охоплення великої кількість різноманітних даних.

Головним недоліком ГЗМ є те, що якщо мережа натренована не досить якісно і надійно, то мережа буде працювати нестабільно при тренуванні та експлуатації. Для уникнення перенавчання чи поганої якості роботи мереж, необхідно застосувати певний ряд дій та модифікацій та провести детальний аналіз набору даних для навчання.

2.4 Висновки до розділу 2

Отже, в даному розділі отримані такі результати:

1. Описано алгоритми і методи роботи генератора мережі: метод зворотного перетворення, вибірка з відхиленням, алгоритм Метрополіса — Гастингса.
2. Описано алгоритми і методи роботи генератора і дискримінатора.
3. Приведено: алгоритми і методи роботи генеративно-змагальної мережі; алгоритм навчання ГЗМ.
4. Сформульовано алгоритм навчання дискримінатора.
5. Наведена схема роботи ГЗМ.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ГЕНЕРАТОРА ІМУНОГІСТОХІМІЧНИХ ЗОБРАЖЕНЬ

3.1 Вибір програмних засобів та бібліотек

Для програмної реалізації поставленої задачі, необхідно вибрати програмні засоби, мову програмування та бібліотеку для роботи з зображеннями.

На сьогодні існує багато мов програмування для задач машинного навчання. За останніх 5 років найбільшої популярності набрала мова Python.

Відзначимо її переваги над іншими мовами машинного навчання [54].

1. Простота мови, її синтаксису та принципів роботи. Для розробки та тренування власних алгоритмів не потрібно великого досвіду у програмуванні.
2. Це швидкість. Python вважається досить повільною мовою, через те, що код на цій мові не компілюється, а інтерпретується. Однак для нейронних мереж на сьогодні в Python вже існує дуже велика кількість бібліотек та фреймворків. Завдяки їм значно прискорюється процес навчання мереж, проходить оптимізація розподілення даних поміж процесорами GPU, та швидкість роботи з числами.
3. Величезна кількість бібліотек з реалізацією більшості існуючих задач та алгоритмів.

Як і будь-яка мова Python має певні недоліки.

Головним вважається те, що швидкість навчання з застосуванням графічного чіпу буде нижчою, ніж на мові C++.

На другому і третьому місці за популярністю в задачах машинного навчання стоять мови C++ і R.

Але мова C++ не має деяких необхідних бібліотек та є важкою для розробки подібних систем та їх аналізу. Також має обмежені можливості з аналізу коду та детекції помилок.

Мова R, є на сьогодні популярною мовою у сфері статистичного аналізу, розв’язку задач регресії, класифікації та формування дерев рішень [55]. Вона працює з великими даними, має потужні засоби візуалізації інформації, що дозволяє виводити проміжні результати, графіки функцій помилок та ін. параметри. Але мова R як і C++ не має деяких необхідних бібліотек і погано працює з задачами

генерації/редагування зображень.

ПЗ для реалізації архітектури нейронної мереж, проведення її тренування і тестування та інших дій, повинно базуватися на певній бібліотеці машинного навчання – «фреймворку».

На сьогодні існують декілька фреймворків машинного навчання. Вони мають все необхідне для розробки класів нейронних мереж, реалізації базових операцій (згортка, об'єднання, розгортка, роз'єднання), оптимізації мереж при навчанні та т.п.). Звернемо увагу на відмінності між ними.

Фреймворк Tensorflow (розробник Google) [56]. На сьогодні є найбільш популярним. Бібліотека має весь базовий функціонал для поглибленого навчання, засоби для створення нейронних мереж під мобільні платформи (iOS, Android). Але бібліотека вимагає код великого об'єму; її функції мають складний незрозумілий інтуїтивно синтаксис; при тренуванні архітектура створеної мережі є сталою і немає можливості її змінювати протягом навчання. Тому при внесенні незначних змін до архітектури, увесь процес тренування необхідно повторювати з початку.

Бібліотека Keras є обгорткою до бібліотеки TensorFlow [57]. Вона спрощує процес розробки, має більш простий та доступний синтаксис. Даний фреймворк зручний для вирішення простих задач, але не для інноваційних та великих за розміром рішень.

Бібліотека PyTorch є на сьогодні основним засобом для побудови нейронних мереж [58]. Граф підрахунків, на відміну від Tensorflow, є динамічним, тому проводити заново тренування після внесення змін в архітектуру не потрібно. Фреймворк підтримує розподілене (тренування на різних комп'ютерах) та паралельне (на багатьох графічних процесорах одночасно) тренування. Бібліотека має велику кількість раніше натренованих (pre-trained) моделей для виконання різних задач.

До нейронних мереж дані поступають у нормалізованому вигляді (зазвичай числа від 0 до 1) та запаковані в тензор. Тензор – спеціальний тип даних. Для того, щоб перетворити дані до цього вигляду і проводити різні операції над ними, треба використовувати спеціальну бібліотеку для роботи з багатомірними даними, як, наприклад, бібліотека $^{\text{numpy}}$ для мови Python. За допомогою цього фреймворку, вхідні дані обробляються у вигляді $^{\text{numpy}}$ -масивів, нормалізуються та пакуються

у тензори. Аналогічно отримані результати будуть конвертуватися у вигляд, який необхідний для візуалізації.

Для паралельних обчислень існує програмно-апаратна архітектура – CUDA яка дозволяє суттєво прискорити та оптимізувати обчислювальну продуктивність за допомогою графічних процесорів компанії Nvidia (AMD-відеокарти не підтримуються) [49]. Використовується при тренуванні алгоритмів поглибленого навчання, розрахунку статистичних моделей, симуляцій реального світу (дослідження клімату, екології, медицини та ін.). Архітектура CUDA має зручний інтерфейс програмування (CUDA API), виконує більш ефективні транзакції між пам'яттю центрального процесора і відео пам'яттю, надає повну апаратну підтримку цілочисельних та побітових операцій [59], оптимально використовує ресурси комп'ютера. Побудова навчання на даній архітектурі може прискорити увесь процес тренування у 2-4 рази у порівнянні з відеокартою без використання CUDA та у 10-15 разів у порівнянні з проведенням обчислень на CPU.

3.2 Системні вимоги

Для побудови модуля реалізації алгоритмів генерування імуногістохімічних зображень необхідно правильно підібрати програмно-апаратний комплекс для швидкої та безперебійної роботи системи. Характеристики апаратного забезпечення

– процесор з тактовою частотою не менш ніж 2,2ГГц;

– оперативна пам'ять обсягом 8Гб та поколінням пам'яті DDR4 або DDR3 (наприклад, Kingston DDR4-2400 HyperX Fury);

– відеокарта з підтримкою технології CUDA та обсягом пам'яті не менш ніж 4Гб та з поколінням пам'яті не нижче GDDR5 (наприклад, GeForce GTX 1070 8GB);

– жорсткий диск розміром 50Гб.

Також на комп’ютері користувача має бути встановлене наступне програмне забезпечення:

– операційна система Windows 10 x64 або Ubuntu x64;

– середовище Python 3.6;

– середовище Anaconda із встановленим Tensorflow ≥ 1.13;

– остання версія драйверів для відеокарти.

Оптимальний варіант для реалізації поставленого завдання приведено у таблиці 3.1.

Таблиця 3.1 – Характеристики оптимального варіанту комп’ютера

Процесор	2 ядра + 13Гб пам’яті
Диск	100Гб
Графічний процесор	Nvidia Tesla K80, 12Гб

3.3 Структура програмного модулю

Головними компонентами створеного програмного модулю є мережа-генератор та мережа-дискримінатор.

3.3.1 Модуль генерації зображень

У даному модулі відбувається процес обробки вхідного зображення. Мережа-генератор, побудована на архітектурі U-Net (рисунок 3.1). Вона є повною згортковою мережею, має блоки згортки вхідного зображення і розгортки. Розгортка враховує внесену під час обробки допоміжну інформацію з інших гілок модуля [60]. Архітектура мережі: 10 послідовних блоків згортки.

Блоки 1-4 після кожної згортки зменшують вдвічі інформацію у просторовому вимірі та збільшують вдвічі у вимірі особливостей.

Блоки 5-6 у тензор додається додаткова інформація.

Блоки 7-10 – блоки розгортки.

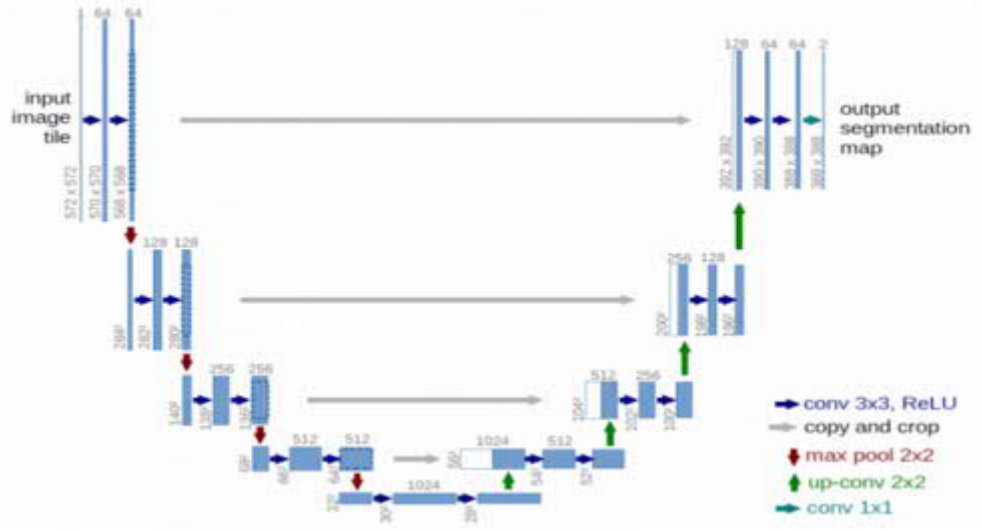

Рисунок 3.1 – Стандартна архітектура мережі U-Net

У мережі U-Net присутні міжшарові поєднання (shortcut connections). Призначення їх – врахування особливостей різних рівнів на всіх етапах обробки і уникнення забування контексту. Міжшарові поєднання передають інформацію про особливості мережі у наступні шари і у симетричні шари розгортки (з 2-го шару згортки інформація передається в 9-й шар розгортки, з 3-го у 8-й і з 4-го у 7-й). Для відвертання інформаційних викидів (коли деякі особливості мають дуже великі чи малі значення, порівняно з іншими особливостями у батчі) у мережі використовується техніка Batch Normalization.

Мережа має шлях звуження зліва та – розширення справа.

Шлях звуження є типовою архітектурою ЗНМ. Блоки цієї під-мережі: дві згортки розміром 3 × 3, після кожного – функція активації ReLU, max-pooling (2 × 2 з відступом 2) для зниження розмірів тензорів. Вкінці звужуючого блоку здійснюється копіювання та відтинання карти особливостей і передача на вхід до блоку розгортки та блоку розширення.

Блок розширення: три шари згортки, після кожного – функція активації ReLU, вкінці – операція підвищення розмірності з ядром розміру 2 × 2. Останнім шаром архітектури є згортка з розміром ядра 1 × 1, яка виконує приведення отриманої карти особливостей до розміру результуючого зображення.

Всього мережа містить 23 шари згортки.

3.3.2 Модуль дискримінатора

Мережа-дискримінатор – це згорткова нейронна мережа, яка бінарно класифікує вхідні дані, та формує результат у вигляді одномірного вектора ймовірностей $\left[p_{gen}, p_{true} \right]$, де p_{gen} – ймовірність того, що вхідне зображення було створено генератором, p_{true} – ймовірність того, що зображення було реальним з набору вхідних даних.

Для побудови мережі-дискримінатора використано архітектуру PatchGAN (рисунок 3.2).

Архітектура PatchGAN [61]. Зображення розбивається на N × N ізольованих сегментів невеликого розміру, для них виконується бінарна класифікація, далі результати класифікації поєднують вигляді одномірного вектора ймовірностей для загального висновку щодо зображення. На вхід поступає одне зображення, розбивається воно на 32 частини, здійснюються окремі передбачення для кожної ізольованої частини зображення, після цього видається результат у вигляді масиву з вірогідностями справжності зображення. Результатом мережі є тензор 32 × 32, який містить ймовірності для кожної з 1024 частин вхідного зображення. Архітектура складається з послідовно з'єднаних шарів згортки, об'єднання. Після кожної згортки застосовується операція нормалізації даних, між шарами виконується функція активації Leaky ReLu.

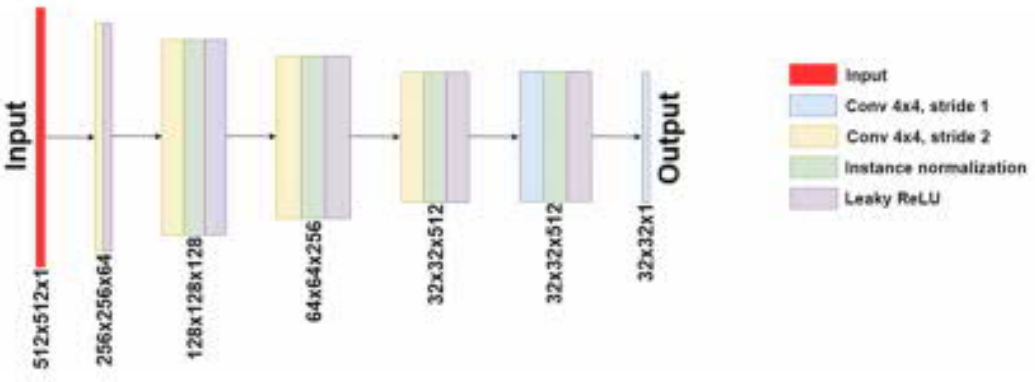

Рисунок 3.2 – Приклад архітектури PatchGAN

Під час тренування, на вхід дискримінатора по черзі подавались справжні зображення (очікуваний результат для яких є масив [1.0, 0.0]), та зображення, згенеровані мережею (очікуваний результат роботи для яких є масив [0.0, 1.0]).

Надалі отримані ймовірності враховувались при розрахунку значення лосс-функції дискримінатора, яка показує наскільки якісно він натренований відрізняти підробки від справжніх фото.

3.3.3 Модуль навчання і тестування

Навчання нейронних мереж – це складний процес, результат якого залежить від:

- вибору та підготовки даних;
- архітектури обраної нейронної мережі;
- правильної логічної структури (правильні послідовності операцій згортки, об’єднання, функції активації);
- гіпер параметрів мережі:
  - розмір батчу (batch size);
  - алгоритм оптимізації градієнтного спуску;
  - кількість епох навчання;
  - коефіцієнт швидкості навчання (learning rate);
  - загасання швидкості навчання (learning rate decay);
  - розмірність вхідних даних;
  - внутрішні параметри функцій помилок, що використовуються.

Розмірність вхідних даних обирається емпіричним шляхом, проте чим вища розмірність (в розумних межах, щоб не виходити за межі відеопам’яті, доступної для навчання) вхідних та вихідних даних, тим більш задовільний результат роботи ми отримуємо у результаті навчання.

Розмір батчу показує, скільки вхідних зображень мережа оброблюватиме за одну ітерацію навчання. Від нього залежить швидкість навчання. Так само як і на розмірність вхідних даних, на цей параметр накладається обмеження стосовно кількості існуючої відеопам’яті.

Learning rate – це параметр навчання, що показує швидкість (міру) навчання мережі за ітерацію. Від нього залежить те, наскільки сильно у сторону шуканого глобального мінімуму будуть змінюватися ваги мережі. Якщо він

занадто малий, то навчання дуже повільне та не може вийти з локальних мінімумів, якщо завеликий – тоді навчання може звестись до якогось локального мінімуму, а пропустити глобальний. Коефіцієнт швидкості навчання обирається емпірично, при навчанні моделей було обрано значення learning rate = 0.001. Дане значення було узято на основі розглянутих схожих рішень для обробки зображень.

Також було саме застосовано learning rate decay – поступове зменшення learning rate з часом (зменшувався кожні 100 ітерацій). Алгоритм Adam був обраним в якості алгоритму оптимізації.

Найголовнішою операцією при навчанні нейронних мереж є градієнтний спуск. Це – метод пошуку оптимального розв'язку поставленої задачі у багатомірному просторі. Градієнтний спуск поступово приводить значення ваг різних шарів нейронної мережі у відповідність до даних, на яких вона навчається. Це робиться для того, щоб шари нейронної мережі та їх фільтри навчалися виявляти та розподіляти між собою певні особливості вхідних даних та інтерпретувати їх необхідним чином, для знаходження якісного результату. Окрім методу градієнтного спуску існують методи градієнтного спуску з певними модифікаціями. Модифіковані методи (стохастичний градієнтний спуск, RMSProp, momentum, Adam) допомагають стабілізувати навчання, уникнути потрапляння у локальний мінімум функції, не обчислювати середнє значення градієнта на усьому наборі даних та ін.

Стохастичний градієнтний спуск модифікує ваги, обчислюючи середнє значення результатів на одному батчі (а не на усьому наборі) вхідних даних [62]. Завдяки модифікації суттєво прискорюється процес навчання, збільшується обсяг даних.

Ваги оновлюються за формулою:

$$w_{t+1} = w_t - \alpha \frac{\partial L}{\partial w_t},$$

RMSProp – метод оптимізації, в якому враховуються попередні зсуви градієнту в ту чи іншу сторону, завдяки діленню learning rate на експоненціальну ковзну середню. Навчання у напрямку проходить більш адаптивно. Формула обчислення нових ваг наступна:

$$w_{t+1} = w_t - \frac{\alpha}{\sqrt{v_t + \varepsilon}} \frac{\partial L}{\partial w_t},$$

де $\varepsilon = 10^{-6}$ (щоб не було ділення на 0), α – learning rate, $\frac{\partial L}{\partial w_t}$ – поточне значення градієнту, v_t визначається з формули:

$$v_t = \beta v_{t-1} - (1-\beta)\left[\frac{\partial L}{\partial w_t}\right]^2,$$

де β – константа, що показує вплив попереднього значення параметра на наступне.

Алгоритм оптимізації градієнтного спуску momentum теж є адаптивним до градієнтів [62]. Згідно методу проходить акумуляція попередніх значень градієнтів для того, щоб врахувати їх значення для наступних. Моментум визначається за формулою:

$$m_t = \beta m_{t-1} - (1-\beta)\frac{\partial L}{\partial w_t},$$

а ваги обчислюються за наступною формулою:

$$w_{t+1} = w_t - \alpha m_t,$$

де α – learning rate. Зазвичай параметр β обирають рівним 0.9, тоді це сприяє тому, що більшу вагу мають попередні значення моментуму та градієнтів, ніж поточне. Значення ваг змінюються у напрямку спадання заданої функції, обходяться локальні мінімуми, кількість ітерацій для знаходження оптимального розв'язку задачі зменшується.

Adam – метод оптимізації градієнтного спуску, що ґрунтується на ідеях momentum, та RMSProp. Для коригування ваг, тут враховується попередні зсуви градієнтів. Загальна формула для ваг має вигляд:

$$w_{t+1} = w_t - \frac{\alpha}{\sqrt{v_t + \varepsilon}} m_t,$$

m_t та v_t визначаються за формулами:

$$m_t = \beta_1 m_{t-1} - (1-\beta_1)\frac{\partial L}{\partial w_t};\ v_t = \beta_2 v_{t-1} - (1-\beta_2)\left[\frac{\partial L}{\partial w_t}\right]^2.$$

Параметри β_1 та β_2 задаються при ініціалізації і частіше за все мають значення $\beta_1 = 0.9$ та $\beta_2 = 0.999$, а початкові значення m_0 і v_0 – нулі. Тому m_t і v_t зсуваються до нуля. Щоб цього не сталося замість m_t і v_t використовують:

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t},\ \hat{v}_t = \frac{v_t}{1-\beta_2^t}.$$

Досліджено, що алгоритм Adam є дуже надійним та якісним оптимізатором для нейронних мереж, особливо часто він зустрічається при навчання згорткових нейронних мереж та при роботі з зображеннями.

Навчання відбувалося на усьому зібраному наборі даних імуногістохімічних зображень 64×64 пікселі, що розділені на чотири класи: цито-фібро-кістозна мастопатія, цито-кістозна мастопатія, цито-непроліферативна-фібромастопія, цито-непроліферативна мастопатія. Навчання

проходило протягом 6-7 днів. Після навчання ГЗМ мережі було згенеровано приблизно 1600 гістологічних (приблизно 320 для кожного класу) зображень.

Для виконання тренування обраної архітектури було обрано метод оптимізації Adam.

Навчання було поділено на епохи, за одну епоху мережа навчалася на усьому наборі даних. Епоха поділена на ітерації, тому, що є деякі обмеження кількості даних, які за один раз можуть пройти обробку у мережі. З обраним розміром батчу 2, кількість ітерацій у межах однієї епохи становила приблизно 500 000. Тому під час проведення тренування, розмір батчу становив 2, отже за одну ітерацію, мережа оброблювала 2 зображення розміром 64×64 пікселів одночасно.

Одна ітерація навчання мала наступний вигляд:

– підготовка вхідних даних для модуля генерації;

– подання вхідних даних для розрахунку у модулі генерації;

– подання згенерованого та справжнього зображення до модуля дискримінатора;

– підрахунок лосс-функцій та здійснення зворотного поширення;

На рисунках 3.4 і 3.5 приведені графіки значень лосс-функції мережі-генератора та мережі-дискримінатора.

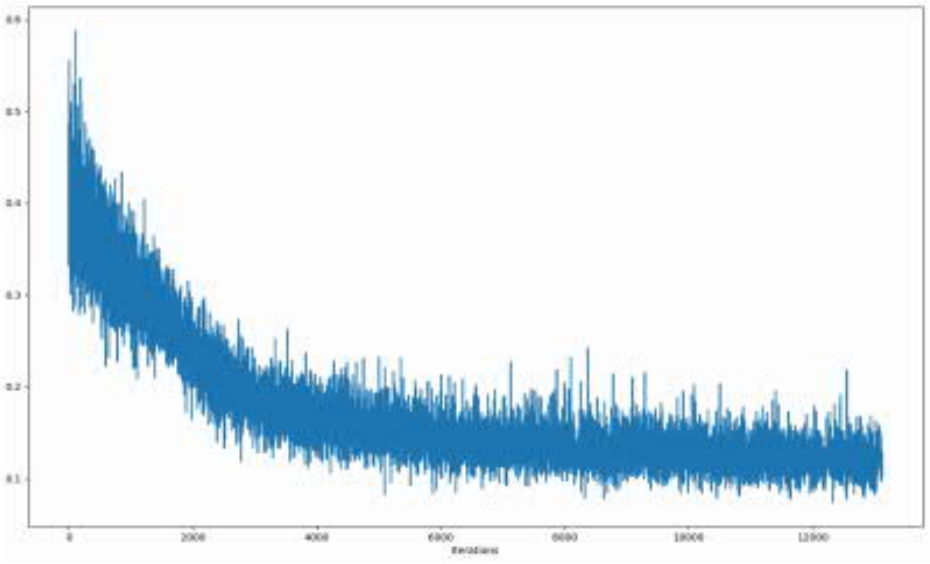

Рисунок 3.4 – Графік значень лосс-функції мережі-генератора протягом навчання

Монотонне спадання значень функції помилок на графіку мережі-генератора (рисунок 3.4), свідчить про те, що мережа успішно навчається виконувати поставлену задачу. Аналогічно, з графіку мережі-дискримінатора видно, що ця мережа також досить успішно навчається.

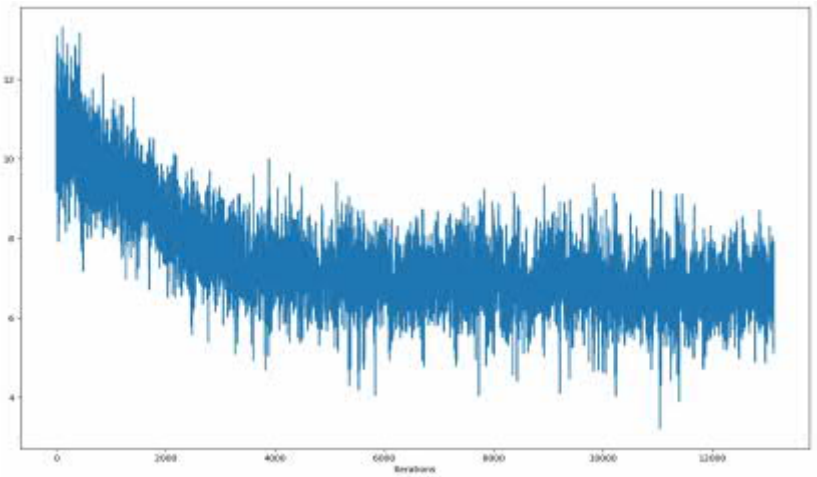

Рисунок 3.4 – Графік лосс-функції мережі-дискримінатора

На рисунку 3.5 приведено навчальну вибірку імуногістохімічних зображень.

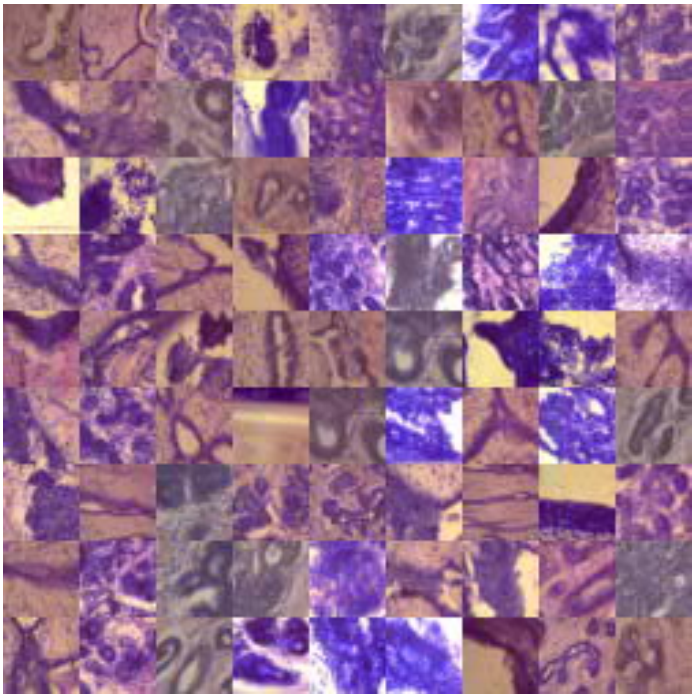

Рисунок 3.5 – Реальна вибірка імуногістохімічних зображень

На рисунку 3.6 приведено розширену згенеровану вибірку.

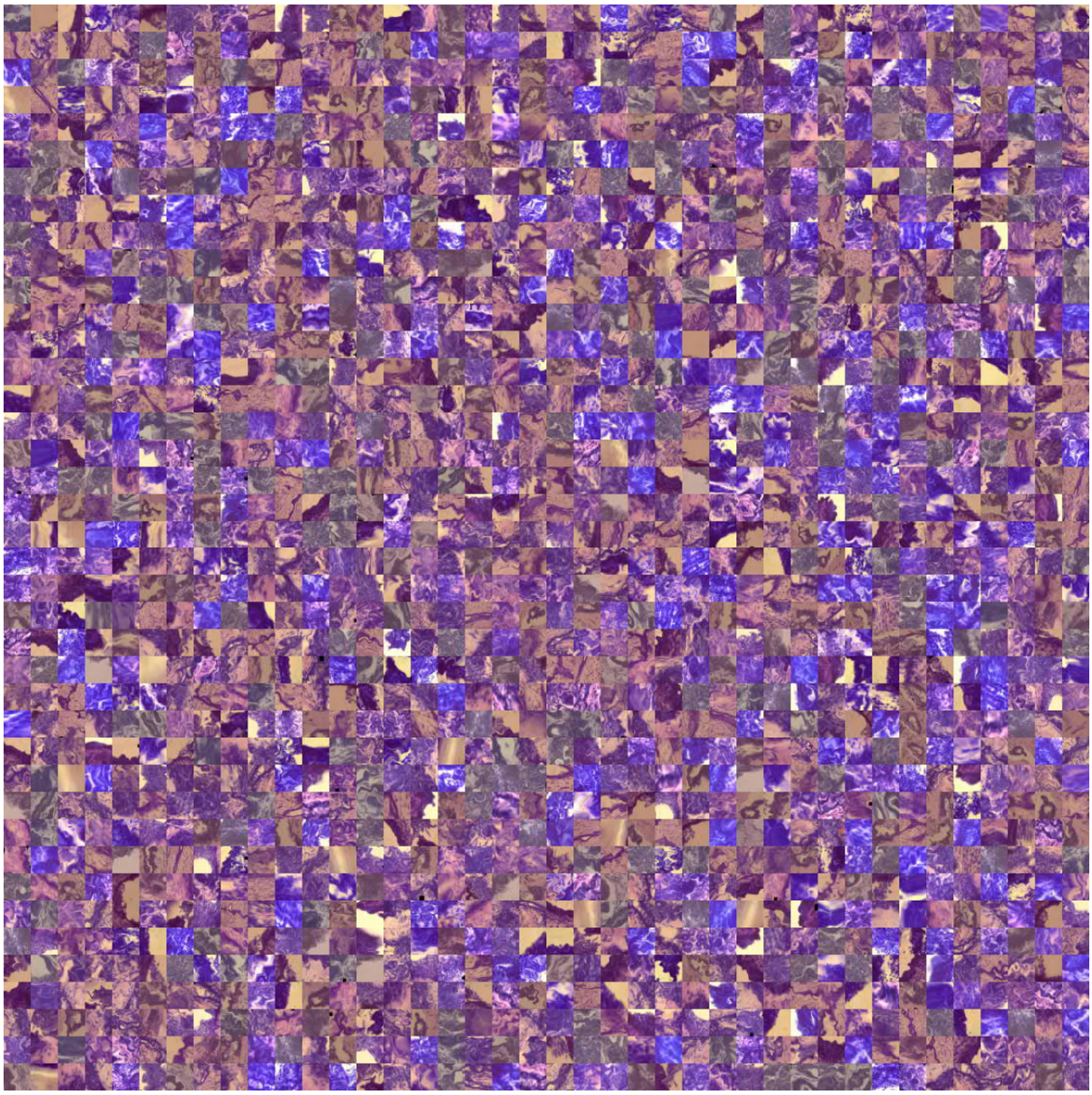

Рисунок 3.6 – Згенерована навчальна вибірка

ROC-криву для класифікатора (імуногістохімія, реальна вибірка) зображено на рисунку 3.7.

ROC-криву для класифікатора (імуногістохімія, згенеровані зображення) зображено на рисунку 3.8.

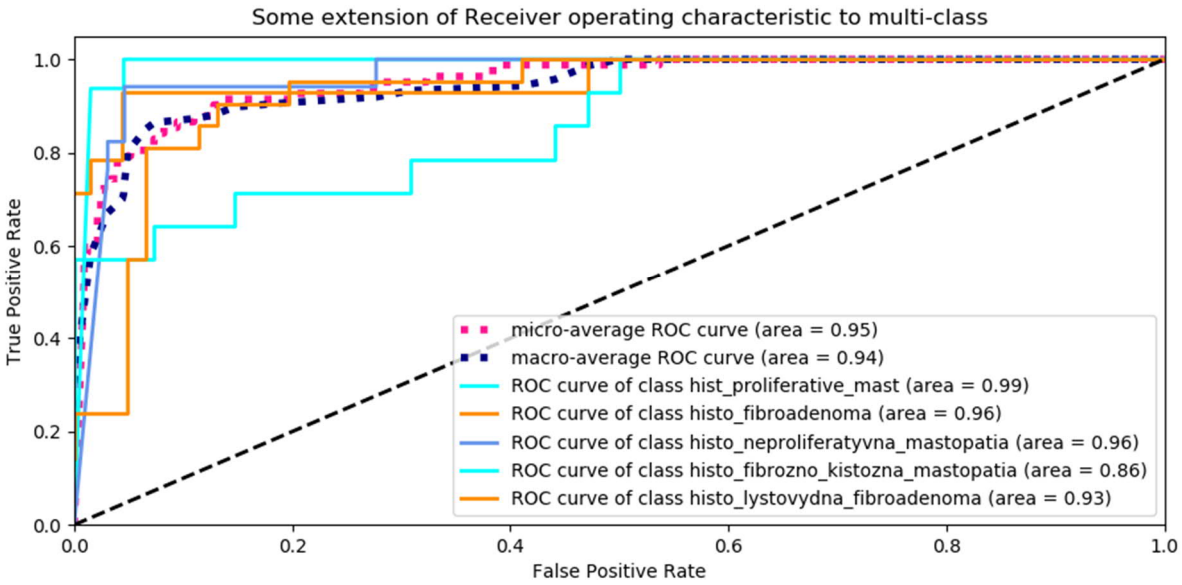

Рисунок 3.7 – ROC-крива класифікатора (реальна вибірка)

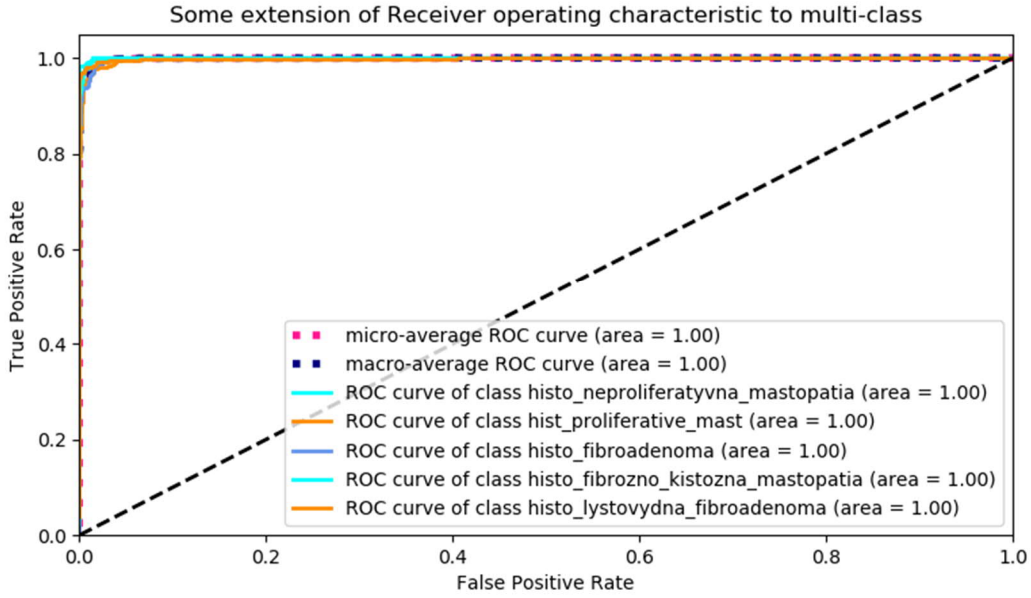

Рисунок 3.8 – ROC-крива класифікатора (згенеровані зображення)

З рисунків бачимо, що згенеровані зображення подібні до оригінальної вибірки. А класифікатор дає правильну мітку зображенням із кожного класу.

3.4 Висновки до розділу 3

У третьому розділі отримано такі результати:

- вибрано програмні засоби та бібліотеки;
- приведено системні вимоги до програмної системи;
- розроблена структура програмного модуля;
- проведені комп’ютерні експерименти.

ВИСНОВКИ

У кваліфікаційній роботі отримано такі результати:

1. Проаналізовано об’єкт досліджень – імуногістохімічні зображення.
2. Зроблено аналіз методів та алгоритмів синтезу зображень.
3. Проаналізовано програмні засоби синтезу зображень.
4. Зроблено аналіз алгоритмів і методів роботи генератора та дискримінатора мережі;
5. Проаналізовано алгоритми і методи роботи генеративно-змагальної мережі її архітектуру.
6. Програмно реалізовано генератор імуногістохімічних зображень.
7. Проведено тестування реалізованих алгоритмів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Генеративна змагальна мережа: веб-сайт. URL: https://uk.wikipedia.org/wiki/Генеративна_змагальна_мережа.

2. Березький О. М. Інформаційно-аналітична система для дослідження і діагностування пухлинних (ракових) клітин людини на основі аналізу їх зображень / О. М. Березький, Т.В. Дацко, Ю.М Батько // Каталог матеріалів Міжнародного Форуму «Регіони знань: Україна в європейському просторі освіти – науки - інновацій для ревіталізації та процвітання територій», 26-27 березня 2010, м. Тернопіль. – Тернопіль, 2010. – С. 37.

3. Березький О. М. Інформаційно-аналітична система дослідження та діагностування пухлинних клітин на основі аналізу їх зображень / О. М. Березький, Ю.М. Батько, Г.М.Мельник // Вісник Хмельницького національного університету. Технічні науки. – 2008. – №4. – С.33-41.

4. Комп’ютерна програма «Інформаційно-аналітична система для дослідження та діагностування пухлинних (ракових) клітин людини «Morphosys» / Березький О. М., Батько Ю.М., Дацко Т.В., Мельник Г.М.: Свідоцтво про реєстрацію авторського права на твір № 35888 від 30.11.2010 р.

5. Berezsky O. Fuzzy system diagnosing of precancerous and cancerous conditions of the breas / O. Berezsky, S. Verbovyy, L. Dubchak, T. Datsko // Proceedings of the XIth International Scientific and Technical Conference Computer Sciences and Information Technologies (CSIT’2016), Lviv, 6-10 September, 2016. – Lviv, 2016. – P. 200–203.

6. Березький О. М. Нечітка база знань інтелектуальної системи діагностування видів раку молочної залози / О. М. Березький, Г. М. Мельник, К. М. Березька // Вісник Хмельницького національного університету. Технічні науки. – 2013. – №6. – С.284-291.

7. Березький О. М. Інтелектуальна система для діагностування різних форм раку молочної залози на основі аналізу гістологічних і цитологічних зображень / О. М. Березький, Г.М.Мельник, Ю.М. Батько, Т. В. Дацко //

Науковий вісник НЛТУ України: зб. наук.-техн. праць. – Львів: РВВ НЛТУ України. – 2013. – Вип. 23.13. – С. 357-367.

8. Комп’ютерна програма «Інтелектуальна система для діагностування різних форм раку молочної залози на основі аналізу гістологічних та цитологічних зображень «IntelliD» («IntelliD») / Березький О. М., Мельник Г.М., Батько Ю.М., Дацко Т.В., Вальків В.: Свідоцтво про реєстрацію авторського права на твір № 52096 від 11.11.2013 р.

9. Berezsky O. The intelligent system for diagnosing breast cancers based on image analysis / Oleh Berezsky, Tamara Datsko, Sergiy Verbovy // Proceedings of Information Technologies in Innovation Business (ITIB), 7-9 October, 2015, Kharkiv, Ukraine. – P. 27-30.

10. База даних цифрових гістологічних та цитологічних зображень різних форм раку молочної залози «CIFDB» («CIFDB») / Березький О. М., Мельник Г.М., Николюк В.Д., Дацко Т.В.: Свідоцтво про реєстрацію авторського права на твір № 52743 від 23.12.2013 р.

11. Березький О.М., Лящинський П.М., Лящинський П.М., Сухович А.Р., Долинюк Т.М. Синтез біомедичних зображень на основі генеративно-змагальних мереж. Український журнал інформаційних технологій. 2019, т. 1, № 1. С. 35-40.

12. Гарматюк В.Р. Кухарук В.Р. Алгоритми оптимізації структур згорткових нейронних мереж: III Наук.-практ. конф. молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі». 26 листопада 2020 р. Тернопіль, 2020. С. 49.

13. Кухарук В.Р., Гарматюк В. Р. Алгоритми генерування зображень на основі нейронних мереж: III Наук.-практ. конф. молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі». 26 листопада 2020 р. Тернопіль, 2020. С. 50.

14. Неміш В.М., Березька К.М. Алгоритми синтезу структур нейронних мереж: III Наук.-практ. конф. молодих вчених і студентів «Інтелектуальні комп’ютерні системи та мережі». 26 листопада 2020 р. Тернопіль, 2020. С. 48.

15. Методичні рекомендації до виконання дипломної роботи з освітньо-кваліфікаційного рівня «Магістр». Спеціальність «Комп’ютерні системи та

мережі» / О.М. Березький, Л.О. Дубчак, Г.М. Мельник /Під ред. О.М. Березького. Тернопіль: ТНЕУ, 2016. 47 с.

16. Імуногістохімія: веб-сайт. URL: https://uk.wikipedia.org/wiki/Імуногістохімія.

17. Ramos-Vara, JA (2005). Technical Aspects of Immunohistochemistry. Vet Pathol 42 (4): 405–426.

18. Coons AH, Creech HJ, Jones RN: Immunological properties of an antibody containing a fluorescent group. Proc Soc Exp Biol Med 1941; 47: 200–202.

19. Иммуногистохимические методы: Руководство / Ed. by George L. Kumar, Lars Rudbeck.: DAKO / Пер. с англ. под ред. Г. А. Франка и П. Г. Малькова. – М.: ЗАО АМТЕО-М, 2011. – 224 с.

20. Бухвалов И. Б., Бекер В. Иммуногистохимия: основы и методы. Берлин, Гейдельберг: шпрингер, 2010, 153 с.

21. Эллиниди В.Н., Аникеева Н.В. Иммуногистоцитохимия: теория и практика: учебное пособие. СПб.: Политехника-сервис, 2013. 52 с.

22. Комп'ютерна графіка: веб-сайт. URL: https://uk.wikipedia.org/wiki/Комп%27ютерна_графіка.

23. Корриган Дж. Компьютерная графика: Секреты и решения: Пер. с англ. М.: Энтроп, 1995. 352 с.

24. Галузинський Г.П., Гордієнко І.В. Сучасні технологічні засоби обробки інформації: Навч. посібник. К.:КНЕУ, 1998. 224 с.

25. Грицик В. В., Березька К. М., Березький О. М. Моделювання та синтез складних зображень симетричної структури: моногр. Львів: УАД – ДНДІІІ, 2005, 2005. 140 с.

26. Фу К. Структурные методы в распознавании образов: Пер. с англ. М.: Мир, 1977. 320 с.

27. Рвачов В. Л. Теорія R-функцій та деякі її застосування. Київ: Наук. думка, 1982. 552 с.

28. R-функция: веб-сайт. URL: https://ru.wikipedia.org/wiki/R-функция.

29. Максименко-Шейко К.В. R-функции в математическом моделировании геометрических объектов и физических полей: веб-сайт. URL: http://dspace.univer.kharkov.ua/bitstream/123456789/7649/2/Макс-Шейко.pdf.

30. Федер Е. Фракталы: пер. с англ. М.: УРСС: Ленанд, 2014. 256 с.

31. Кроновер Р. М. Фракталы и хаос в динамических системах. М.: Постмаркет, 2000. 352 с.

32. Уэлстид С. Фракталы и вейвлеты для сжатия изображений в действии. М.: Триумф, 2003. 320 с.

33. Синтез изображений с помощью глубоких нейросетей: веб-сайт. URL: https://habr.com/ru/company/yandex/blog/314508/.

34. Методи, алгоритми і програмні засоби опрацювання біомедичних зображень / Березький О. М., Батько Ю.М., Березька К.М. і ін. Тернопіль: Економічна думка, ТНЕУ, 2017. 330 с.

35. Глушаков С. В. и др. Компьютерная графика ; Харьковский ин-т информационных технологий. 3. изд., доп. и перераб. Х.: Фолио, 2006. 512 с.

36. Сидоренко В. М. Інженерна та комп'ютерна графіка: навч. посібник; Державний вищий навчальний заклад "Київський національний економічний ун-т ім. Вадима Гетьмана". К.: КНЕУ, 2007. 329 с.

37. Феличи Д. Типографика: шрифт, верстка, дизайн; пер. с англ. и коммент. С. И. Пономаренко. СПб.: БХВ-Петербург, 2008. 470 с.

38. Евгения Тучкевич: Самоучитель Adobe Illustrator CC. СПб.: BHV, 2019. 384 с.

39. Городенко Л. Системи верстки: Практичний посібник для студентів інститутів і факультетів журналістики та відділень видавничої справи і редагування. К.: Центр Вільної Преси, 2006. 520 с.

40. Безручко В. Т. Презентации PowerPoint. М.: Финансы и статистика, 2016. 112 с.

41. Гвоздак А. П. Створення презентацій у MSO PowerPoint для наукової доповіді. Ч. 1. Створення структури і редагування презентації. Навчально-методичний посібник для самостійної роботи студентів. Дніпро ПДАФКіС, 2019. 76 с.

42. Ракута В. М. Microsoft Office PowerPoint 2007 (2010): навчальний посібник. Чернігів: ЧОІППО ім. К. Д. Ушинського, 2013. 43с.

43. Джамбруно М. Трехмерная графика и анимация; пер. с англ. Е. В. Кикинева [и др.]. 2.изд. М.; СПб.; К.: Издательский дом «Вильямс», 2002. 639 с.

44. Борисенко В. Д., Біднiченко О. Г., Котляр Д. В. Основи побудови об'ємних зображень у середовищі проектування AutoCAD: навч. посіб. для студ. вищ. навч. закл.; Нац. ун-т кораблебудування ім. адмірала Макарова. Миколаїв: НУК, 2012. 334 с.

45. Монтаж (кінематограф): веб-сайт. URL: https://uk.wikipedia.org/wiki/Монтаж_(кінематограф).

46. Система автоматизованого проектування AutoCAD: веб-сайт. URL: https://studopedia.com.ua/1_12826_sistema-avtomatizovanogo-proektuvannya-AutoCAD.html.

47. Бирнз Д. AutoCAD 2012 для чайников; пер. с англ. и ред. А. Г. Сысонюка. М.; СПБ.; К.: Диалектика: Вильямс, 2011. 491 с.

48. Метод обратного преобразования: веб-сайт. URL: https://ru.wikipedia.org/wiki/Метод_обратного_преобразования.

49. Выборка с отклонением: веб-сайт. URL: https://ru.wikipedia.org/wiki/Выборка_с_отклонением.

50. Алгоритм Метрополиса — Гастингса: веб-сайт. URL: https://ru.wikipedia.org/wiki/Алгоритм_Метрополиса_—_Гастингса.

51. Convolutional Neural Networks for Visual Recognition: веб-сайт. URL: https://cs231n.github.io/convolutional-networks/.

52. Image to Image translation: веб-сайт. URL: https://towardsdatascience.com/image-to-image-translation-69c10c18f6ff.

53. Автокодувальник: веб-сайт. URL: https://uk.wikipedia.org/wiki/Автокодувальник.

54. Python Tutorial. Release 3.7.0: веб-сайт. URL: https://bugs.python.org/file47781/Tutorial_EDIT.pdf.

55. R Language: веб-сайт. URL: https://cran.r-project.org/doc/contrib/Paradis-rdebuts_en.pdf.

56. tensorflow.org: веб-сайт. URL: https://www.tensorflow.org/.

57. keras.io: веб-сайт. URL: https://keras.io/.

58. pytorch.org: веб-сайт. URL: https://pytorch.org/.

59. docs.nvidia.com: веб-сайт. URL: https://docs.nvidia.com/cuda/.

60. U-Net: Convolutional Networks for Biomedical Image Segmentation: веб-сайт. URL: https://arxiv.org/pdf/1505.04597.pdf.

61. Image-to-Image Translation with Conditional Adversarial Networks: веб-сайт. URL: https://arxiv.org/pdf/1611.07004.pdf.

62. Gradient Descent Optimization algorithms: веб-сайт. URL: https://towardsdatascience.com/10-gradient-descent-optimisation- algorithms-86989510b5e9.