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

Навчально-науковий інститут прикладного системного аналізу

Кафедра штучного інтелекту

До захисту допущено:

В. о. завідувачки кафедри

_____ Ірина ДЖИГИРЕЙ

«___»_____20__ р.

Дипломна робота

на здобуття ступеня бакалавра

за освітньо-професійною програмою «Системи і методи штучного інтелекту»

спеціальності 122 «Комп'ютерні науки»

на тему: «Інтелектуальна система формування розширеного набору зображень на основі використання нейронної мережі AC-WGAN-GP»

Виконав:
студент IV курсу, групи КІ-11
Агарков Роман Дмитрович _____

Керівник:
професор кафедри ШІ, д.т.н., професор,
Синєглазов Віктор Михайлович _____

Консультант з економічного розділу:
доцент кафедри ЕК ФММ, к.е.н., доцент,
Рощина Надія Василівна _____

Консультант з нормоконтролю:
фахівець першої категорії кафедри ШІ, к.т.н., доцент,
Комариста Богдана Миколаївна _____

Рецензент:
доцент кафедри СП, к.т.н.,
Безносик Олександр Юрійович _____

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

Київ – 2025 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Навчально-науковий інститут прикладного системного аналізу
Кафедра штучного інтелекту

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 122 «Комп'ютерні науки»

Освітньо-професійна програма «Системи і методи штучного інтелекту»

ЗАТВЕРДЖУЮ
В. о. завідувачки кафедри
_____ Ірина ДЖИГИРЕЙ
«15» січня 2025 р.

ЗАВДАННЯ
на дипломну роботу студенту
Агаркову Роману Дмитровичу

1. Тема роботи «Інтелектуальна система формування розширеного набору зображень на основі використання нейронної мережі AC-WGAN-GP», керівник роботи Синєглазов Віктор Михайлович, д.т.н., професор, затверджені наказом по НН ІПСА від «26» травня 2025 р. № 1759-с.

2. Термін подання студентом роботи «09» червня 2025 року.

3. Вихідні дані до роботи: гіперспектральна сцена Indian Pines; мітки класів для кожного пікселю сцени Indian Pines.

4. Зміст роботи: аналіз гіперспектральних зображень та пов’язаних з ними проблем, дослідження принципу роботи та модифікацій генеративно-змагальних мереж, розробка модифікації нейронної мережі AC-WGAN-GP, тренування мережі та оцінка результатів.

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо): схеми архітектури, таблиці та графіки результатів, графіки історії тренування моделі, презентація результатів.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Рощина Надія Василівна, доцент, к. е. н.	03.02.2025	09.06.2025

7. Дата видачі завдання «03» лютого 2025 року.

Календарний план

№ з/п	Назва етапів виконання дипломної роботи	Термін виконання етапів роботи	Примітка
1	Аналіз літератури за темою роботи	21.04.2025	Виконано
2	Підготовка першого розділу	28.04.2025	Виконано
3	Підготовка другого розділу	05.05.2025	Виконано
4	Розробка архітектури мережі	12.05.2025	Виконано
5	Підготовка третього розділу	16.05.2025	Виконано
6	Аналіз результатів	19.05.2025	Виконано
7	Підготовка четвертого розділу	26.05.2025	Виконано
8	Підготовка економічної частини	30.05.2025	Виконано
9	Оформлення дипломної роботи	02.06.2025	Виконано
10	Підготовка презентації доповіді	09.06.2025	Виконано

Студент Роман, АГАРКОВ

Керівник Віктор СИНЄГЛАЗОВ

РЕФЕРАТ

Дипломна робота: 93 с., 32 рис., 9 табл., 26 посилань, додаток.

НЕЙРОННІ МЕРЕЖІ, МАШИННЕ НАВЧАННЯ, ШТУЧНИЙ ІНТЕЛЕКТ, ГЕНЕРАТИВНО-ЗМАГАЛЬНІ МЕРЕЖІ, ГІПЕРСПЕКТРАЛЬНІ ЗОБРАЖЕННЯ, ВАРІАЦІЙНИЙ АВТОКОДУВАЛЬНИК, АУГМЕНТАЦІЯ ДАНИХ.

Об’єкт дослідження – гіперспектральні зображення.

Предмет дослідження – аугментація наборів даних гіперспектральних зображень з використанням генеративно-змагальних мереж.

Мета роботи – дослідити покращення в точності класифікаторів у задачі сегментації гіперспектральних зображень.

Досліджено предметну область: проаналізовано застосування сегментації у агрикультурній сфері, специфіку гіперспектральних зображень, принципи їх обробки та сучасні методи аугментації даних. Також розглянуто доцільність застосування WGAN-GP для генерації синтетичних пікселів. Виконано аналіз існуючих підходів та описано датасет Indian Pines, на якому сформульовано експериментальну задачу з описом використаних метрик для оцінки результату. Розроблено модифіковану архітектуру AC-WGAN-GP із двома режимами входу (PCA та VAE), виконано попередню обробку даних і визначено функції втрат. Натреновано модель, порівняно розподіли згенерованих і реальних даних, продемонстровано приріст точності класифікації та покращення карт сегментації. Складник роботи присвячено функціонально-вартісному аналізу програмного продукту. У висновку підбито підсумки щодо потенціалу генеративних мереж для збагачення гіперспектральних датасетів, окреслено напрями подальшого удосконалення моделі.

ABSTRACT

Bachelor's thesis: 93 p., 32 figures, 9 tables, 26 references, appendix.

NEURAL NETWORKS, MACHINE LEARNING, ARTIFICIAL INTELLIGENCE, GENERATIVE-ADVERSARIAL NETWORKS, HYPERSPECTRAL IMAGES, VARIATIONAL AUTOENCODER, DATA AUGMENTATION.

The object of the study is hyperspectral images.

The subject of research is augmentation of hyperspectral image datasets using generative-adversarial networks.

The purpose of the work is to study improvements in the accuracy of classifiers in the task of hyperspectral image segmentation.

The subject area has been explored: the application of segmentation in the agricultural domain, the specifics of hyperspectral images, principles of their processing, and modern data augmentation methods have been analyzed. The feasibility of using WGAN-GP for generating synthetic pixels has also been considered. Existing approaches were analyzed, and the Indian Pines dataset was described, based on which an experimental task was formulated along with the metrics used for result evaluation. A modified AC-WGAN-GP architecture was developed with two input modes (PCA and VAE), data preprocessing was performed, and loss functions were defined. The model was trained, distributions of generated and real data were compared, and improvements in classification accuracy and segmentation maps were demonstrated. A component of the work is dedicated to functional-cost analysis of the software product. The conclusion summarizes the potential of generative networks for enriching hyperspectral datasets and outlines directions for further model improvement.

ЗМІСТ

ВСТУП

Штучний[1] інтелект є однією з найдинамічніших і найперспективніших галузей сучасної науки та техніки. Його розвиток сприяє автоматизації складних завдань, які раніше вимагали значних людських ресурсів та експертних знань. Завдяки методам машинного навчання ШІ знаходить застосування у численних сферах, включаючи медицину, фінанси, промисловість, сільське господарство та багато інших. Зокрема, машинне навчання дозволяє створювати моделі, здатні аналізувати великі обсяги даних, знаходити закономірності та приймати рішення без людського втручання.

Одним із важливих напрямків досліджень у галузі ШІ є генеративні моделі, які використовуються для створення нових даних, схожих на реальні. Вони відіграють головну роль у таких завданнях, як відновлення втрачених даних, підвищення роздільної здатності зображень, а також збагачення датасетів для навчання машинних моделей. Серед генеративних моделей особлива увага приділяється генеративно-змагальним мережам (GAN, Generative Adversarial Networks), запропонованим Яном Ґудфелоу у 2014 році [10].

GAN складаються з двох нейронних мереж: генератора та дискримінатора, які знаходяться у стані постійного змагання. Генератор створює нові зразки даних, в той час як дискримінатор оцінює їхню достовірність, намагаючись відрізнити справжні дані від згенерованих. Ця конкуренція сприяє покращенню якості згенерованих зображень, текстів або інших видів даних.

---

[1]Тут і нижче використані такі інструменти штучного інтелекту як чат-бот з генеративним штучним інтелектом ChatGPT, DeepSeek, виключно для корегування та редагування тексту, створеного автором цієї дипломної роботи, на основі автоматизованої перевірки граматики, структури та стилю, що відповідає Політиці використання штучного інтелекту для академічної діяльності в КПІ ім. Ігоря Сікорського (протокол №11 Вченої ради КПІ ім. Ігоря Сікорського від 11 грудня 2023 р.).

Проте класичні GAN часто стикаються з проблемами нестабільного навчання та низької якості синтетичних даних. Для вирішення цих проблем було розроблено модифікації GAN, серед яких Wasserstein GAN (WGAN) та його вдосконалена версія WGAN-GP. WGAN використовує відстань Вассерштейна як функцію втрат, що забезпечує більш стабільне навчання та високу якість згенерованих зразків. WGAN-GP додатково вводить градієнтний штраф, який ще більше підвищує стійкість алгоритму та забезпечує більш якісну генерацію даних. [14]

У контексті аграрних досліджень гіперспектральні зображення є надзвичайно цінним джерелом даних, оскільки вони дозволяють детально аналізувати спектральні характеристики рослинності, ґрунтів, рівня вологості та інших ключових факторів, що впливають на стан сільськогосподарських угідь. Проте отримання великої кількості якісних гіперспектральних знімків є дорогим, витратним і технічно складним процесом.

Застосування WGAN-GP для генерації синтетичних гіперспектральних зображень відкриває можливість суттєвого розширення наявних датасетів, що, у свою чергу, сприяє покращенню точності моделей автоматизованого моніторингу посівів, класифікації культур, виявлення захворювань і прогнозування врожайності.

Таким чином, використання генеративно-змагальних мереж, зокрема WGAN-GP, для збагачення датасетів у сфері аграрної гіперспектральної зйомки є перспективним напрямом досліджень. Це сприяє розвитку технологій точного землеробства, підвищенню ефективності прийняття рішень в агросекторі та оптимізації управління ресурсами.

РОЗДІЛ 1 АНАЛІЗ ГІПЕРСПЕКТРАЛЬНИХ ЗОБРАЖЕНЬ ТА МЕТОДІВ АУГМЕНТАЦІЇ

Наявні безліч причин, чому фермери шукають стійкі та розумні способи підвищення врожайності та ефективності свого виробництва. До таких причин належать: засухи, дефіцит землі, екологічні проблеми, тощо. Багато з них ведуть до точного землеробства, яке полягає у оптимальному контролі процесу росту за допомогою передових технологій. Однією з таких технологій є гіперспектральна зйомка [12].

Гіперспектральні зображення використовують той факт, що невеликі зміни в рослинності або ґрунті, які будуть невидимими для людського ока та для звичайних камер, можуть бути помітними в їх спектральних сигнатурах. Ці спектральні сигнатури збираються за допомогою гіперспектральних камер та в подальшому можуть аналізуватися за допомогою комп’ютерних технологій, в тому числі – технологій машинного навчання.

1.1 Об’єкт дослідження

Гіперспектральні зображення мають широке коло застосувань в агрикультурі. Одним з напрямів є сегментація гіперспектральних знімків для визначення сільськогосподарських культур. Основною перевагою гіперспектральної зйомки над звичайною є покращене відокремлення схожих об'єктів. Яскравим прикладом в агрикультурі є відокремлення посівів від бур'янів або інших інвазійних видів.

Гіперспектральна сегментація може використовуватися для мапування культур та перевірки, чи правильне насіння було висаджено в правильному місці, а також виявляти будь-які випадкові (ненавмисно посаджені) рослини.

Наприклад, якщо на карті фермерського поля передбачається, що в одній зоні буде кукурудза, а в іншій - боби, гіперспектральне зображення з дрона може чітко розмежувати відповідні культури за їхніми спектральними сигнатурами.

Такий процес набагато простіший, ніж покладання на ручну перевірку, особливо на початку сезону, коли багато молодих культур виглядають однаково. Дослідження показують, що навіть різні сорти однієї сільськогосподарської культури можна розрізнити за їхнім спектральним відбитком за умови наявності достатньої кількості спектральних смуг [26].

Такі можливості гарантують, що плани управління фермою можуть бути адаптовані до фактичної культури, що вирощується в кожній зоні. Наприклад, дрони компанії Gamaya знімають промислові ферми (іноді площею до 25 000 футбольних полів) і створюють детальні карти, які допомагають фермерам «підвищити врожайність, уникнути хвороб і вносити лише воду та поживні речовини, необхідні для отримання щедрого врожаю» [9].

Попри значні переваги гіперспектральних зображень у виявленні змін, непомітних у видимому спектрі, їх використання супроводжується низкою викликів. Головною проблемою використання гіперспектральних зображень є висока вартість та складність їх отримання. Додатково обробка таких даних потребує значної обчислювальної потужності, оскільки кількість каналів часто налічує сотні, в той час як у звичайних зображеннях наявні лише 3. Для вирішення основної проблеми – складності у зборі великої кількості гіперспектральних зображень – використовується техніка аугментації даних.

В цій роботі буде досліджено застосування техніки аугментації гіперспектральних даних для покращення роботи класифікаторів в задачах по-піксельної сегментації гіперспектральних сцен. Сегментація гіперспектральних сцен може застосовуватися для визначення ландшафту, типів ґрунту та видів рослинності на посівах.

1.2 Підходи до аугментації даних

Техніка аугментації даних полягає в створенні різних версій реальних даних для подальшого штучного збільшення розмірності тренувального набору. Такі техніки часто застосовуються для комп’ютерного зору в задачах з недостатньою кількістю даних або незбалансованими класами. Незбалансованість класів в задачах класифікації та сегментації призводить до поганого узагальнювання в моделях машинного навчання. Моделі, що натреновані на таких наборах даних, можуть мати труднощі з точною ідентифікацією прикладів з недостатньо представлених класів.

Аугментація дозволяє збільшити точність моделей машинного навчання завдяки збільшенню кількості тренувальних даних, особливо у класах меншин. Це важливо в тих випадках, коли зібрати велику кількість реальних зображень є складним або дорогим завданням, як у випадку з гіперспектральними зображеннями [7].

Найпоширенішими видами аугментації візуальних даних є геометричні трансформації. До геометричних трансформацій належать обертання, зсуви, масштабування, обрізання, віддзеркалення, тощо. Такі операції дозволяють імітувати варіативність, яка природно виникає у реальному світі – зміни ракурсу або положення об'єкта в кадрі.

Іншим типом аугментації є зміни кольорових характеристик зображення: варіації яскравості, контрасту, насиченості та відтінку. Такі техніки дозволяють покривати іншу варіативність, що виникає у реальному світі під час зміни умов освітлення або використанні іншої камери.

Складнішими типами аугментації є такі методи, як: CutMix, Mixup, Random Erasing, накладання шуму, тощо. Застосування таких технік особливо актуальне в ситуаціях, коли класифікація або сегментація має бути стійкою до складних фонів, непередбачуваного перекриття чи часткової втрати інформації.

Хоча класичні методи аугментації зображень – такі як повороти, обрізання, масштабування, тощо – є простими у реалізації та ефективними для багатьох задач, вони мають низку обмежень. Насамперед, ці трансформації не додають нової семантичної інформації до зображень, а лише варіюють уже наявну. У результаті модель може навчитися бути стійкою до простих змін, але залишатись вразливою до складніших сценаріїв. Більше того, стандартні методи не враховують контекст чи структуру зображення: наприклад, випадкове обрізання може видалити ключові об'єкти або їх частини, що знижує якість навчання. У результаті моделі, натреновані використовуючи виключно класичні види аугментації, можуть мати обмежену здатність до узагальнення.

Пропонується використати генеративні моделі для аугментації зображень як альтернативу традиційним підходам. На відміну від стандартних методів, що оперують простими геометричними або колірними трансформаціями, генеративні моделі здатні створювати нові, семантично багаті зразки, що краще відображають реальну варіативність даних. Такий підхід дозволяє суттєво покращити якість навчального набору, особливо у випадках обмеженого обсягу даних, класового дисбалансу або високої складності задачі. Крім того, такі методи добре масштабуються та можуть бути адаптовані під специфіку доменної області.

1.3 Гіперспектральні зображення та їх ознаки

Звичайні камери, що працюють у три-канальному форматі RGB, здатні захоплювати зображення в діапазоні червоного, зеленого та синього кольорів. Така інформація часто є достатньою для вирішення базових комп’ютерних задач, зокрема – детекції об’єктів, класифікації сцен чи зображень. Однак, у випадках, коли потрібно отримати глибшу інформацію про властивості матеріалів, структуру об'єкта або його спектральні характеристики,

можливостей звичайної RGB-зйомки виявляється недостатньо. У таких випадках доцільно використовувати гіперспектральну зйомку, яка дозволяє фіксувати десятки або навіть сотні спектральних каналів у широкому діапазоні довжин хвиль, що значно підвищує точність аналізу.

Гіперспектральна зйомка збирає та обробляє інформацію з усього електромагнітного спектру. Пікселі отриманого гіперспектрального зображення є векторами значень, кожне з яких відповідає певному діапазону довжини хвилі світла. Зазвичай гіперспектральні зображення налічують більше 100 спектральних каналів. Існує 4 способи отримання гіперспектральних та мультиспектральних зображень, які подані на рисунку 1.1:

- ***просторовий***: полягає у застосуванні push broom та whisk broom сканерів для зчитування спектральної інформації з об'єкту шляхом сканування його поверхні вузькими смугами; кожен двовимірний вихід датчика представляє повний спектр щілини (x, λ); системи лінійного сканування особливо поширені в дистанційному зондуванні, де доцільно використовувати мобільні платформи [5];
- ***спектральний***: полягає у застосуванні band sequential сканерів, які отримують зображення місцевості на різних довжинах хвиль; у цьому методі вся сцена реєструється одночасно, але для кожної окремої довжини хвилі. 2D вихід датчика представляє просторову карту сцени (x, y) для певної довжини хвилі; таке сканування використовує замінні фільтри, в той час як платформа залишається нерухомою; головною проблемою такого підходу є візуальне розмивання, якщо в кадрі відбувається певний рух [5];
- ***миттєвий*** (non-scanning): полягає у застосуванні миттєвих гіперспектральних камер, які дозволяють отримувати повні спектральні зображення всієї сцени за один момент часу, без необхідності поступового сканування. Вихід такого датчика містить усю просторову (x, y) та спектральну λ інформацію. Перевагою

такого підходу є швидкість отримання гіперспектральних зображень [5];

- ***просторово-спектральний***: реєструє всю сцену за раз, де кожен піксель містить лише частину спектру; для отримання повної спектральної інформації потрібен рух камери або сцени.

Кінцевим виходом гіперспектральної зйомки є 3-вимірний куб (x, y, λ), який містить просторову та спектральну інформацію. Такі дані використовуються для аналізу матеріалів та об'єктів, невидимих для людського ока. Інженери розробляють спектральні сенсори та системи обробки даних для застосування в широкому спектрі галузей, таких як:

- агрикультура – використання гіперспектрального дистанційного зондування для моніторингу розвитку та здоров'я рослин; камери HSI також можуть виявляти стрес від важких металів у рослинах, що є швидким і ефективним методом порівняно з хімічними методами після збору врожаю [5];
- сортування та переробка сміття – точне розділення великої кількості різних матеріалів для подальшого їх сортування; ця інформація допомагає автоматизувати процес переробки сміття та підвищити цінність переробленого матеріалу [25];
- спостереження – використання гіперспектральних камер для детекції та ідентифікації небезпек та небезпечних матеріалів; можливе застосування спектральної зйомки для покращеної детекції обличь;
- мінералогія – визначення нових родовищ корисних копалин за допомогою супутникових спектральних знімків;
- навколишнє середовище – моніторинг викидів небезпечних газів, які виділяються вугільними та мазутними електростанціями, сміттєспалювальними заводами, тощо;

- військова сфера - використання спектральних зображень для розвідки, спостереження та виявлення замаскованих або прихованих об’єктів; гіперспектральні сенсори дозволяють ідентифікувати техніку, розрізняти матеріали, виявляти зміни в ландшафті та аналізувати вибухові речовини.

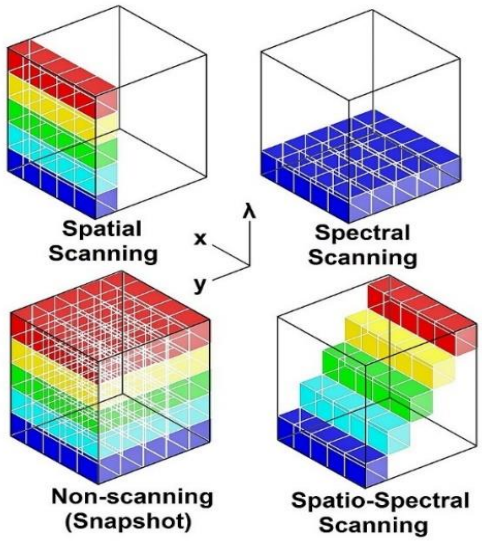

Рисунок 1.1 – Методи отримання спектральних зображень[2]

1.4 Принципи обробки гіперспектральних зображень

Обробка гіперспектральних зображень (HSI) — це складна задача, оскільки таке зображення має тривимірну структуру: дві просторові осі (висота та ширина) і спектральну вісь, яка охоплює сотні діапазонів довжин хвиль. Така кількість інформації створює виклики для зберігання, обробки та навчання моделей, що потребують спеціалізованих методів та архітектур. Основними

[2]Джерело зображення: https://upload.wikimedia.org/wikipedia/commons/thumb/d/d0/AcquisitionTechniques.jpg/250px-AcquisitionTechniques.jpg

підходами до обробки гіперспектральних зображень за допомогою нейронних мереж є 1D, 2D та 3D згортки [18].

1. ***1D згортка*** працює виключно зі спектральною інформацією. Для кожного пікселя береться спектральний вектор, який подається на вхід згорткової мережі. Це дає змогу виявляти спектральні залежності й використовувати HSI для задач класифікації або сегментації на рівні пікселя. При цьому просторовий контекст ігнорується, що може знижувати точність для неоднорідних ділянок зображення.
2. ***2D згортка*** використовує лише просторову інформацію, наприклад, обробляючи зображення, де кожен спектральний канал подається окремо або після зменшення розмірності (наприклад, через PCA). Такі мережі добре працюють у задачах виявлення об’єктів або класифікації, але не використовують багатство спектральної інформації повністю.
3. ***3D згортка*** поєднує просторовий і спектральний контекст. Для цього зображення розбивають на куби навколо кожного пікселя. До цих кубів застосовують 3D-ядра, які одночасно охоплюють два просторові виміри та спектральний. Це дозволяє моделі вивчати локальні структури в усіх трьох вимірах. Попри високу ефективність, 3D згортки є обчислювально дорогими та потребують великого обсягу оперативної пам’яті, особливо для великих зображень.
4. Крім згорткових мереж, у роботі з гіперспектральними зображеннями активно використовуються інші моделі, такі як ***автоенкодери***, які дозволяють навчитися компактному спектральному представленню, зберігаючи головні ознаки.

1.5 Принцип роботи та проблеми генеративно-змагальних мереж

Генеративно-змагальна мережа (Generative Adversarial Network, GAN) — це архітектура штучної нейронної мережі, яка складається з двох основних компонентів: генератора та дискримінатора. Ці дві мережі змагаються одна з одною в рамках гри з нульовою сумою, що стимулює обидві сторони до покращення своєї роботи з часом. Основна мета GAN — навчити генератор створювати дані, що є настільки подібними до справжніх, що дискримінатор не може відрізнити їх від реальних прикладів [10].

Генератор (G) — це нейронна мережа, яка приймає на вхід вектор випадкового шуму (зазвичай з нормального або рівномірного розподілу) та перетворює його на згенерований приклад, який імітує дані з реального розподілу. Метою генератора є "обманути" дискримінатор, змушуючи його класифікувати згенеровані приклади як справжні. ***Дискримінатор*** (D) — це інша нейронна мережа, яка отримує на вхід як справжні приклади з реального датасету, так і фальшиві приклади від генератора. Завдання дискримінатора — відрізняти справжні дані від згенерованих, тобто класифікувати вхідні зразки як "реальні" або "штучні".

Навчання GAN здійснюється в процесі ітеративної оптимізації функції втрат (1.1): дискримінатор навчається краще розпізнавати підробки, тоді як генератор навчається створювати більш реалістичні зразки, щоб обдурити дискримінатор. Загальну схему генеративно-змагальної мережі представлено на рисунку 1.2. Ідеально збалансована GAN досягає точки, коли генератор створює зразки настільки реалістичні, що дискримінатор більше не здатен їх розрізнити з ймовірністю, що перевищує випадкову (тобто 50%).

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x} \sim p_{data}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \\ + \mathbb{E}_{\boldsymbol{z} \sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))], \quad (1.1)$$

де $p_{data}(\boldsymbol{x})$ – розподіл реальних даних; $D(\boldsymbol{x})$ – оцінка дискримінатора для реальних даних; $p_z(\boldsymbol{z})$ – розподіл латентного простору; $D(G(\boldsymbol{z}))$ – оцінка дискримінатора для згенерованих даних.

Одна з ключових проблем традиційних генеративно-змагальних мереж (GAN) — це колапс режиму (mode collapse), який настає, коли генератор навчається продукувати лише обмежену кількість зразків, які добре "обманюють" дискримінатор. Це суттєво обмежує різноманітність згенерованого контенту. Для вирішення такої проблеми зазвичай застосовуються модифікації мережі, у вигляді використання нової архітектури чи функції втрат. Настання колапсу режиму у стандартній генеративно-змагальній мережі під час створення MNIST зображень показано на рисунку 1.3.

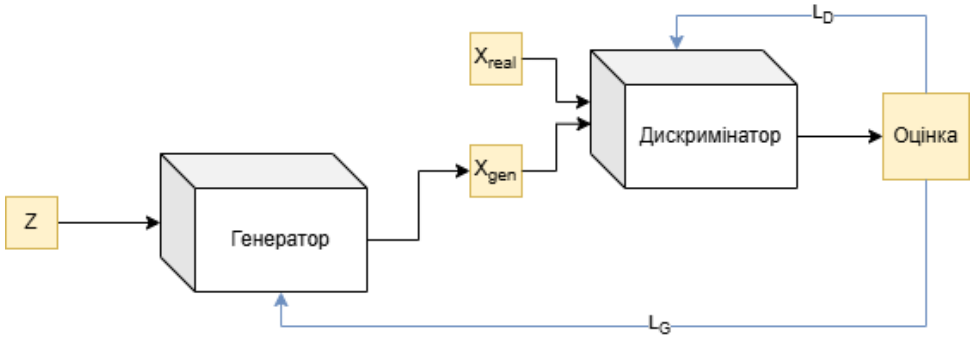

Рисунок 1.2 – Схематичний вигляд генеративно-змагальної мережі

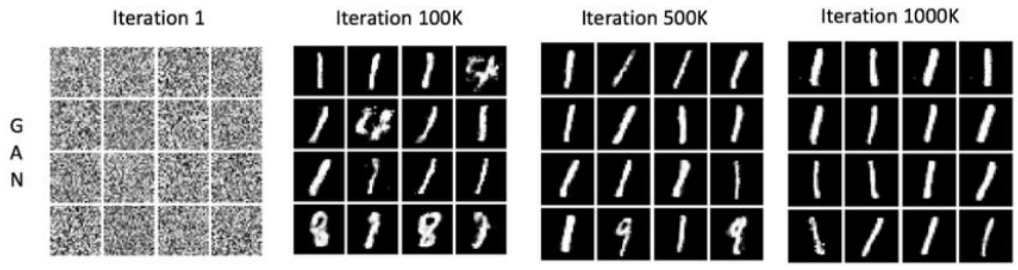

Рисунок 1.3 – Приклад колапсу режиму у GAN[3]

[3] Джерело зображення: https://www.researchgate.net/profile/Lu-Mi-3/publication/329705388/figure/fig2/AS:704668560805888@1545017485759/Generated-images-by-GAN-and-WGAN-models-trained-on-MNIST-after-1-100k-500k-1000k_W640.jpg

Ще одна суттєва проблема у GAN – це падіння (затухання) градієнтів під час навчання. Ця проблема полягає у великому зменшенні значень градієнтів протягом навчання, що призводить до недостатнього зворотного зв'язку від дискримінатора до генератора. Це, в свою чергу, викликає дуже сильне сповільнення або повне припинення навчання генератора. Така проблема зазвичай слабо пов'язана з архітектурою генеративно-змагальної мережі, а більше залежить від обраних функцій втрат та функцій активації. Так, функції, які насичуються, тобто мають місця з дуже малою похідною, можуть сприяти затуханню градієнтів.

1.6 Класифікація генеративно-змагальних мереж

Для вирішення широкого спектра задач, стандартної моделі GAN часто виявляється недостатньо, тому було розроблено безліч модифікацій. Кожна з них спрямована на усунення недоліків та покращення різних аспектів генеративного процесу. Зазвичай це досягається шляхом зміни архітектури або введення нової функції втрат. Найбільш значущі для даного дослідження удосконалення описані далі.

cGAN (Conditional GAN) – модифікація генеративної моделі, в якій генератор, окрім шуму, отримує на вхід умовну інформацію (клас, текст, тощо), яка дозволяє створювати зображення конкретних категорій. Функція втрат cGAN включає в себе умовну інформацію y, але загалом залишається незмінною:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x} \sim p_{data}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \\ + \mathbb{E}_{\boldsymbol{z} \sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))], \quad (1.2)$$

де $\boldsymbol{y}$ – умовна інформація, що подається на входи генератора та дискримінатора.

Архітектура cGAN може варіюватися залежно від конкретного завдання та вибраного способу інтеграції умовної інформації. Вона може включати різні типи нейронних мереж, а також відрізнятися за способом подачі умовних даних – їх можна додавати на вхід як частину латентного вектора або вводити на проміжних шарах генератора та дискримінатора [19]. Загальна схематична архітектура умовної генеративно-змагальної мережі зображена на рисунку 1.4.

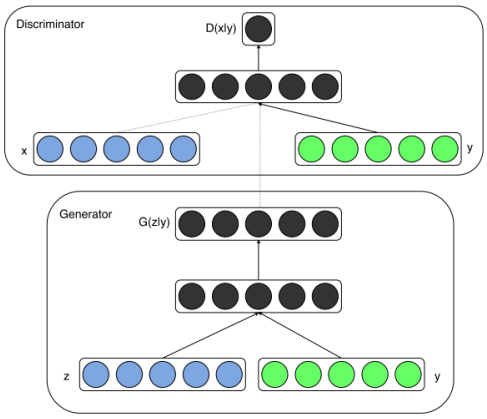

Рисунок 1.4 – Схематична архітектура cGAN [19]

WGAN; WGAN-GP (Wasserstein GAN) замінює стандартну функцію втрат на відстань Вассерштейна, що дозволяє покращити стабільність навчання та забезпечити більш плавне оновлення параметрів нейронної мережі. Функція втрат Вассерштейна записана у формулі (1.3). Її використання дозволяє запобігати головним проблемам тренування класичних генеративно-змагальних мереж: колапсу режиму та зникаючим градієнтам. Для використання такої функції втрат потрібно, щоб норма градієнту дискримінатора була рівна одиниці. У WGAN це досягається шляхом градієнтного відсікання, але такий підхід негативно впливає на стабільність тренування мережі. У WGAN-GP замість цього використовується градієнтний штраф, який додається до функції втрат дискримінатора і обчислюється за формулою (1.4) [3]. На відміну від інших модифікацій, WGAN не встановлює чітких обмежень на архітектуру

генератора та дискримінатора, що дає змогу адаптувати їх до специфіки вхідних даних і цільової задачі.

$$L = E_{\tilde{x}\sim P_g}[D(\tilde{x})] - E_{x\sim P_r}[D(x)], \tag{1.3}$$

$$gp = \lambda E_{\hat{x}\sim P_{\hat{x}}}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2], \tag{1.4}$$

де $\tilde{x}$ – згенеровані дані; x – реальні дані; λ – ваговий коефіцієнт градієнтного штрафу; $\nabla_{\hat{x}} D(\hat{x})$ – вектор градієнту дискримінатора; $\hat{x}$ – лінійно-інтерпольований елемент даних між реальними та фейковими.

VAEGAN (Variational Autoencoder GAN) – додатково використовується варіаційний автокодувальник, який дозволяє краще організовувати латентний простір в порівнянні з звичайним GAN. В такій архітектурі декодувальник та генератор є однією й тією ж мережею. Схематична топологія мережі VAEGAN зображена на рисунку 1.5. Автодкодувальник приймає на вхід елемент даних, стискає його до латентного представлення, після чого з даного латентного представлення отримує синтетичний приклад. Оновлена функція втрат інкорпорує як функцію втрат автокодувальника, так і функцію втрат GAN. Загальна функція втрат визначається за формулою (1.5), а її окремі компоненти описані у формулах (1.6 – 1.8) [4].

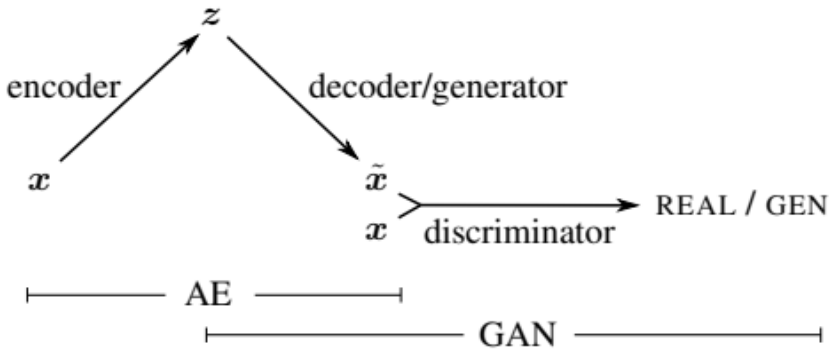

Рисунок 1.5 – Схематичний вигляд VAEGAN [4].

$$L = L_{prior} + L_{llike}^{Dis_l} + L_{GAN}, \tag{1.5}$$

$$L_{prior} = D_{KL}(q(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})), \tag{1.6}$$

$$L_{llike}^{Dis_l} = -E_{q(\boldsymbol{z}|\boldsymbol{x})}[\log p(Dis_l(\boldsymbol{x})|\boldsymbol{z})], \tag{1.7}$$

$$L_{GAN} = \log\big(Dis(\boldsymbol{x})\big) + \log\Big(1 - Dis\big(Dec(\boldsymbol{z})\big)\Big) + \\ + \log(1 - Dis(Dec(Enc(\boldsymbol{x})))), \tag{1.8}$$

де D_{KL} – розбіжність Кульбака-Лейблера; $q(\boldsymbol{z}|\boldsymbol{x})$ – апостеріорний латентний розподіл; $p(\boldsymbol{z})$ – апріорний латентний розподіл; Dis – дискримінатор; Dec, Enc – декодувальник та кодувальник VAE.

1.7 Обґрунтування використання WGAN-GP

Вибір архітектури WGAN-GP для вирішення даної задачі зумовлено двома причинами. По-перше, підвищеною стійкістю даної мережі до основних проблем генеративно-змагальних мереж: колапсу режиму та падінню градієнта. По-друге, можливістю застосування будь-яких архітектур для генератора та дискримінатора.

У WGAN-GP проблема колапсу режиму пом’якшується завдяки використанню відстані Вассерштейна як функції втрат. Порівняння стандартної функції втрат та відстані Вассерштейна зображено на рисунку 1.6. Стандартна функція втрат GAN має дуже низькі значення градієнта в двох випадках, коли згенеровані зображення далекі від реальних, а також коли вони близькі до них, що негативно впливає на процес тренування мережі і може призводити до колапсу режиму, особливо на початку, коли згенеровані зображення ще знаходяться далеко від істинних. На відміну від класичної крос-ентропії, функція втрат Вассерштейна є більш плавною і забезпечує стабільні градієнти для будь-якого аргументу, прибираючи місця, де градієнти наближаються до 0.

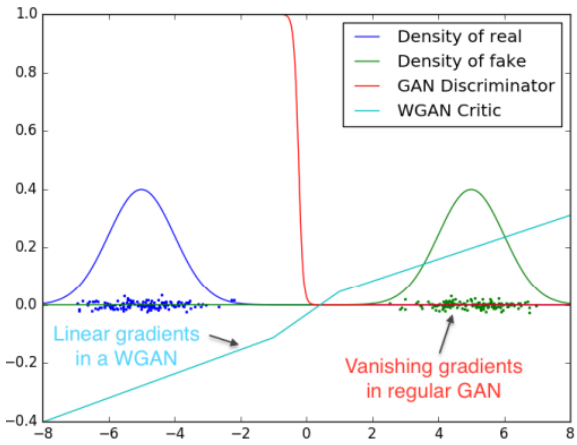

Рисунок 1.6 – Порівняння відстані Вассерштейна з крос-ентропією [3]

Для забезпечення коректності відстані Вассерштейна, дискримінатор повинен бути 1-ліпшицевою функцією, тобто мати норму градієнта, що не перевищує 1. WGAN-GP впроваджує градієнтний штраф, який додає член регуляризації у функцію втрат. Цей член карає дискримінатор у випадку, якщо норма градієнта відхиляється від 1. Зазвичай це реалізується шляхом обчислення градієнта дискримінатора по точках, які лежать на прямій між справжніми та згенерованими зразками. Такий метод забезпечує більш стабільне й надійне навчання, уникаючи падіння градієнтів.

Відсутність жорстких обмежень на архітектуру відкриває широкі можливості для експериментування з глибиною, кількістю параметрів та типами шарів, що дозволяє точно адаптувати модель до особливостей гіперспектральних зображень. Крім того, така гнучкість дає змогу легко інтегрувати елементи з інших модифікацій генеративних мереж, зокрема умовні залежності з cGAN або варіаційне автокодування з VAEGAN, що може суттєво покращити якість генерації та керованість результатів.

Висновки до розділу 1

У цьому розділі розглянуто роль та значення сегментації гіперспектральних зображень у контексті сучасного аграрного виробництва, а також проаналізовано ключові переваги та виклики, пов’язані з їх використанням. Висока спектральна роздільна здатність дозволяє отримувати значно точнішу інформацію, ніж це можливо при використанні традиційної RGB-зйомки.

Водночас, незважаючи на значні переваги, використання гіперспектральної зйомки супроводжується низкою проблем. Насамперед, це висока вартість обладнання та обмежена доступність даних, особливо у відкритих джерелах. Така проблема спричинена складністю у збиранні та обробці великих масивів інформації. Гіперспектральні дані мають високі вимоги до обчислювальних ресурсів і потребують застосування складних алгоритмів для коректної інтерпретації.

У зв’язку з цим особливої актуальності набувають методи оптимізації обробки гіперспектральних зображень, серед яких значну роль відіграє генерація синтетичних даних та застосування алгоритмів машинного навчання. Використання технік аугментації, зокрема на основі генеративних змагальних мереж, дозволяє збільшити обсяг доступного навчального матеріалу, покращити узагальнюючу здатність моделей та зменшити залежність від обмежених датасетів.

Далі виконано аналіз принципів роботи генеративно-змагальних мережі, а також описані ключові проблеми, які можуть виникати під час їхнього навчання: колапс режиму та затухання градієнтів. Після цього було проаналізовано низку модифікацій GAN, серед яких особливу увагу приділено WGAN-GP моделі та доцільності її використання.

РОЗДІЛ 2 ДОСЛІДЖЕННЯ ЗАСТОСУВАННЯ WGAN-GP ДЛЯ ГЕНЕРАЦІЇ ГІПЕРСПЕКТРАЛЬНИХ ЗОБРАЖЕНЬ

2.1 Огляд робіт присвячених WGAN-GP

У цьому пункті представлено аналіз досліджень, які демонструють ефективність WGAN-GP для генерації високоякісних синтетичних зразків у сфері гіперспектральних зображень.

У статті [2] розглянуто застосування WGAN-GP для генерації синтетичних пікселів у галузі гіперспектральних зображень, що зумовлено складністю в отриманні великої кількості тренувальних даних для створення якісного та точного класифікатора. Запропонований фреймворк наведено на рисунку 2.1. Автори пропонують:

1) використання нової архітектури AC-WGAN-GP (Additional Classifier WGAN-GP), яка об’єднує в собі генератор, дискримінатор та додатковий класифікатор;
2) новий механізм онлайн-генерації;
3) алгоритм відбору прикладів, що базується на KNN.

Задача генератора в даній архітектурі – створювати синтетичні гіперспектральні пікселі. Він приймає на вхід гауссовий шум z, умовну інформацію про клас пікселя у вигляді вектору з унітарним кодуванням c, а також вектор з 30 головних компонент X_{pc}, отриманих з реального пікселя методом головних компонент. Всі ці вектори конкатенуються в один, який подається на вхід генератора. Отже, функцію генератора можна записати як: $X_{fake} = G(z, c, X_{pc})$. Архітектура AC-WGAN-GP схематично зображена на рисунку 2.2.

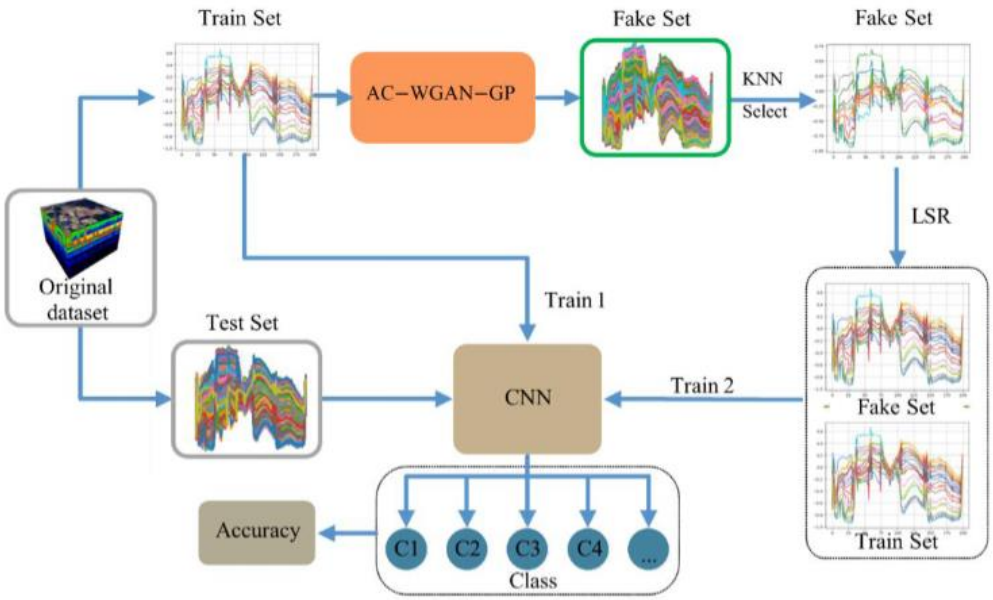

Рисунок 2.1 – Фреймворк мережі AC-WGAN-GP [2]

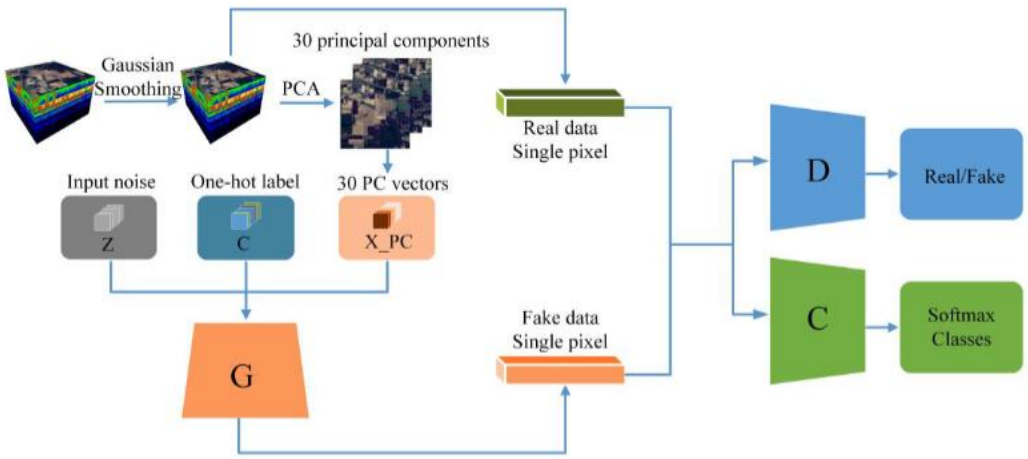

Рисунок 2.2 – Архітектура AC-WGAN-GP [2]

Задача додаткового класифікатора – допомагати генератору зі створенням пікселів з різних класів. Для тренування класифікатора застосовуються лише реальні гіперспектральні пікселі, а вихід класифікатора для згенерованих даних застосовується у процесі тренування генератора. Отже, класифікатор допомагає генератору посилити різницю між синтетичними прикладами з різних категорій (класів).

Оновлена функція втрат для генератора наведена в (2.1), для класифікатора в (2.2). Функція втрат для дискримінатора залишається незмінною зі стандартної WGAN-GP.

$$L_G = -E_{\tilde{x}\sim p(g)}[D(\tilde{x})] - E\left[\log p\left(C = c \middle| X_{fake}\right)\right], \quad (2.1)$$

$$L_C = -E_{x\sim p(x)}[\log P(C = c|X_{real})], \quad (2.2)$$

де $\tilde{x}, X_{fake}$ – згенеровані дані; c – клас пікселю; x, X_{real} – дані з реального розподілу.

Новий механізм онлайн-генерації запропонований у статті [2] полягає у застосуванні генератора для створення та збереження синтетичних прикладів на певних епохах до закінчення тренування. Це дозволяє отримати більшу різноманітність згенерованих прикладів, але може вплинути на їх якість, так як генератор ще не закінчив тренування. Для подолання втрати якості застосовується механізм відбору прикладів, який полягає в наступному:

1. Розділити початковий тренувальний набір на дві множини: (X_{train}, C_{train}), яка застосовується для тренування мережі та X_t, яка використовуватиметься для механізму відбору прикладів.
2. Послідовно використовуючи різні алгоритми кластеризації (DBSCAN, K-means, EM, Mean shift) повторювати кроки, представлені нижче.
   a. Кластеризувати набір X_t у N кластерів.
   b. Обрати центральний елемент кожного кластеру.
   c. Обрати M випадкових елементів з кластеризованої множини.
3. Об’єднати усі отримані центральні та випадкові елементи з кожного алгоритму у одну множину X_u''.
4. Використати KNN для отримання найближчих синтетичних прикладів до кожного елементу $x_u'' \in X_u''$. Ці приклади і будуть відібрані для подальшого застосування в збагаченні датасету.

5. Для кожного обраного прикладу застосувати згладжування міток за такою формулою: $C''_{fake} = C'_{fake}(1-\epsilon) + \frac{\epsilon}{K}$, де $K -$ кількість класів, ϵ – гіперпараметр згладжування.

Переваги.

1. Інтеграція інформації з реальних даних на вхід генератора за допомогою PCA.
2. Контрольоване генерування класів, використовуючи допоміжний класифікатор.
3. Підвищення різноманітності та якості завдяки механізму онлайн-генерації.
4. Висока точність при дуже малому об’ємі даних.

Недоліки.

1. Лінійність методу PCA для зменшення розмірності.
2. Висока залежність від вибору параметрів, таких як: розмірність даних після використання PCA, частота онлайн-генерації, тощо.

У статті [22] автори пропонують використовувати модель WGAN-GP для видобуття просторово-спектральних ознак з гіперспектральних зображень. Застосування WGAN обумовлено можливістю його використання у навчанні без учителя, в той час як більшість інших моделей глибокого навчання потребують розмічених даних. Додатково автори застосовують метод класифікації зображень, що базується на 3D-WGAN-GP.

Перед тим як почати тренування мережі, через велику розмірність даних, до гіперспектральних зображень застосовується PCA (аналіз головних компонент). З отриманих головних компонент залишаються перші 10. Далі, для кожного пікселя вибирається вікно $w \times w$, яке буде використовуватися для тренування мережі замість повного зображення.

Після попередньої обробки даних починається тренування 3D-WGAN-GP. В мережі застосовуються 3D згортки для одночасної роботи як з просторовими,

так і зі спектральними ознаками. Дискримінатор та генератор є повністю згортковими мережами.

Після тренування мережі дискримінатор використовується для видобування ознак із гіперспектральних зображень. Отримані ознаки є входом до класифікатора. Загальний фреймворк зображено на рисунку 2.3.

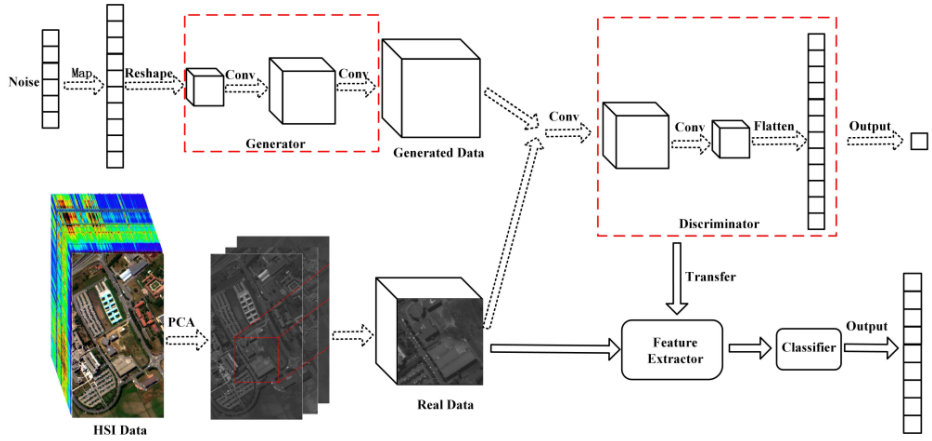

Рисунок 2.3 – Фреймворк мережі для некерованого видобуття ознак [22]

Переваги.

1. Незалежність від розмічених даних (навчання без учителя).
2. 3D-архітектура дозволяє одночасно обробляти спектральні та просторові дані.
3. Зменшення розмірності за допомогою PCA, що спрощує навчання моделі.

Недоліки.

1. 3D-архітектура вимагає великої кількості параметрів та обчислювальних ресурсів.
2. Лінійність методу PCA для зменшення розмірності.
3. Необхідність великого обсягу зразків для тренування.

Автори статті [11] запропонували нову архітектуру змагальної нейронної мережі для генерації міждоменних гіперспектральних даних, яка називається

Hyper-CycleGAN. Запропонований підхід дозволяє збагатити набір даних з одного домену, використовуючи при цьому дані з іншого, що зменшує кількість зображень потрібних для тренування класифікаційних моделей.

Ціль даної мережі – навчитися відображенню з домену-джерела S в цільовий домен T. Для цього використовується пара генераторів: $G_{S\to T}, G_{T\to S}$. Генератори складаються з двох частин: кодувальника та декодувальника, що дає змогу приймати на вхід зображення з одного домену, а на виході – видавати зображення з іншого. Наявність зворотного генератору покращує циклічну узгодженість зображень і забезпечує їх ефективну реконструкцію. Додатково у генераторах застосовується мульти-масштабний механізм уваги, який полягає у використанні MultiHeadAttention блоків на різних масштабах у декодувальній частині генератора. Фреймворк мережі зображено на рисунку 2.4.

Для навчання Hyper-CycleGAN використовуються 3 функції втрат: змагальні втрати, втрати узгодженості циклу та втрати ідентичного відображення. Змагальні втрати відповідають стандартній функції втрат у WGAN-GP і включають два терміни – по одному для кожного з дискримінаторів. Втрати узгодженості циклу полягають у порівнянні результату подвійного перетворення прикладу з одного домену з його оригіналом, формула наведена у (2.3). Втрати ідентичного відображення використовуються для збереження ідентичності цільового класу в мережі, формула наведена у (2.4).

$$L_{cyc}(G_{S\to T}, G_{T\to S}) = E_{s\sim P_S(s)}\left[\left\|G_{T\to S}\left(G_{S\to T}(s)\right) - s\right\|_1\right] + \\ + E_{t\sim P_T(t)}[\left\|G_{S\to T}\left(G_{T\to S}(t)\right) - t\right\|_1, \tag{2.3}$$

$$L_{id}(G_{S\to T}, G_{T\to S}) = E_{s\sim P_S(s)}[\|G_{T\to S}(s) - s\|_1] + \\ + E_{t\sim P_T(t)}[\|G_{S\to T}(t) - 1\|_1], \tag{2.4}$$

де P_S – набір даних, що відповідає домену-джерелу; P_T – набір даних, що відповідає цільовому домену; $G_{T\to S}$ – генератор з цільового домену у домен-джерело; $G_{S\to T}$ – генератор з домену-джерела в цільовий.

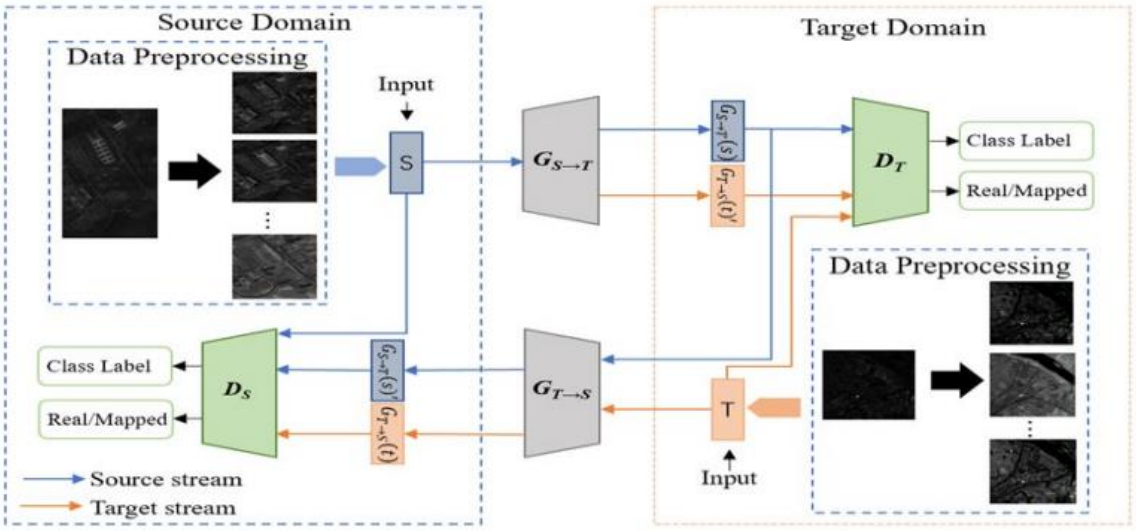

Рисунок 2.4 – Фреймворк мережі для міждоменної генерації [12].

Переваги.

1. Врахування просторово-спектральних взаємозв’язків завдяки використанню механізму уваги.
2. Гнучкість до нових сцен: підхід дозволяє переносити інформацію з одного джерела до цільової сцени.

Недоліки.

1. Низька точність класифікації на межах класів та шумних ділянках зображення.
2. Велика обчислювальна складність через використання декількох генераторів, дискримінаторів та механізму уваги.
3. Неможливість використання на зовсім різних доменах.

Стаття [23] досліджує генерацію синтетичних даних з використанням WGAN-GP моделі для покращення продуктивності моделей глибокого навчання у задачах класифікації зрілості винограду та оцінки вмісту TSS.

Генератор та дискримінатор в даній задачі є багатошаровими персептронами. Обидві мережі додатково приймають на вхід умовну інформацію c, яка відповідає зрілості винограду або його TSS вмісту. Функція

втрат залишається незмінною зі стандартної WGAN-GP моделі. Загальний фреймворк даної мережі представлено на рисунку 2.5.

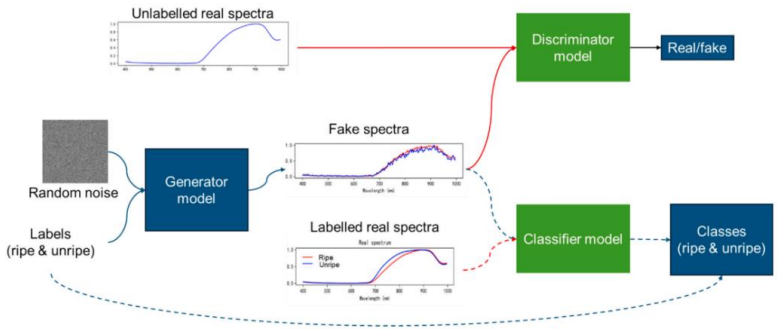

Рисунок 2.5 – Фреймворк мережі для генерації спектрів винограду [23]

Переваги.

1. Контрольоване генерування класів, використовуючи допоміжний класифікатор.
2. Простота в реалізації завдяки використанню звичайних багатошарових персептронів.

Недоліки.

1. Відсутність просторової інформації під час генерації 1D-спектрів для винограду.
2. Використання багатошарових персептронів замість згорткових шарів.

У статті [21] досліджується генерація штучних інфрачервоних та RGB зображень для ідентифікації дикого редису на посівах. Таке застосування пояснюється тим, що традиційні методи виявлення, які базуються на візуальному аналізі або на обробці RGB зображень дають неточні результати через подібність редису до іншої рослинності на посіві.

В цій статті пропонується використовувати стандартну мережу WGAN-GP, де генератор та дискримінатор є 2D-згортковими мережами. Метою роботи

є створення синтетичних зображень редису для збагачення існуючих датасетів і поліпшення точності його виявлення серед інших культур.

Переваги.

1. Простота архітектури запропонованого WGAN-GP.

Недоліки.

1. Відсутність тестування на глибоких нейронних мережах для оцінки поліпшення класифікаційної точності.
2. Наявність великої кількості шуму навіть на найкращих синтетичних зображеннях.

Автори статті [20] вирішували проблему, пов’язану з гіперспектральним розділенням (hyperspectral unmixing) – спектральну варіативність. Гіперспектральне розділення полягає у декомпозиції гіперспектрального пікселю x_p на чисті матеріали A_p та їх частки поширення s_p (abundance fractions).

Проблема спектральної варіативності полягає у тому, що один і той же чистий матеріал може мати різне спектральне представлення в залежності від освітлення, вологості чи сенсорного шуму. Проблемою також є те, що реальні набори даних зазвичай не мають легко вимірюваного ступеня спектральної варіативності. В свою чергу згенеровані дані часто мають нульову спектральну варіативність. Пропонується використати генеративно-змагальну мережу для створення синтетичних гіперспектральних зображень з контрольованою спектральною варіативністю.

Генеративно-змагальна мережа базується на стандартній моделі cWGAN-GP з доданою TV-регуляризацією, яка потрібна для згладжування виходу генератора. Умовний вхід отримується шляхом множення унітарно закодованого вектора, де одиниця знаходиться на позиції максимального значення вектора s_p, на значення «чистоти». Для контролю спектральної варіативності використовуються два параметри: «чистота» та величина усічення нормального розподілу, з якого отримується вхідний вектор шуму.

На виході генератора отримується вектор, що відповідає спектру чистого матеріалу. Для створення одного синтетичного пікселя, генератор комбінує декілька векторів чистих матеріалів для отримання матриці A_p, після чого обчислюється лінійна комбінація: $x_p = A_p s_p$. Наприкінці, істинні спектри чистих матеріалів отримуються як середнє значення великої кількості згенерованих спектрів.

Переваги.

1. Точне налаштування рівню спектральної мінливості, що дозволяє генерувати більш різноманітні дані.
2. Створення реалістичних спектрів, що враховують складні залежності (наприклад, ефекти освітлення).
3. Запобігання зашумленим синтетичним даним завдяки використанню TV-регуляризації.

Недоліки.

1. Відсутність врахування просторових залежностей між пікселями.
2. Пікселі генеруються як лінійна комбінація чистих матеріалів, втрачаючи нелінійні залежності.
3. Складність підбору оптимальних гіперпараметрів.
4. Обов’язкова наявність істинних значень часток поширення для кожного пікселя.

У статті [17] досліджується використання мульти-масштабного cWGAN-GP, названого HSIGAN, для синтезу гіперспектральних 3D-кубів. Такий підхід дозволяє генерувати повноцінні гіперспектральні сцени відразу без потреби по-піксельної генерації та подальшого їх комбінування.

Мережа складається з трьох нейронних підмереж: генератора, дискримінатора та класифікатора. Дискримінатор приймає на вхід гіперспектральні 3D-куби, в той час як класифікатор отримує на вхід одиничні пікселі. Для роботи з 3D даними у генераторі та дискримінаторі

використовуються 3D-згортки, які дозволяють одночасно обробляти спектральну та просторову інформацію.

Для того, щоб зменшити складність роботи з тривимірними даними, усі підмережі працюють на декількох масштабах. Генератор приймає на вхід умовну інформацію разом з вектором шуму, які поступово проходять через декілька підгенераторів, кожен з яких створює гіперспектральний куб різного масштабу. Вихід одного підгенератора стає входом для наступного, що зумовлює поступове збільшення розмірності і полегшує процес генерації. Дискримінатор також розбивається на декілька підмереж, кожна з яких приймає 3D-патч з генератора та реальної сцени на різних рівнях масштабу. На додачу до оцінки реалістичності, дискримінатор видає ймовірності класифікації. Класифікатор отримує на вхід пару гіперспектральних пікселів, що є центрами відповідних патчів на різних масштабах. Виходом класифікатору є ймовірності класифікації для вхідного пікселя. Схематичну топологію мережі зображено на рисунку 2.6.

Як функції втрат у даній мережі використовуються: стандартна функція втрат WGAN-GP та крос-ентропія для втрат пов’язаних з класифікацією. На додачу до цього у генераторі використовується втрата відповідності ознак (feature-matching loss), яка змушує отримані дискримінатором ознаки з синтетичних прикладів бути ближчими до ознак, отриманих з реальних даних.

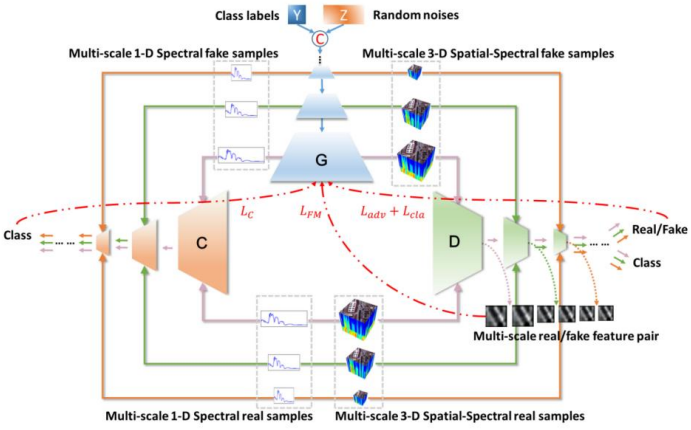

Рисунок 2.6 – Схематична топологія моделі HSIGAN [17]

Переваги.

1. Генерує повноцінні гіперспектральні куби.
2. 3D-архітектура дозволяє одночасно витягати спектральні та просторові ознаки.
3. Використання 3D dropout блоків у генераторі та дискримінаторі для запобігання перенавчанню та колапсу режиму.
4. Контрольоване генерування класів, використовуючи допоміжний класифікатор.

Недоліки.

1. Складна архітектура для навчання, через мульти-масштабний дизайн та 3 підмережі для тренування.
2. Складність підбору оптимальних гіперпараметрів.

У статті [24] досліджується можливість використання WGAN-GP моделі для аугментації наборів гіперспектральних зображень томатів «чері». Збільшення розмірності набору даних обумовлене задачею визначення вмісту розчинних цукрів. Стандартні методи визначення, що ґрунтуються на хімічному складі томату, є точними, але потребують знищення зразка, через що є затратними та повільними. Використання гіперспектральної зйомки дозволяє визначати вміст цукрів, аналізуючи спектральну інформацію без шкоди продукту.

На першому етапі відбувається отримання гіперспектральних зображень томатів та проводиться вимірювання вмісту розчинних цукрів. Після цього для обчислення спектру кожного томату відбувається усереднення значення спектру за зображенням. Для моделювання вмісту розчинних цукрів використовують згорткову нейронну мережу, яка навчається на комбінації справжніх і синтетичних спектральних даних.

Переваги.

1. Простота в реалізації завдяки використанню 1D спектральних векторів.

Недоліки.

1. Відсутність просторової інформації під час генерації 1D-спектрів.
2. Відсутній контроль над згенерованими даними, що може негативно вплинути на баланс класів у результуючому наборі даних.

2.2 Постановка задачі

У дослідженнях, пов'язаних із застосуванням методів глибокого навчання для аналізу гіперспектральних зображень в аграрній сфері, ключовою проблемою є нестача якісних і різноманітних даних. Гіперспектральні датасети зазвичай є обмеженими за обсягом, оскільки їх отримання вимагає спеціалізованого обладнання, тривалого збору даних та обробки великої кількості інформації.

Отже, необхідною є збалансована, повністю маркована вибірка гіперспектральних пікселів, що одночасно задовольняє три ключові метрики: OA, AA та Kappa. Загальна точність (OA) описує пропорцію коректно класифікованих прикладів до всіх прикладів (2.5). Середня точність (AA) описує середній відсоток коректно класифікованих прикладів (2.6). Коефіцієнт Kappa вимірює узгодженість для категоріальних даних (2.7).

$$OA = \frac{\sum_{i=1}^{\bar{C}} h_{ii}}{N}, \tag{2.5}$$

$$AA = \frac{1}{\bar{C}} \sum_{i=1}^{\bar{C}} \frac{h_{ii}}{N_i}, \tag{2.6}$$

$$Kappa = \frac{N \sum_{i=1}^{\bar{C}} h_{ii} - \sum_{i=1}^{\bar{C}} (h_{i+} h_{+i})}{N^2 - \sum_{i=1}^{\bar{C}} (h_{i+} h_{+i})}, \tag{2.7}$$

де $\bar{C}$ – загальна кількість класів (категорій); h_{ii} – кількість прикладів класу i, які класифіковані як клас i; N – загальна кількість розмічених прикладів; N_i –

кількість прикладів, що належать до класу i; h_{i+} – кількість істинних прикладів класу i; h_{+i} – кількість передбачених прикладів класу i.

Маємо вхідну гіперспектральну сцену $X = \{x_{ij} | i = \overline{1,N}, j = \overline{1,M}\}$. Вхідна сцена є матрицею, що складається з гіперспектральних пікселів. Кожен гіперспектральний піксель є вектором значень, отже, X можна записати у вигляді гіперспектрального кубу $X \in R^{H \times W \times C}$, де: H, W – висота та ширина зображення, C – кількість спектральних каналів. Також наявна множина класифікаційних міток Y, що описують, до якого класу належить кожен піксель. $Y \in \mathbb{R}^{H \times W}$ – матриця значень, де кожне значення $y_i \in \{1,2, \dots, K\}$, K – кількість класів на зображенні.

Завдання полягає у вирішенні проблеми недостатньої кількості реальних даних шляхом їх аугментації за допомогою генеративно-змагальної мережі. Для симуляції проблеми недостатності даних, буде обрано певну підмножину реальних пікселів для подальшої роботи. Нехай, $\hat{X}$ – підвибірка множини реальних гіперспектральних пікселів, взята з датасету. $\left|\hat{X}\right| = n \ll N$, де: n – обрана кількість даних для підвибірки, N – загальна кількість даних у датасеті. Для коректності експерименту, потрібно відібрати стратифіковану підвибірку, яка буде репрезентативною для кожного існуючого класу, тобто пропорція елементів для усіх підкласів в підвибірці повинна бути такою ж, що й в оригінальному наборі даних.

Подальшим етапом є побудова (тренування) функції генерації $G: z \in \mathbb{R}^Z \rightarrow \tilde{x} \in \mathbb{R}^C$. Дана функція, роль якої виконуватиме генератор у генеративно-змагальній мережі, що буде описана далі, буде приймати на вхід певну інформацію у вигляді шуму та класової мітки, та видавати відповідний синтетичний гіперспектральний піксель розмірністю в C каналів. Після цього відбуватиметься тестування якості згенерованих прикладів для різних значень співвідношення реальних даних до згенерованих (1:1, 1:2, тощо).

2.3 Огляд набору даних

Для дослідження ефекту збагачення наборів даних синтетичними прикладами буде використаний датасет Indian Pines.

Набір даних Indian Pines був зібраний аналогічним AVIRIS сенсором над випробувальним полігоном Indian Pines у північно-західній Індіані. Дві треті зібраної сцени містять агрикультурні елементи, тобто різні овочеві або зернові культурі, в той час як остання третина містить ліс та іншу природну багаторічну рослинність. Загальна розмірність зображення складає 145×145 пікселів. Спектральна інформація складає 224 рефлективні канали у проміжку $0.4 - 2.5 * 10^{-6}$ метрів. Аналогічно попередньому датасету рекомендовано зменшити кількість каналів до 200, видаляючи ті, що покривають зони сильного поглинання води: 104-108, 150-163, 220. На сцені присутні дві основні автомагістралі, залізнична лінія, а також житлова забудова, інші споруди та менші дороги. Знімок було зроблено в червні, тому деякі з наявних культур – кукурудза, соя – перебувають на ранніх стадіях росту. Наявні дані поділено на шістнадцять класів, не всі з яких є взаємовиключними. Пікселі, що належать до класу заднього фону прибрано з набору даних. Розмітка відображена на рисунку 2.7. Розподіл пікселів за класами наведено у таблиці 2.1 [13].

Таблиця 2.1 – Розподіл пікселів у датасеті Indian Pines

Номер в датасеті	*Клас*	*Кількість пікселів*
1	Alfalfa	46
2	Corn-notill	1428
3	Corn-mintill	830
4	Corn	237
5	Grass-pasture	483
6	Grass-trees	730

Кінець таблиці 2.1

Номер в датасеті	*Клас*	*Кількість пікселів*
7	Grass-pasture-mowed	28
8	Hay-windrowed	478
9	Oats	20
10	Soybean-notill	972
11	Soybean-mintill	2455
12	Soybean-clean	593
13	Wheat	205
14	Woods	1265
15	Buildings-Grass-Trees-Drives	386
16	Stone-Steel-Towers	93

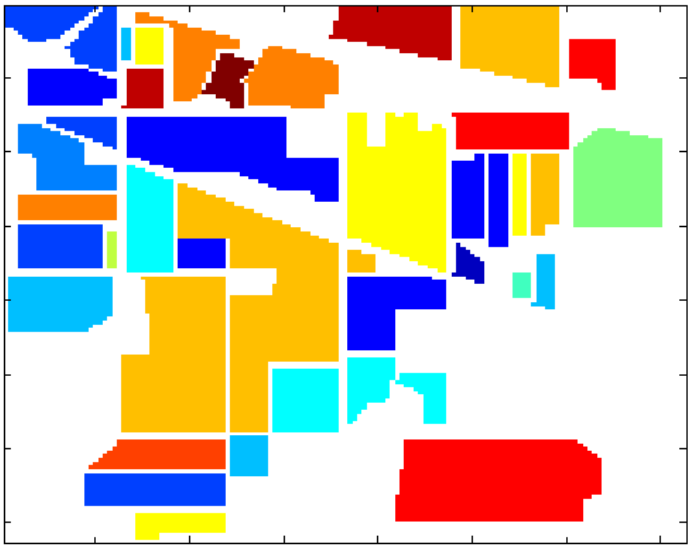

Рисунок 2.7 – Істинна розмітка сцени Indian Pines[4]

[4]Джерело зображення: https://www.ehu.eus/ccwintco/uploads/c/c6/Indian_pines_gt.png

Висновки до розділу 2

В розділі 2 проаналізовано наукові роботи, що використовують WGAN-GP у роботі з гіперспектральними даними. Помічено високу ефективність використання WGAN-GP моделі у задачах генерації даних для аграрного сектору. Розглянуті підходи охоплювали як по-піксельну генерацію, так і повноцінне створення синтетичних 3D-кубів. Деякі з підходів також застосовували допоміжні моделі у вигляді класифікаторів, а також механізми уваги для кращого видобуття просторово-спектральної інформації. Аналіз статей показав, що основними перевагами у використанні WGAN-GP моделей та її модифікацій є:

- висока стабільність навчання (жоден з методів не зазнав колапсу режиму),
- гнучкість архітектури, яка дозволяє адаптувати модель під різні формати вхідних даних,
- можливість контролю згенерованих класів з використанням умовної генерації та додаткових класифікаторів,
- успішне використання в умовах малого обсягу реальних даних.

Разом з цим, були виявлені певні недоліки:

- висока обчислювальна складність при роботі з декількома підмережами та використанні 3D-згорткових шарів,
- відсутність врахування просторової інформації у більшості з підходів,
- висока кількість гіперпараметрів та складність їх налаштування,
- більшість підходів використовували лише мітки класу як додаткову інформацію, а додаткові ознаки – якщо й залучалися – отримувалися переважно лінійними методами, такими як PCA.

Наприкінці розділу розглянуто набори даних, що будуть використані у даній роботі, а також виконано формальну постановку задачі з описом метрик для остаточної оцінки. У наступному розділі представлено процес розробки архітектури, описано загальну топологію мережі та наведено застосовану попередню обробку даних.

РОЗДІЛ 3 ПОБУДОВА АРХІТЕКТУРИ ТА ПОПЕРЕДНЯ ОБРОБКА ВХІДНИХ ДАНИХ

3.1 Попередня обробка даних

Розглянемо застосовану попередню обробку даних. Вона полягає у використанні двох алгоритмів: нормалізації та гауссового згладжування. Спочатку до початкового набору гіперспектральних зображень застосовується нормалізація. Нормалізація полягає у зведенні всіх значень у гіперспектральному 3D кубі до одного проміжку $[-1, 1]$. Вона виконується для кожного спектрального каналу окремо. Потрібно обов'язково нормалізувати дані до такого проміжку, який буде отримано на виході генератора. Так як генератор матиме функцію активації tanh, область значень якої дорівнює $[-1, 1]$, то й нормалізувати потрібно до того ж самого проміжку, щоб забезпечити коректну роботу мережі. Формула для нормалізації наведена нижче:

$$band_{normalized} = 2 * \left(\frac{band - \min(band)}{\max(band) - \min(band)}\right) - 1, \quad (3.1)$$

де $band$ – матриця значень одного спектрального каналу; $band_{normalized}$ – значення гіперспектрального каналу, зведені до проміжку $[-1, 1]$.

Коли вхідні дані не мають однакового масштабу, це може призводити до того, що одна ознака буде домінувати під час навчання, а інші ознаки будуть майже ігноруватись. Нормалізація допомагає запобігати таким ситуаціям, зводячи усі ознаки до спільного проміжку значень. Отже, головною причиною застосування нормалізації є підвищення стабільності процесу оптимізації, що сприяє швидшій збіжності під час навчання, а також пом'якшення проблем, пов'язаних зі зникненням або вибухом градієнтів [15].

Після нормалізації до вхідних гіперспектральних зображень застосовується гауссове згладжування. Сусідні пікселі гіперспектрального

зображення вважаються пов’язаними між собою, тому для видобуття додаткової просторової інформації про піксель, до зображення застосовується гауссове згладжування. Воно також допомагає зменшити просторову складність зображення, через що можливе застосування більш простих архітектур як для генератора, так і для дискримінатора. Для згладжування застосовується гауссовий фільтр, який зважує суму пікселів відповідно до відстані між сусідніми та центральним пікселем. Формула для обчислення згладженого пікселя в позиції mn, наведена нижче:

$$x_{mn}^{smooth} = \frac{\sum_i \sum_j x_{ij} \exp\left(-\|(i,j)-(m,n)\|^2/2\sigma^2\right)}{\sum_i \sum_j \exp\left(-\|(i,j)-(m,n)\|^2/2\sigma^2\right)}, \quad (3.2)$$

де x_{mn} – піксель в позиції mn; (i,j) – координати сусіднього пікселя; σ – параметр згладжування.

Як видно з формули (3.2), чим далі знаходиться сусідній піксель від центрального, тим менше він впливає на отримане значення, тому для меншої обчислювальної складності можна обмежитися лише певним вікном навколо пікселю.

3.2 Основа топології AC-WGAN-GP

Як зазначалося раніше, буде досліджуватися використання WGAN-GP для генерації синтетичних даних. За основу мережі буде обрано модифікацію WGAN-GP із статті [2] – AC-WGAN-GP. Дана модель розглядалася в попередньому розділі. Нагадаємо, що основними внесками статті є:

1) використання нової архітектури, яка поєднує в собі генератор, дискримінатор та додатковий класифікатор;
2) новий механізм онлайн-генерації;

3) алгоритм відбору прикладів, що базується на KNN.

Генератор мережі приймає на вхід випадковий вектор шуму z, який відповідає за варіації отриманого результату. В даній роботі цей латентний вектор отримується з 100-вимірного гауссового розподілу $\mathcal{N}$, що є стандартною практикою в генеративно-змагальних мережах. На додачу до випадкового шуму, генератору можуть надаватися додаткові входи, які допомагають з генерацією більш чітких та якісних зображень. Одним з таких входів є класова (умовна) інформація, яка використовується у мережах типу cGAN. В даній роботі унітарно закодований вектор класу для пікселя подається на вхід генератора. Окрім унітарно закодованого вектора класу c, а також вектора шуму z, на вхід генератора подається вектор головних компонент реального пікселя того ж класу, що й бажаний згенерований.

Головні компоненти отримуються за допомогою однойменного методу, який зазвичай використовується для зменшення розмірності багатовимірних даних. Для видобуття головних компонент гіперспектрального зображення, дані проектуються на 2D матрицю X, тобто замість $H \times W \times B$ гіперспектрального кубу ми отримуємо $(H * W) \times B$ матрицю. В даній матриці кожен рядок відповідає пікселю, а стовпчик – певному спектру. Дані центруються за кожним спектром відповідно формулі (3.3).

$$X_b^{centered} = X_b - \mu_b, \tag{3.3}$$

де X_b – b-й стовпик проектованої матриці; μ_b – середнє значення b-го стовпчика проектованої матриці.

Після цього обчислюється матриця коваріації за формулою (3.4), з якої видобуваються власні вектори та власні числа. Власні вектори даної матриці є головними компонентами. В роботі AC-WGAN-GP з них відбираються ті, яким відповідають 30 найбільших власних чисел. Відібрані власні вектори формують

матрицю V, яка використовується для проекції центрованої матриці за формулою (3.5).

$$Cov = \frac{1}{N-1} X_{centered}^{T} X_{centered}, \tag{3.4}$$

$$X_{PCA} = X_{centered} * V, \tag{3.5}$$

Отримана розмірність матриці X_{PCA} дорівнює $(H \times W) \times 30$. Трансформовані дані проектуються назад у 3D куб. З даного куба вибирається відповідний 30-розмірний вектор, який подається на вхід генератора. Всі вектори, що подаються на вхід генератора конкатенуються за довжиною, отримуючи вектор розмірності $100 + 30 + c$, де c – кількість класів у наборі даних.

Архітектура генератора наведена на рисунку 3.1. Першим шаром в ньому йде повнозв'язний, який приймає на вхід початковий вектор, а на виході видає вектор розмірності $\frac{1}{16} C * 512$. Після цього застосовується шар переформатування, який змінює розмірність даних на $\frac{1}{16} C \times 1 \times 512$, де 512 – кількість каналів, а $\frac{1}{16} C$ – довжина вектору. Далі в генераторі застосовуються 1D шари розгортки, які поступово збільшують розмірність вектору та зменшують його кількість каналів до того, поки вектор не набуде розмірності $C \times 1 \times 1$, що є одним гіперспектральним пікселем.

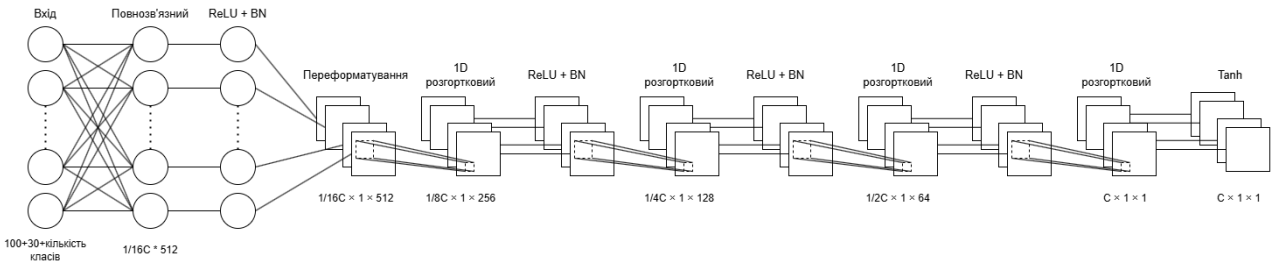

Рисунок 3.1 – Архітектура генератора

Між шарами застосовується функція активації ReLU. На виході генератора використовується tanh, який зводить значення усіх пікселів до проміжку [-1,1]. Функція ReLU визначається за формулою (3.6), а tanh – за формулою (3.7)

$$ReLU(x) = \max(0, x), \tag{3.6}$$

$$\tanh(\mathrm{x}) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}, \tag{3.7}$$

Мережа дискримінатора приймає на вхід один гіперспектральний піксель, який береться або з реального розподілу, або зі згенерованого. На виході мережі отримується числове значення, яке позначає наскільки вона впевнена у тому, що піксель взятий зі справжнього розподілу. Архітектура дискримінатора є оберненою до генератора, і представлена на рисунку 3.2.

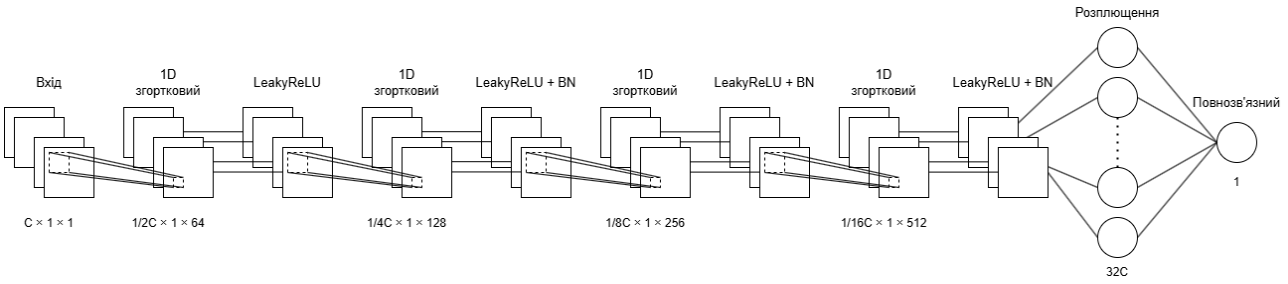

Рисунок 3.2 – Архітектура дискримінатора

Замість розгорткових шарів, в ній використовуються згорткові. Передостаннім йде шар розплющення (flatten), який перетворює отриману 2-вимірну структуру у вектор значень. Дискримінатор використовує функцію активації LeakyReLU між шарами для запобігання проблеми відмираючих нейронів, яка може виникати при використанні стандартного ReLU. Формула для LeakyReLU наведена у (3.8).

$$LeakyReLU(a, x) = \begin{cases} x, x \geq 0 \\ ax, x < 0 \end{cases}, \tag{3.8}$$

де a – параметр, що визначає нахил від'ємної частини функції.

Генератор та дискримінатор використовують шари пакетної нормалізації. Такий шар допомагає зменшити ефект перенавчання, а також запобігати проблемі затухаючих градієнтів. Пакетна нормалізація полягає у обрахуванні середнього значення та стандартного відхилення для кожної з ознак у пакеті, після чого відбувається нормалізація кожної ознаки для кожного окремого елементу даних за формулою:

$$\hat{A}_{ij} = \frac{A_{ij} - \mu_j}{\sigma_j}, \tag{3.9}$$

де A_{ij} – j-та ознака і-го прикладу в пакеті; μ_j – середнє значення для j-ї ознаки в пакеті; σ_j – стандартне відхилення для j-ї ознаки в пакеті.

Використання пакетної нормалізації обумовлено тією ж логікою, що й використання нормалізації для вхідного шару. Вона застосовується на активації попередніх шарів для того, щоб вони знаходилися у схожому проміжку значень, що зумовлює градієнти не розбігатися та не затухати [8].

Архітектура класифікатора складається з 3х шарів. Мережа має ідентичний вхід з дискримінатором. Перший шар мережі є згортковим з функцією активації tanh, другий – є шаром розплющення, а третій – повнозв'язним з функцією активації softmax. Формула для softmax наведена у (3.10). Виходом класифікатора є вектор, що описує ймовірності належності пікселя до кожного класу. Архітектура відображена на рисунку 3.3.

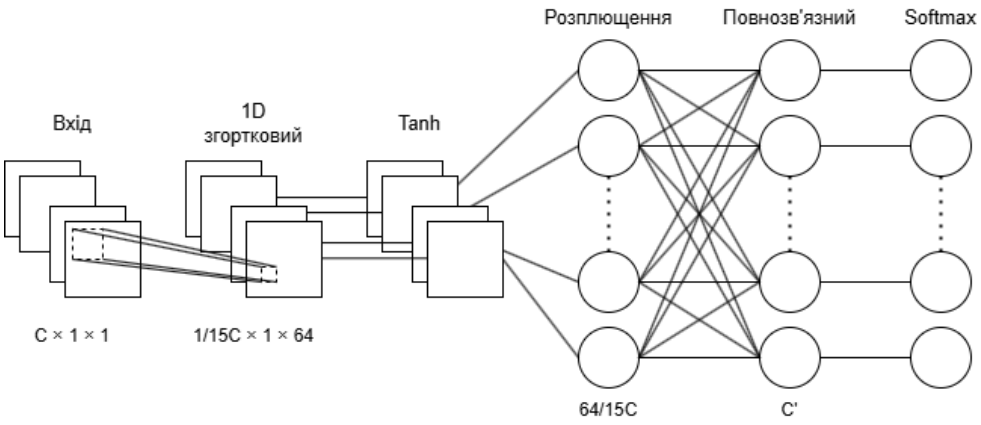

Рисунок 3.3 – Архітектура класифікатора

$$softmax(\boldsymbol{x})_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}. \quad (3.10)$$

3.3 Функції втрат

Для тренування мережі AC-WGAN-GP, як і для будь-якої іншої, використовуються функції втрат. Функція втрат – це диференційована функція, яка представляє «витрати», пов’язані з певним виходом мережі. Задача оптимізації нейронної мережі полягає у мінімізації відповідної функції втрат над тренувальним набором даних [1].

В роботі AC-WGAN-GP загальна функція втрат складається з 3х компонентів: функції втрат для генератора, дискримінатора та класифікатора. Кожна з них використовується для навчання відповідної підмережі. Функція втрат для дискримінатора залишилася незмінною з WGAN-GP:

$$L_D = E_{\tilde{x}\sim p(g)}[D(\tilde{x})] - E_{x\sim p(x)}[D(x)] + \lambda E_{\hat{x}\sim p(\hat{x})}[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2]. \quad (3.11)$$

Функція втрат для класифікатора (3.12) інкорпорує у собі лише частину, пов’язану з реальними прикладами. Це означає, що класифікатор навчається лише на істинних даних, а зворотній зв’язок із згенерованих йде на генератор.

$$L_C = -E_{x\sim p(x)}[\log P(C = c|X_{real})], \tag{3.12}$$

де C – вихід класифікатору, тобто передбачений клас; c – істинний клас пікселя.

Функція втрат генератора (3.13) складається з двох частин, одна з яких є зворотнім зв’язком дискримінатора, а інша – класифікатора. Зворотній зв’язок дискримінатора допомагає генератору створювати більш реалістичні приклади, а класифікатор допомагає у генерації таких прикладів, які більш явно належать до відповідного класу.

$$L_G = -E_{\tilde{x}\sim p(g)}[D(\tilde{x})] - E\big[\log p\big(C = c\big|X_{fake}\big)\big]. \tag{3.13}$$

3.4 Запропоновані модифікації мережі

Одним з ключових аспектів удосконалення AC-WGAN-GP моделі є новий підхід до формування вхідних ознак, отриманих з реальних даних, що подаються на вхід генератора. У базовій реалізації використовується метод головних компонент (PCA) для стискання спектральної інформації, але як уже було зазначено, такий підхід має низку обмежень, зокрема – лінійність. У зв’язку з цим, перспективним напрямком подальшого розвитку є пошук нелінійних методів, здатних ефективніше моделювати розподіл вхідних даних.

Потрібен такий метод, що буде ефективно зменшувати розмірність вхідного простору, надаючи найбільш репрезентативні ознаки для реальних гіперспектральних пікселів. Найбільш підходящим типом мереж для такої

задачі є автокодувальники. Автокодувальник – мережа, яка складається з кодувальника та декодувальника, які стискають елемент даних до простору меншої розмірності та реконструюють його назад до початкового. Існує два типи автокодувальників: звичайний або варіаційний. Варіаційний автокодувальник (VAE) формує латентний простір стохастичним чином, де для кожного вхідного елементу даних він надає певний розподіл над латентним простором. В той же час звичайний автокодувальник (AE) надає детерміновану репрезентацію з латентного простору. Зобразимо переваги та недоліки кожного підходу у таблиці 3.1.

Таблиця 3.1 – Порівняння VAE та AE підходів

Ознаки	*AE*	*VAE*
Тип латентного простору	Детермінований	Стохастичний
Латентне представлення	$z = Enc(x)$	$z \sim N(\mu(x), \sigma(x))$
Функція втрат	Реконструкція	Реконструкція + KL-дивергенція
Гладкість латентного простору	Не гарантована	Гарантується
Обчислювальна складність	Проста	Складніша

Ключовим є формування такого латентного простору, який є придатним для генерації нових зразків, подібних до реальних. Для такої задачі варіаційний автокодувальник є більш придатним в порівнянні зі звичайним. На відміну від останнього, VAE формує неперервний латентний простір, у якому близькі приклади проектуються в суміжні латентні представлення. Крім того, стохастична природа варіаційного автокодувальника сприяє кращому узагальненню, тому що дозволяє уникнути перенавчання на детермінованих представленнях, а також забезпечує більшу варіативність синтезованих зразків. Таким чином, замість випадкових векторів шуму, що використовуються у класичних генеративних мережах, генератор отримує структуровані латентні

вектори, що несуть інформацію, вилучену з реальних даних. Загальна топологія модифікованої мережі AC-WGAN-GP, яку назвемо AC-VAE-WGAN-GP, викладена на рисунку 3.4.

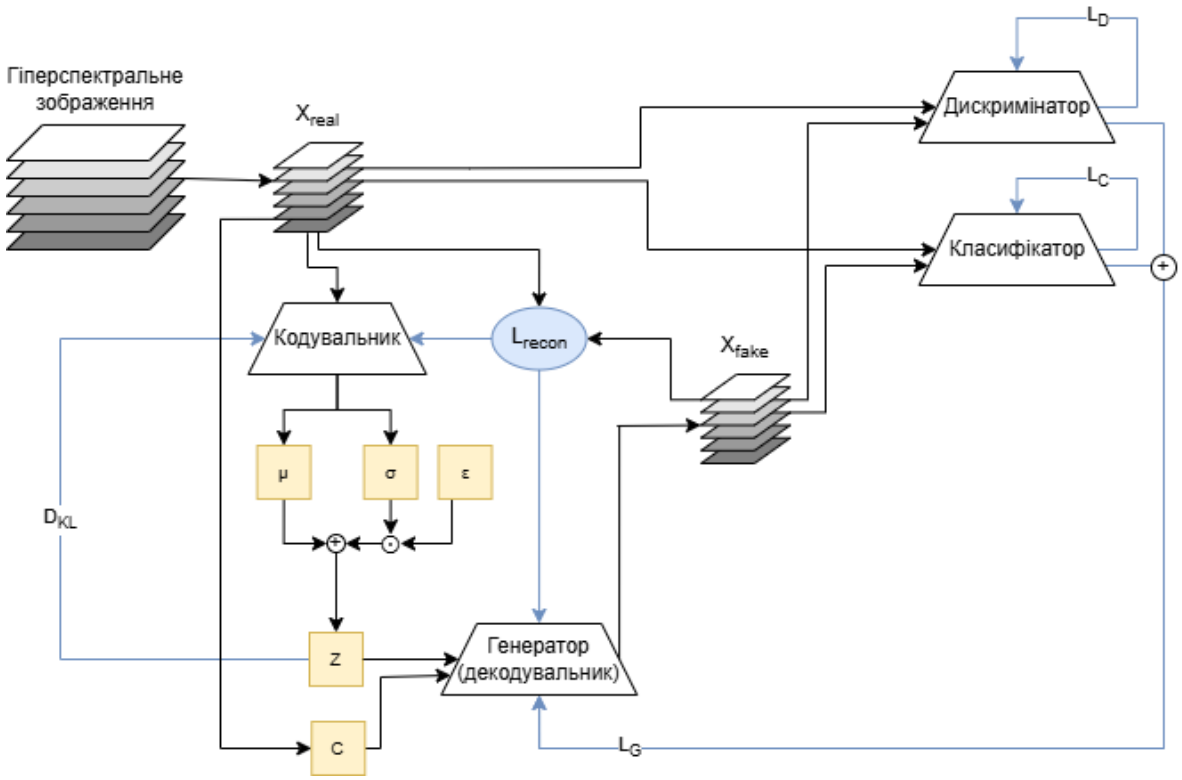

Рисунок 3.4 – Схематичне зображення запропонованої топології

Головним недоліком використання варіаційного автокодувальника у генеративних задачах вважається схильність до створення розмитих або згладжених зображень, що пов’язано з особливостями оптимізації функції втрат для реконструкції. Проте в контексті даної роботи, де генерація здійснюється на рівні окремих гіперспектральних пікселів, а не повноцінних зображень, цей недолік втрачає свою критичність. Крім того, для задач гіперспектральної класифікації визначальною є точність спектрального підпису, а не візуальні характеристики результату.

Для стискання гіперспектрального пікселя у латентний простір буде застосовано 1D згорткові шари, між якими використана функція активації

LeakyReLU та пакетна нормалізація. Архітектуру запропонованого кодувальника наведено на рисунку 3.5. Оновлену архітектуру генератора (декодувальника), зважаючи на вихід кодувальника, зображено на рисунку 3.6.

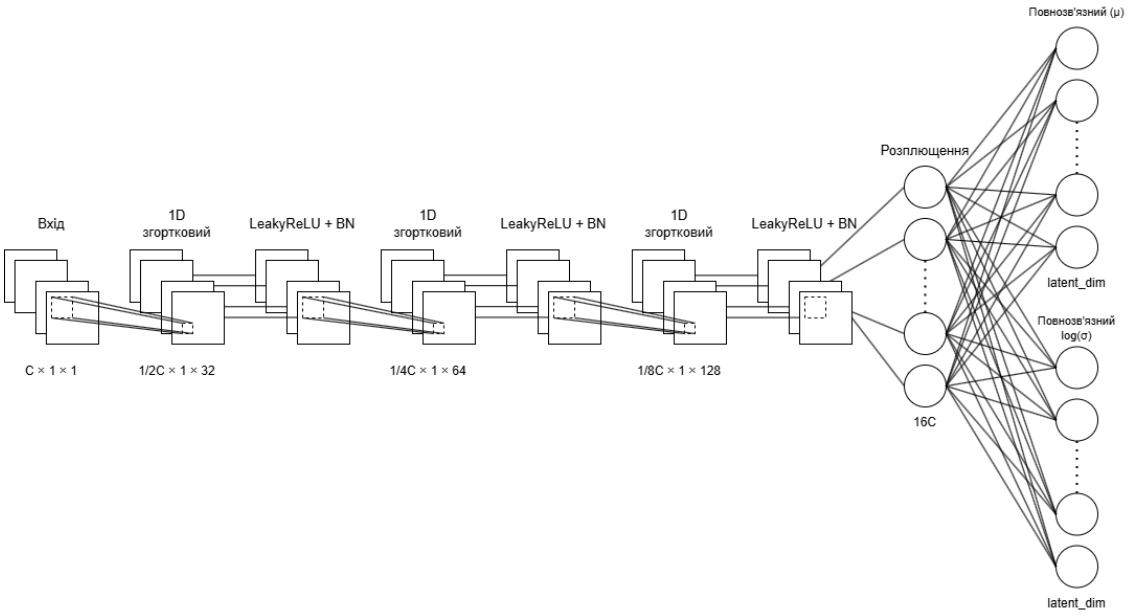

Рисунок 3.5 – Запропонована архітектура варіаційного кодувальника

На виході кодувальника отримуємо вектор середніх значень та логарифмів дисперсій для кожної позиції у векторі латентного простору. Для отримання латентного вектора z з параметрів μ та σ, що прогнозуються кодувальником, необхідно використати репараметризацію. Безпосередній семплінг з розподілу $z \sim \mathcal{N}(\mu, \sigma^2)$ порушує диференційованість, оскільки вибір випадкового зразка не має похідної. Репараметризація дозволяє обійти цю проблему шляхом представлення випадкової змінної z у вигляді (3.14). У такій формі всі обчислення залишаються диференційованими, що дозволяє коректно здійснювати градієнтний спуск.

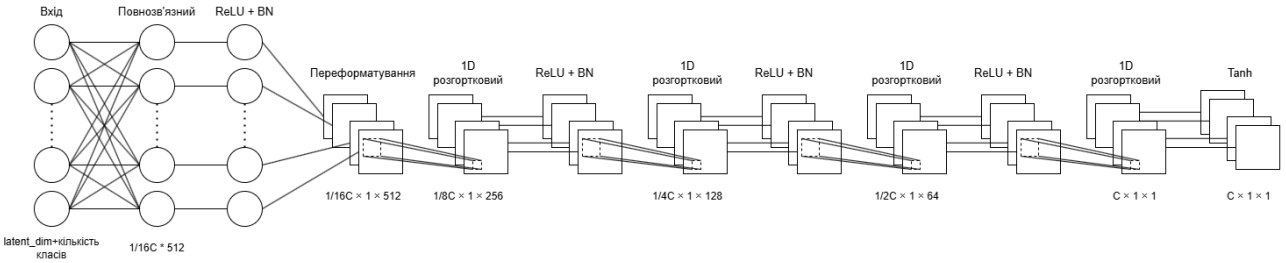

Рисунок 3.6 – Оновлена архітектура генератора (декодувальника)

$$z = \mu + \varepsilon \odot \sigma, \tag{3.14}$$

де $\varepsilon \sim \mathcal{N}(0,1)$; $\odot$ - поелементне множення.

Для підвищення стабільності у тренування генеративно-змагальної мережі пропонується модифікувати мережу дискримінатора, замінивши шари пакетної нормалізації на шари нормалізації за прикладом (Instance normalization). Вона виконує нормалізацію окремо для кожного прикладу. Завдяки цьому відпадає залежність від розміру пакета та від того, з якими іншими зображеннями поточний зразок опиняється в одному пакеті. В результаті нормалізація за прикладом забезпечує більш стабільне масштабування ознак для кожного екземпляра, зменшуючи «шум» від коливань статистик пакету між ітераціями. Саме тому її використання в дискримінаторі дає кращу збіжність і стійкість під час тренування GAN: вона мінімізує артефакти, що виникають через невеликі чи нерівномірні пакети. Оновлена архітектура дискримінатора представлена на рисунку 3.7.

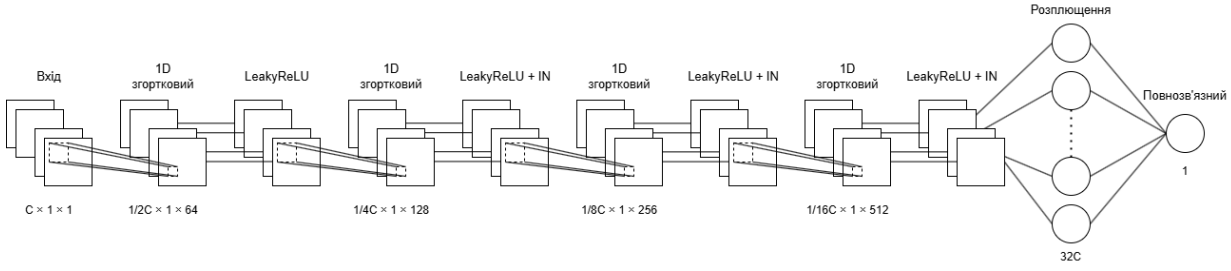

Рисунок 3.7 – Оновлена архітектура дискримінатора

Для якісного відновлення гіперспектральних пікселів важливо, щоб модель не лише повертала значення, близькі до оригінальних, але й зберігала загальну форму спектра – тобто характерну поведінку відбивання на різних довжинах хвиль. З цією метою в роботі було обрано комбіновану функцію втрат, що поєднує середньоквадратичну похибку (MSE) та спектральну кутову різницю (SAM). MSE оцінює відхилення відновлених значень від реальних у кожному спектральному каналі, забезпечуючи узгодженість в абсолютних значеннях (3.15). SAM вимірює кут між реальним та відновленим спектральним векторами, оцінюючи спектральну форму незалежно від масштабу значень (3.16). Загальна функція втрат реконструкції для VAE наведена у (3.17). Повна функція втрат для варіаційного автокодувальника представлена у формула (3.18).

$$MSE(x, \hat{x}) = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2, \tag{3.15}$$

$$SAM(x, \hat{x}) = \arccos\left(\frac{x \cdot \hat{x}}{\|x\| \cdot \|\hat{x}\| + \varepsilon}\right), \tag{3.16}$$

$$L_{recon}(x, \hat{x}) = (1 - \alpha) * MSE(x, \hat{x}) + \alpha * SAM(x, \hat{x}), \tag{3.17}$$

$$L_{VAE} = L_G + L_{recon} + \beta * D_{KL}(q(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})), \tag{3.18}$$

де x – реальний піксель, $\hat{x}$ – згенерований піксель, ε – невелика константа для числової стабільності, α – ваговий коефіцієнт відповідних частин реконструкції; β – ваговий коефіцієнт розбіжності Кульбака-Лейблера.

Висновки до розділу 3

В даному розділі було описано процес побудови модифікованої архітектури генеративно-змагальної мережі AC-WGAN-GP. Спочатку проаналізовано попередню обробку даних, яка включала нормалізацію та

гауссове згладжування. Нормалізація до проміжку [-1, 1] дозволяє узгодити вихід генератора з функцією активації tanh, а гауссове згладжування дозволяє інтегрувати просторову інформацію про сусідні пікселі перед початком навчання.

Модифікація AC-WGAN-GP полягала у заміні методу головних компонент на варіаційний автокодувальник для стискання спектральної інформації. Це дасть змогу подолати обмеження лінійності PCA та створити семантично збагачений латентний простір.

Архітектура генератора, дискримінатора та класифікатора була детально описана, з акцентом на використання 1D-згорткових шарів. В дискримінаторі запропоновано замінити вид нормалізації для підвищення стабільності навчання. Функції втрат для кожної підмережі були обґрунтовані, зокрема, комбінація MSE та SAM у декодувальнику для забезпечення якісної реконструкції спектральних ознак у синтетичних прикладах.

Це створює основу для подальшого експериментального дослідження ефективності моделі у розділі 4, де буде оцінено вплив згенерованих даних на результати класифікації та їх розподіл у порівнянні з реальними даними. Також у четвертому розділі буде проведено вибір оптимальних класифікаторів для оцінки якості синтетичних даних.

РОЗДІЛ 4 ТРЕНУВАННЯ МОДЕЛІ ТА АНАЛІЗ РЕЗУЛЬТАТІВ

4.1 Тренування моделей AC-WGAN-GP

Для тренування моделей з набору даних Indian Pines відібрано 215 прикладів (2%), щоб симулювати нестачу даних та класову незбалансованість. Всі інші приклади використовувались для формування тестового набору. Тренування моделі було поділено на 2 режими: PCA та VAE. Перший відповідає архітектурі, що описана в статті [2], другий – модифікованій архітектурі запропонованій в цій роботі.

Процес тренування полягав у налаштуванні однієї з підмереж, в той час як параметри інших були «заморожені». Цикл тренування наступний: дискримінатор → класифікатор → генератор (автокодувальник). Така послідовність обумовлена тим, що для якісної роботи генератора, класифікатор та дискримінатор повинні надавати корисний зворотний зв’язок.

Для тренування мережі потрібно спочатку підібрати оптимальні гіперпараметри. Гіперпараметри для налаштування моделі в режимі PCA взято зі статті [2]. Список гіперпараметрів та їх значень наведено нижче:

- кількість епох: 5000;
- розмірність вектора z: 100;
- темп навчання дискримінатора та класифікатора: 1e-4;
- темп навчання генератора: 1e-3;
- розмір пакету: 64;
- ваговий коефіцієнт градієнтного штрафу λ: 10;
- вектор коефіцієнтів β: (0.5, 0.9).

Тренування мережі в режимі VAE потребує певного пошуку в просторі гіперпараметрів. Такий пошук націлений на знаходження оптимальної їх комбінації, яка надає найкращу продуктивність моделі. Провести повноцінний пошук є обчислювально затратним процесом, тому вирішено протестувати

лише чотири комбінації гіперпараметрів. Значення більшості спільних між режимами гіперпараметрів було збережено. Найбільш оптимальна комбінація наведена нижче:

- розмірність вектора z: 60;
- ваговий коефіцієнт всередині втрат реконструкції α: 0.3;
- τ: 2500;
- темп навчання кодувальника: 1e-3.

Для тренування обох моделей застосовувався оптимізатор Adam з наведеними вище темпами навчання та параметрами β. Спостерігання за процесом навчання моделі відбувалося за допомогою сервісу weights and biases, куди передавалися деякі проміжні значення в процесі тренування. До таких значень належать величини функцій втрат та форми реальних і синтетичних спектрів. Значення функцій втрат для PCA моделі представлено на рисунку 4.1, для VAE моделі на рисунку 4.2. Справжні та згенеровані моделями спектри відображено на рисунках 4.3 та 4.4.

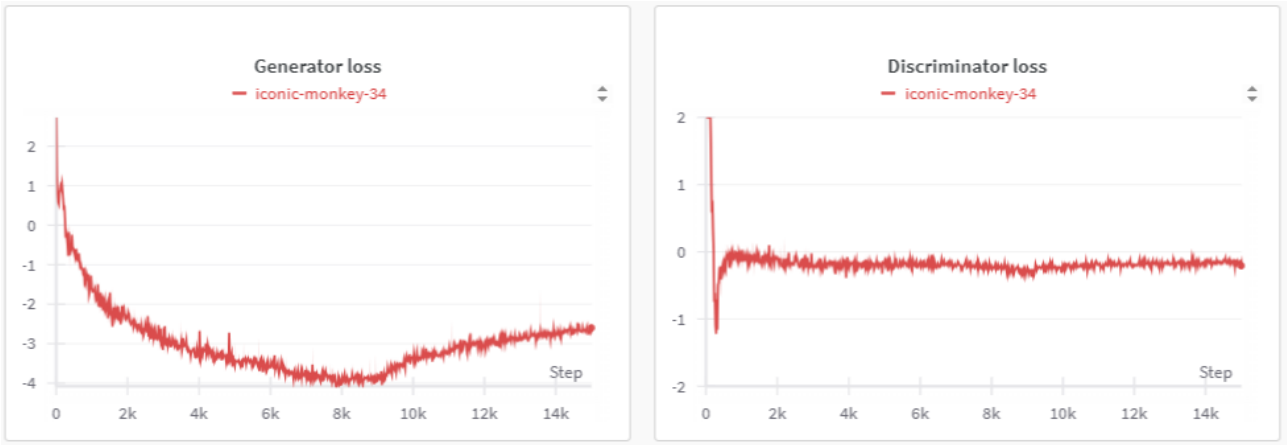

Рисунок 4.1 – Значення функцій втрат для PCA моделі

З огляду на динаміку втрат, тренування є стабільним. Втрати генератора стрімко зменшуються у перші кілька тисяч кроків, досягаючи плато близько -4, а далі повільно зростає й коливається у вузькому інтервалі. Одночасно втрати

дискримінатора після різкого стартового сплеску швидко «осідають» біля нуля і залишаються там із незначними коливаннями. Така поведінка характерна для здорового змагального процесу: жодна зі сторін не переважає.

Помітних ознак колапсу режиму (раптових стрибків втрат чи різкого падіння однієї з мереж) немає. Отже, оптимізація пройшла успішно, модель навчилася генерувати якісні зразки, здатні «обманути» генератор, а співвідношення сил між генератором і дискримінатором залишалося в робочому балансі впродовж усього навчання.

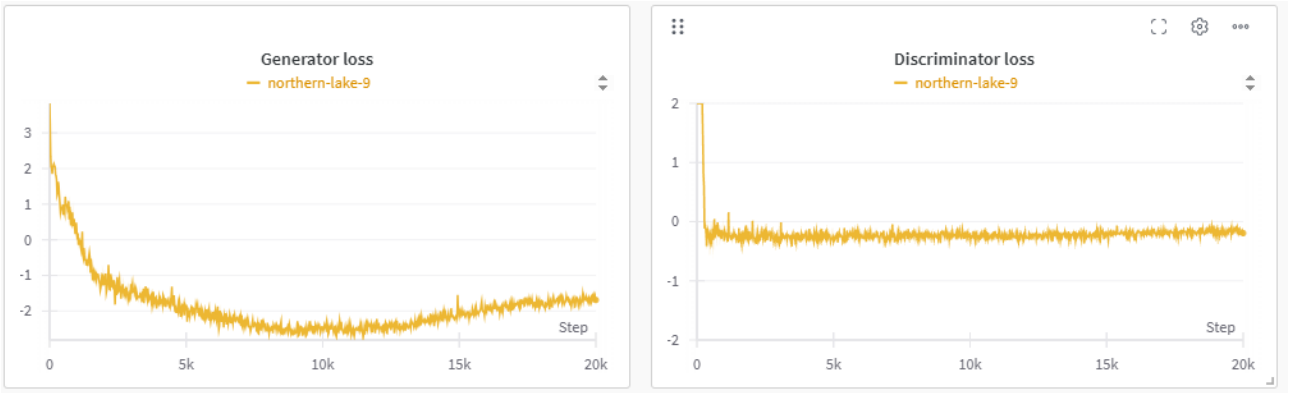

Рисунок 4.2 – Значення функцій втрат для VAE моделі

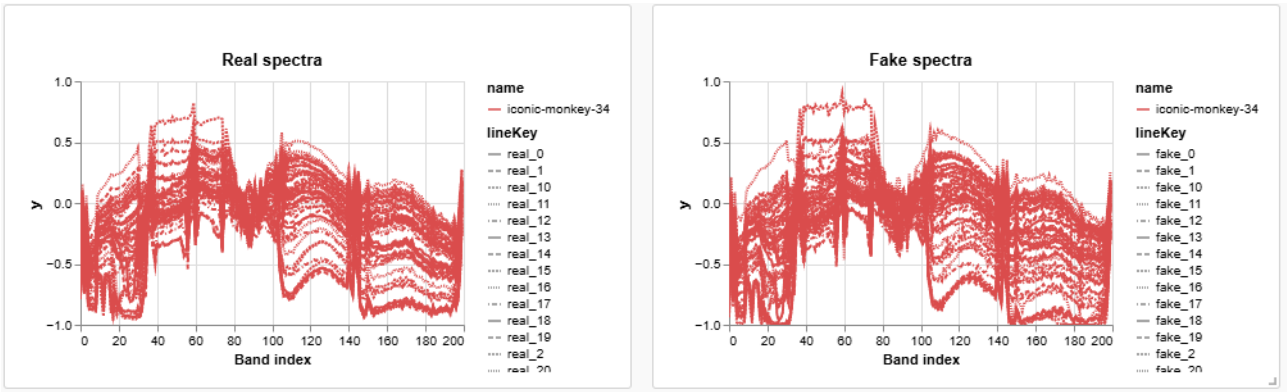

Рисунок 4.3 – Справжні та згенеровані PCA моделлю спектри

Дивлячись на отримані графіки спектрів, можна зробити висновок, що згенеровані спектри якісно відтворюють розподіл реальних пікселів: зберігається локальний максимум у діапазоні 40–70-х смуг, різкий спад сигналу

між 70-ю та 90-ю смугами. В той самий час у штучних спектрах спостерігається злегка менша дисперсія в ділянці понад 160-ту смугу; однак ця різниця є несуттєвою й, найімовірніше, не впливатиме на подальше використання згенерованих даних.

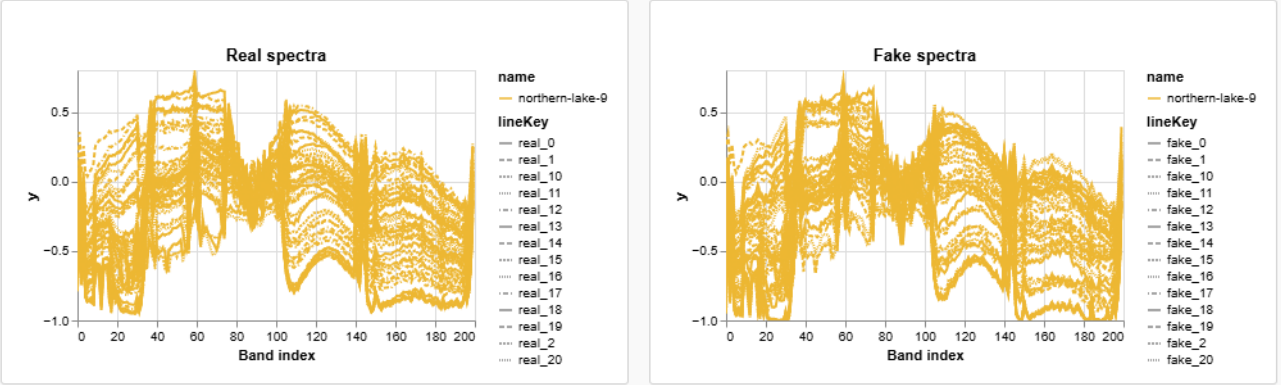

Рисунок 4.4 - Справжні та згенеровані VAE моделлю спектри

4.2 Порівняння розподілів згенерованих та реальних даних

Перш ніж використовувати синтетичні зразки для класифікації, потрібно переконатися, що їхній статистичний розподіл узгоджується з розподілом реальних даних. Оскільки кожен клас має власну структуру, штучно згенеровані пікселі з однією й тією ж класовою міткою повинні відтворювати той самий розподіл, що й відповідний клас у вхідній вибірці. Для порівняння розподілів використаємо 2 методи. По-перше, застосуємо метод головних компонент (PCA) до наборів реальних і синтетичних спектрів, де перші дві головні компоненти представлено на рисунку 4.5. Червоний колір позначає реальні спектри, зелений – згенеровані. По-друге, зобразимо спектральні сигнатури усереднених прикладів для кожного класу на рисунку 4.6.

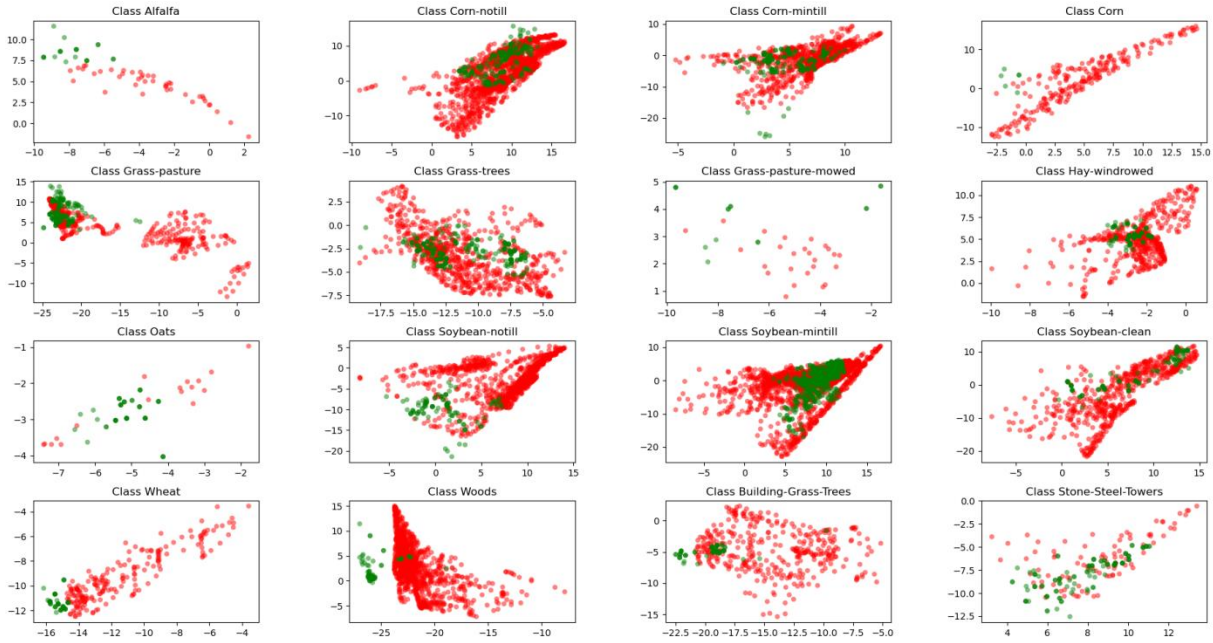

Рисунок 4.5 – Розподіл справжніх та згенерованих даних за класами

З рисунка 4.5 видно, що для більшості категорій синтетичні та реальні вибірки демонструють близькі розподіли. Водночас у класах із невеликою кількістю реальних зразків (наприклад, Alfalfa чи Grass-pasture-mowed) спостерігаються відчутні розбіжності. Додатково можна відзначити, що AC-WGAN-GP добре відтворює простіші розподіли (як-от Soybean-clean або Corn-notill), тоді як для складніших класів (Grass-pasture) репрезентативність згенерованих даних виявляється нижчою.

Рисунок 4.6 демонструє, що у переважній більшості випадків розподіли реальних і синтетичних спектрів добре збігаються. Однак є класи, де синтетичні спектри виявляють систематичне зміщення порівняно з реальними (зокрема, Soybean-notill та Soybean-clean).

У підсумку можна стверджувати, що згенеровані моделлю AC-WGAN-GP спектральні розподіли є достатньо репрезентативними та придатними для подальшої аугментації даних.

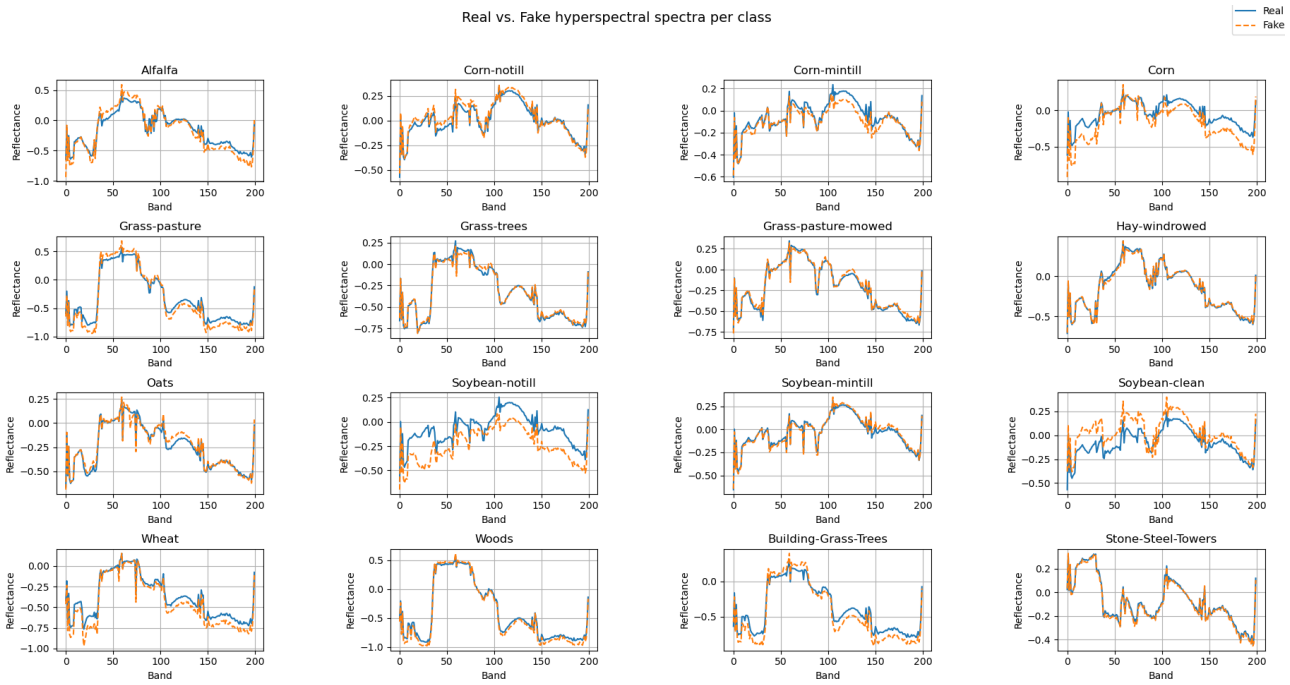

Рисунок 4.6 – Порівняння спектрів реальних та згенерованих даних за класами

4.3 Аналіз покращення класифікації за різних пропорцій згенерованих даних

Співвідношення справжніх та синтетичних даних у тренувальній вибірці може впливати на остаточну якість класифікації, тож в цьому підрозділі проведено дослідження впливу різних співвідношень на кінцеву продуктивність класифікаційної моделі. Обрано декілька поширених пропорцій для тестування: 1:1, 1:2, 1:3. А також проведено порівняння якості класифікації при відсутності аугментації з іншими класифікаторами. Порівняння класифікації без аугментації тренувального датасету наведено у таблиці 4.1.

Для порівняння класифікації використано 3 різних класифікатори. 1D-S-CNN – класифікатор з AC-WGAN-GP. 1D-CNN модель взята з статті [6], а 3D-CNN – зі статті [16]. 1D-S-CNN тренувалася на даних, до яких попередньо було застосовано гауссове згладжування, в той час як 1D-CNN та 3D-CNN навчалися

на незгладжених даних. За результатами таблиці можна зробити висновок, що 1D-S-CNN є найкращою серед наведених класифікаційних моделей навіть без застосування аугментації. Це демонструє ефективність застосування гауссового згладжування для вилучення додаткової просторової інформації для задачі класифікації гіперспектральних пікселів.

Таблиця 4.1 – Точність класифікації різних моделей без аугментації на Indian Pines

Класифікатор	*Метрика*		
	OA, %	AA, %	Kappa, %
1D-S-CNN	82.4	84.6	79.9
1D-CNN	69.1	67.7	64.8
3D-CNN	74.7	74.9	71.1

У таблиці 4.2 наведено точність класифікації 1D-S-CNN моделі за різних співвідношень реальних та згенерованих даних. Напівжирним виділено найкращі результати для кожного з режимів тренування моделі. При співвідношенні 1:1 обидва підходи демонструють майже ідентичні результати за всіма метриками. Зі збільшенням співвідношення синтетичних пікселів перевага переходить до VAE-режиму. На рівні 1:3 досягаються найвищі показники, що перевищують аналогічні значення для PCA приблизно на 1–1,5%, а відповідні найкращі значення – на ~0.5%.

Подальше збільшення співвідношення не забезпечує істотного приросту якості, але підтверджує стабільність VAE-режиму, оскільки його результати залишаються не нижчими за PCA на всіх рівнях аугментації. Отже, найкращий баланс між розширенням вибірки та збереженням високої точності досягається за використання VAE-режиму при співвідношенні реальних і згенерованих даних близько 1:3.

Таблиця 4.2 – Точність класифікації AC-WGAN-GP в PCA та VAE режимах

Ре-жим	*Співвідношення (справжні : згенеровані)*											
	1:1			1:2			1:3			1:4		
	OA, %	AA, %	Kap-pa, %	OA, %	AA, %	Kap-pa, %	OA, %	AA, %	Kap-pa, %	OA, %	AA, %	Kap-pa, %
PCA	**83.8**	**85.9**	**81.5**	82.3	85.3	80.5	83.3	85.6	81	83.6	85.5	81.3
VAE	83.7	85.6	81.4	83.6	84.8	81.3	**84.2**	**86.5**	**82**	84.2	86.2	81.9

4.4 Аналіз карт сегментації

У цьому підрозділі візуально продемонстровано результати по-піксельної класифікації сцени Indian Pines за допомогою карт сегментації. Для порівняння наведені карти сегментації мережами: 1D-CNN, 1D-S-CNN та AC-WGAN-GP. Карти подано на рисунку 4.7, де кожен колір відповідає певному класу на зображенні.

З малюнку видно, що гауссове згладжування, застосоване в мережах 1D-S-CNN та AC-WGAN-GP сильно покращує якість сегментації зображення всередині класів. При цьому все ще залишаються помилкові точки на границі. Це показує, що використання гауссового згладжування може ефективно згладжувати спектральні вибірки всередині класу, але при цьому може створювати артефакти на границі.

Використання аугментації також покращує якість сегментації гіперспектральної сцени. На карті сегментації AC-WGAN-GP білими прямокутниками помічено зони, в яких було помітне покращення в якості порівняно з 1D-S-CNN, а червоними – погіршення. Очевидно, що білі зони переважають червоні. Це вказує на те, що аугментація дійсно сприяє покращенню класифікації.

Рисунок 4.7 – Карти сегментації сцени Indian Pines для різних методів класифікації

Висновки до розділу 4

У цьому розділі проведено повний цикл експериментального аналізу розробленого фреймворку AC-WGAN-GP. По-перше, проаналізовано криві втрат генератора та дискримінатора під час навчання, які вказують на стабільний перебіг навчання без ознак колапсу режиму: обидві втрати швидко виходили на плато й залишалися в робочому діапазоні протягом усіх 5 000 епох. По-друге, порівняння спектрів показало, що згенеровані пікселі відтворюють ключові спектральні особливості реальних даних.

Подальший аналіз впливу аугментації засвідчив істотне зростання класифікаційної якості. Найбільший приріст досягнуто у VAE-режимі за співвідношенням реальних і згенерованих даних 1:3, де OA зросла до 84,2%, а коефіцієнт Kappa – до 82%, що перевищує найкращі результати PCA-режиму приблизно на 0.5%. Візуальний огляд карт сегментації підтвердив: поєднання гауссового згладжування з аугментацією зменшує шум усередині класів і розширює правильно класифіковані області, хоча певні артефакти залишаються на межах класів.

Отже, розроблена модифікація AC-WGAN-GP успішно збагачує тренувальний набір, покращує точність сегментації гіперспектральних сцен і демонструє найкращі результати саме при використанні варіаційного автокодувальника та співвідношенні 1:3 між реальними та синтетичними даними.

РОЗДІЛ 5 ФУНКЦІОНАЛЬНО–ВАРТІСНИЙ АНАЛІЗ ПРОГРАМНОГО ПРОДУКТУ СПРЯМОВАНОГО НА АУГМЕНТАЦІЮ ГІПЕРСПЕКТРАЛЬНИХ ДАНИХ З ВИКОРИСТАННЯМ GAN

У цьому розділі буде проведено оцінювання основних характеристик для майбутнього програмного продукту, що спеціалізується на дослідженні демографічного стану.

Дана реалізація буде сприяти проведенню усіх необхідних досліджень, що дасть змогу якісно дослідити питання не лише в Україні, проте у всьому світі.

Також в даному дослідженні показано різні варіанти реалізації для забезпечення найбільш коректної та оптимальної стратегії вибору, що має вплив на економічні фактори та сумісність з майбутнім програмним продуктом. Для цього застосовувався апарат функціонально-вартісного аналізу.

Функціонально-вартісний аналіз (ФВА) передбачає собою технологію, що дозволяє оцінити реальну вартість продукту або послуги незалежно від організаційної структури компанії. ФВА проводиться з метою виявлення резервів зниження витрат за рахунок ефективніших варіантів виробництва, кращого співвідношення між споживчою вартістю виробу та витратами на його виготовлення. Для проведення аналізу використовується економічна, технічна та конструкторська інформація.

Алгоритм функціонально-вартісного аналізу включає в себе визначення послідовності етапів розробки продукту, визначення повних витрат (річних) та кількості робочих часів, визначення джерел витрат та кінцевий розрахунок вартості програмного продукту.

5.1 Постановка задачі проектування

У роботі застосовується метод ФВА для проведення техніко-економічного аналізу розробки методу аугментації гіперспектральних даних за допомогою застосування модифікованої генеративно-змагальної мережі AC-WGAN-GP. Оскільки рішення стосовно проектування та реалізації компонентів, що розробляється, впливають на всю систему, кожна окрема підсистема має її задовольняти. Тому фактичний аналіз представляє собою аналіз функцій програмного продукту, призначеного для збору, обробки та проведення аналізу даних по компанії.

Технічні вимоги до програмного продукту такі:

- функціонування на персональних комп'ютерах із стандартним набором компонентів;
- зручність та зрозумілість для користувача;
- швидкість обробки даних та доступ до інформації в реальному часі;
- можливість зручного масштабування та обслуговування;
- мінімальні витрати на впровадження програмного продукту.

5.2 Обґрунтування функцій програмного продукту

Головна функція F_0 – розробка програмного продукту, призначеного для того, щоб проаналізувати доцільність використання генеративно-змагальних моделей у задачі аугментації гіперспектральних даних. Беручи цю функцію за основу, можна виділити наступні:

- F_1 – вибір мови програмування;
- F_2 – вибір фреймворку для машинного навчання;

- F_3 – вибір середовища розробки.

Кожна з наведених вище функцій має декілька варіантів реалізації.

Функція F_1:

а) Python;

б) Java.

Функція F_2:

а) Torch;

б) Tensorflow.

Функція F_3:

а) Visual Studio Code;

б) PyCharm.

Морфологічна карта системи з варіантами реалізації основних функції зображена на рисунку 5.1.

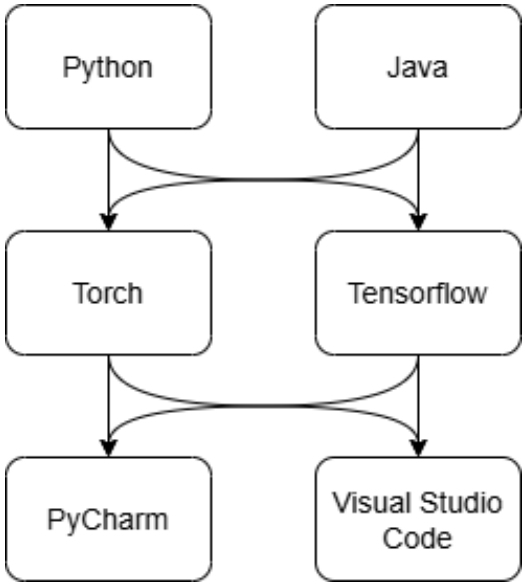

Рисунок 5.1 – Морфологічна карта системи

Морфологічна карта відображає множину всіх можливих варіантів основних функцій. Побудуємо позитивно-негативну матрицю варіантів основних функцій, яка наведена у таблиці 5.1.

Таблиця 5.1 – Позитивно-негативна матриця

Функції	*Варіанти реалізації*	*Переваги*	*Недоліки*
F_1	А	Швидкість розробки, велика екосистема бібліотек, простий синтаксис	Обмежена багатопоточність, відносно низька швидкість роботи
	Б	Висока швидкість виконання коду, строга типізація	Більше споживання пам’яті, більше часу на розробку програми
F_2	А	Динамічний граф обчислень, зручне налагодження, Python-подібний API	Менше готових моделей, нижча швидкодія
	Б	Широкий вибір моделей, вища продуктивність	Великий поріг входження, громіздкий API
F_3	А	Легкий і швидкий старт, велика кількість безкоштовних розширень.	Може сповільнюватися при великій кількості розширень
	Б	Повнофункціональне IDE для Python, потужні інструменти розробки та налагодження	Порівняно висока ресурсоємність та довге завантаження, деякі функції є платними.

На основі аналізу позитивно-негативної матриці робимо висновок, що при розробці програмного продукту деякі варіанти реалізації функцій варто

відкинути, тому, що вони не відповідають поставленим перед програмним продуктом задачам.

Функція F_1.

Перевагу надаємо швидкості розробки, спрощеному написанню коду та великій кількості бібліотек для машинного навчання. Отже, варіант Б має бути відкинутий.

Функція F_2.

Для розробки нейронних мереж можна застосовувати обидва фреймворки, тому не відкидаємо жоден з варіантів.

Функція F_3.

Надаємо перевагу варіанту Б, так як дане IDE спеціалізоване під роботу з мовою програмування Python.

Таким чином, будемо розглядати такий варіант реалізації ПП:

$$F_1\text{а} - F_2\text{а} - F_3\text{б},$$

$$F_1\text{а} - F_2\text{б} - F_3\text{б}.$$

Для оцінки якості розглянутих функцій обрана система параметрів, що описана нижче.

5.3 Обґрунтування системи параметрів програмного продукту

На основі даних, розглянутих вище, визначаються основні параметри вибору, які будуть використані для розрахунку коефіцієнта технічного рівня.

Для того, щоб охарактеризувати програмний продукт, будемо використовувати наступні параметри:

- X_1 – швидкодія мови програмування;
- X_2 – об’єм пам’яті для обчислень та збереження даних;
- X_3 – час навчання даних;

- X_4 – потенційний об’єм програмного коду.

Гірші, середні і кращі значення параметрів вибираються на основі вимог замовника й умов, що характеризують експлуатацію ПП як показано у таблиці 5.2.

Таблиця 5.2 – Основні параметри програмного продукту

Назва параметра	*Умовні позначення*	*Одиниці виміру*	*Значення параметра*		
			гірші	середні	кращі
Швидкодія мови програмування	X_1	оп/мс	9500	13000	16000
Об’єм пам’яті	X_2	Мб	1024	512	256
Час навчання моделі	X_3	с	10000	6000	3000
Потенційний об’єм програмного коду	X_4	кількість рядків коду	2000	1000	500

За даними таблиці 5.2 будуються графічні характеристики параметрів (рисунки 5.2 – 5.5).

Рисунок 5.2 – X_1, швидкодія мови програмування

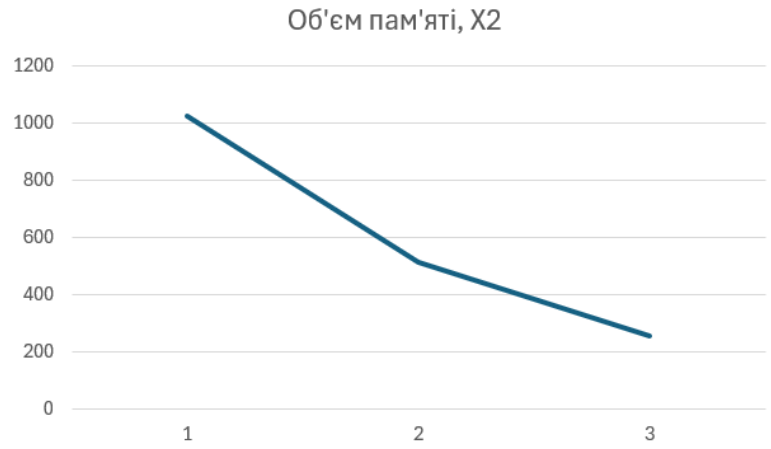

Рисунок 5.3 – X_2, об’єм пам’яті

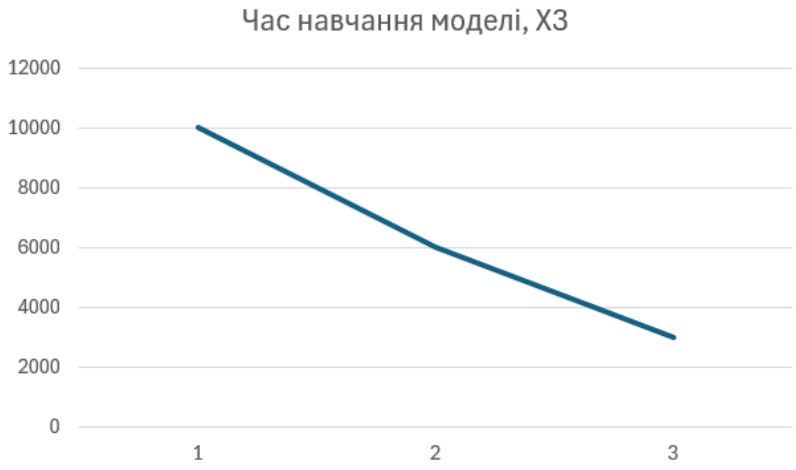

Рисунок 5.4 – X_3, час навчання моделі

Рисунок 5.5 – X_4, потенційний об’єм програмного коду

5.4 Аналіз експертного оцінювання результатів

Після детального обговорення й аналізу кожний експерт оцінює ступінь важливості кожного параметру для конкретно поставленої цілі – розробка програмного продукту, який дає найбільш точні результати при знаходженні параметрів моделей адаптивного прогнозування і обчислення прогнозних значень.

Значимість кожного параметра визначається методом попарного порівняння. Оцінку проводить експертна комісія із 7 людей. Визначення коефіцієнтів значимості передбачає:

- визначення рівня значимості параметра шляхом присвоєння різних рангів;
- перевірку придатності експертних оцінок для подальшого використання;
- визначення оцінки попарного пріоритету параметрів;
- обробку результатів та визначення коефіцієнту значимості.

Результати експертного ранжування наведені у таблиці 5.3.

Таблиця 5.3 – Результати ранжування параметрів

Позначення параметра	*Назва параметра*	*Одиниці виміру*	*Ранг параметра за оцінкою експерта*							*Сума рангів* R_i	*Відхилення* Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
X_1	Швидкодія мови програмування	оп/мс	2	2	2	1	2	1	2	12	-5.5	30.25

Кінець таблиці 5.3

Позначення параметра	*Назва параметра*	*Одиниці виміру*	*Ранг параметра за оцінкою експерта*							*Сума рангів* R_i	*Відхилення* Δ_i	Δ_i^2
X_2	Об’єм пам’яті	Мб	5	4	4	3	3	3	5	27	9.5	90.25
X_3	Час навчання	с	1	1	2	2	1	2	1	10	-7.5	56.25
X_4	Потенційний об’єм програмного коду	Кількість рядків коду	2	3	2	4	4	4	2	21	3.5	12.25
	Разом		10	10	10	10	10	10	10	70	0	189

Для перевірки степені достовірності експертних оцінок, визначимо наступні параметри:

- сума рангів кожного з параметрів і загальна сума рангів:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 70, \tag{5.1}$$

де N – число експертів; n – кількість параметрів;

- середня сума рангів:

$$T = \frac{1}{n} R_{ij} = 17.5. \tag{5.2}$$

- відхилення суми рангів кожного параметра від середньої суми рангів:

$$\Delta_i = R_i - T, \tag{5.3}$$

сума відхилень по всім параметрам повинна дорівнювати 0;

- загальна сума квадратів відхилення:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 189. \quad (5.4)$$

Порахуємо коефіцієнт узгодженість:

$$W = \frac{12S}{N^2(n^3-n)} = \frac{12*189}{7^2(4^3-4)} = 0.7714 \ldots > W_k = 0.67. \quad (5.5)$$

Ранжування можна вважати достовірним, тому що знайдений коефіцієнт узгодженості перевищує нормативний, котрий дорівнює 0.67.

Скориставшись результатами ранжирування, проведемо попарне порівняння всіх параметрів і результати занесемо у таблицю 5.4.

Таблиця 5.4 – Попарне порівняння параметрів

Параметри	Експерти							Кінцева оцінка	Числове значення
	1	2	3	4	5	6	7		
X_1 і X_2	<	<	<	<	<	<	<	<	0.5
X_1 і X_3	>	>	>	<	>	<	>	>	1.5
X_1 і X_4	>	<	>	<	<	<	>	<	0.5
X_2 і X_3	>	>	>	>	>	>	>	>	1.5
X_2 і X_4	>	>	>	<	<	<	>	>	1.5
X_3 і X_4	<	<	>	<	<	<	<	<	0.5

Числове значення, що визначає ступінь переваги i-го параметра над j-тим, a_{ij} визначає по формулі:

$$a_{ij} = \begin{cases} 1.5 \text{ при } X_i > X_j, \\ 1.0 \text{ при } X_i = X_j, \\ 0.5 \text{ при } X_i < X_j. \end{cases} \quad (5.6)$$

З отриманих числових оцінок переваги складемо матрицю $A = \|a_{ij}\|$.

Для кожного параметра зробимо розрахунок вагомості $K_{\text{ві}}$ за наступними формулами:

$$K_{\text{ві}} = \frac{b_i}{\sum_{i=1}^{n} b_i}, \quad (5.7)$$

$$b_i = \sum_{i=1}^{N} a_{ij}. \quad (5.8)$$

Відносні оцінки розраховуються декілька разів доти, поки наступні значення не будуть незначно відрізнятися від попередніх (менше 2%). На другому і наступних кроках відносні оцінки розраховуються за наступними формулами:

$$K_{\text{ві}} = \frac{b_i'}{\sum_{i=1}^{n} b_i'}, \quad (5.9)$$

$$b_i' = \sum_{i=1}^{N} a_{ij} b_j. \quad (5.10)$$

Як видно з таблиці 5.5, різниця значень коефіцієнтів вагомості не перевищує 2%, тому більшої кількості ітерацій не потрібно.

Таблиця 5.5 – Розрахунок вагомості параметрів

Параметри x_i	Параметри x_j				Перша ітер.		Друга ітер.		Третя ітер.	
	X_1	X_2	X_3	X_4	b_i	$K_{\text{ві}}$	b_i^1	$K_{\text{ві}}^1$	b_i^2	$K_{\text{ві}}^2$
X_1	1	0.5	1.5	0.5	3.5	0.22	12.25	0.21	44.875	0.21
X_2	1.5	1	1.5	1.5	5.5	0.34	21.25	0.35	77.875	0.36

Кінець таблиці 5.5

Параметри x_i	Параметри x_j				Перша ітер.		Друга ітер.		Третя ітер.	
X_3	0.5	0.5	1	0.5	2.5	0.16	9.25	0.16	34.125	0.16
X_4	1.5	0.5	1.5	1	4.5	0.28	16.25	0.28	59.125	0.27
Всього:					16	1	59	1	216	1

5.5 Аналіз рівня якості варіантів реалізації функцій

Визначаємо рівень якості кожного варіанту виконання основних функцій окремо.

Абсолютні значення параметрів X_1 (швидкодія мови програмування), X_2 (об'єм пам'яті), X_3 (час навчання), X_4 (потенційний об'єм програмного коду) відповідають технічним вимогам умов функціонування даного ПП.

Коефіцієнт технічного рівня для кожного варіанта реалізації ПП розраховано у таблиці 5.6 за такою формулою:

$$K_K(j) = \sum_{i=1}^{n} K_{\text{ві},j} B_{i,j}, \tag{5.11}$$

де n – кількість параметрів; $K_{\text{ві}}$ – коефіцієнт вагомості i-го параметра; B_i – оцінка i-го параметра в балах.

За даними з таблиці 5.6 за формулою:

$$K_K = K_{\text{ТУ}}[F_{1k}] + K_{\text{ТУ}}[F_{2k}] + \cdots + K_{\text{ТУ}}[F_{zk}], \tag{5.12}$$

визначаємо рівень якості кожного з варіантів:

$$K_{K1} = 0.42 + 1.89 + 2.88 = 5.19.$$
$$K_{K2} = 0.42 + 0.81 + 2.88 = 3.22.$$

Як видно з розрахунків, кращим є 1 варіант, для кого коефіцієнт технічного рівня має більше значення.

Таблиця 5.6 – Розрахунок показників рівня якості варіантів реалізації основних функцій ПП

Основні функції	*Варіант реалізації функції*	*Парамет-ри*	*Абсолют-не значення парамет-ра*	*Бальна оцінка парамет-ра*	*Коефіцієнт вагомості параметра*	*Коефіцієнт рівня якості*
F_1	А	X_1	9500	2	0.21	0.42
F_2	А	X_4	1000	7	0.27	1.89
	Б	X_4	1500	3	0.27	0.81
F_3	Б	X_2	9000	8	0.36	2.88

5.6 Економічний аналіз варіантів розробки ПП

Для визначення вартості розробки ПП спочатку проведемо розрахунок трудомісткості.

Усі варіанти охоплюють два окремих завдання:

1. Розробка проекту програмного продукту.
2. Розробка програмної оболонки.

Завдання 1 за ступенем новизни відноситься до групи А, завдання 2 – до групи Б. За складністю алгоритми, які використовуються в завданні 1 належать до групи 1; а в завданні 2 – до групи 3.

Для реалізації завдання 1 використовується довідкова інформація, а завдання 2 використовує інформацію у вигляді даних.

Проведемо розрахунок норм часу на розробку та програмування для кожного з завдань.

Загальна трудомісткість обчислюється як:

$$T_О = T_Р \cdot K_П \cdot K_{СК} \cdot K_М \cdot K_{СТ} \cdot K_{СТ.М}, \quad (5.13)$$

де $T_Р$ – трудомісткість розробки ПП; $K_П$ – поправочний коефіцієнт; $K_{СК}$ – коефіцієнт на складність вхідної інформації; $K_М$ – коефіцієнт рівня мови програмування; $K_{СТ}$ – коефіцієнт використання стандартних модулів і прикладних програм; $K_{СТ.М}$ – коефіцієнт стандартного математичного забезпечення.

Для першого завдання, виходячи із норм часу для завдань розрахункового характеру степеню новизни А та групи складності алгоритму 1, трудомісткість дорівнює: $T_Р$ = 60 людино-днів. Поправочний коефіцієнт, який враховує вид нормативно-довідкової інформації для першого завдання: $K_П$ = 1.8. Поправочний коефіцієнт, який враховує складність контролю вхідної та вихідної інформації для всіх семи завдань рівний 1: $K_{СК}$ = 1. Оскільки при розробці першого завдання використовуються стандартні модулі, враховуємо це за допомогою коефіцієнта $K_{СТ}$ = 0.9. Тоді загальна трудомісткість програмування першого завдання дорівнює:

$$T_1 = 60 \cdot 1.8 \cdot 0.9 = 97.2 \text{ людино-днів}$$

Проведемо аналогічні розрахунки для подальших завдань.

Для другого завдання (використовується алгоритм третьої групи складності, степінь новизни Б), тобто $T_Р$ = 31 людино-день, $K_П$ = 1.1, $K_{СК}$ = 1, $K_{СТ}$ = 0.8:

$$T_2 = 31 \cdot 1.1 \cdot 0.8 = 27.3 \text{ людино-днів}$$

Складаємо трудомісткість відповідних завдань для кожного з обраних варіантів реалізації програми, щоб отримати їх трудомісткість:

$$T_I = (97.2 + 27.3 + 4.8 + 27.3) \cdot 8 = 1252.8 \text{ людино-годин},$$
$$T_{II} = (97.2 + 27.3 + 6.91 + 27.3) \cdot 8 = 1269.7 \text{ людино-годин}.$$

Найбільш високу трудомісткість має варіант II.

В розробці беруть участь два програмісти з окладом 23000 грн., один аналітик в області даних з окладом 25000 грн. Визначимо середню зарплату за годину за формулою:

$$C_{\text{ч}} = \frac{M}{T_m \cdot t} \text{ грн.}, \quad (5.14)$$

$$C_{\text{ч}} = \frac{23000+23000+25000}{3 \cdot 21 \cdot 8} = 140.87 \text{ грн}, \quad (5.15)$$

де M – місячний оклад працівників; T_m – кількість робочих днів в місяць; t – кількість робочих годин в день.

Тоді, розрахуємо заробітну плату за формулою:

$$C_{\text{ЗП}} = C_{\text{ч}} \cdot T_i \cdot K_{\text{Д}}, \quad (5.16)$$

де $C_{\text{ч}}$ – величина погодинної оплати праці програміста; T_i – трудомісткість відповідного завдання; $K_{\text{Д}}$ – норматив, який враховує додаткову заробітну плату.

Зарплата розробників за варіантами становить:

I. $C_{\text{ЗП}} = 140.87 \cdot 1252.8 \cdot 1.2 = 211778.3$ грн,

II. $C_{\text{ЗП}} = 140.87 \cdot 1269.7 \cdot 1.2 = 214635.2$ грн.

Відрахування на єдиний соціальний внесок становить 22%:

$$\text{I. } C_{\text{ВІД}} = C_{\text{ЗП}} \cdot 0.22 = 211778.3 \cdot 0.22 = 46591.23 \text{ грн},$$
$$\text{II. } C_{\text{ВІД}} = C_{\text{ЗП}} \cdot 0.22 = 214635.2 \cdot 0.22 = 47219.74 \text{ грн}.$$

Тепер визначимо витрати на оплату однієї машино-години. ($\text{С}_\text{М}$)

Так як одна ЕОМ обслуговує одного програміста з окладом 23000 грн., з коефіцієнтом зайнятості 0.2, то для однієї машини отримаємо:

$$C_{\text{Г}} = 12 \cdot M \cdot K_{\text{З}} = 12 \cdot 23000 \cdot 0.2 = 55200 \text{ грн}.$$

З урахуванням додаткової заробітної плати:

$$C_{\text{ЗП}} = C_{\text{Г}} \cdot (1 + K_{\text{З}}) = 55200 \cdot (1 + 0.2) = 66240 \text{ грн}.$$

Відрахування на єдиний соціальний внесок:

$$C_{\text{ВІД}} = C_{\text{ЗП}} \cdot 0.22 = 66240 \cdot 0.22 = 14572.8 \text{ грн}.$$

Амортизаційні відрахування розраховуємо при амортизації 25% та вартості ЕОМ – 35000 грн.

$$C_{\text{А}} = K_{\text{ТМ}} \cdot K_{\text{А}} \cdot \text{Ц}_{\text{ПР}} = 1.3 \cdot 0.25 \cdot 35000 = 11375 \text{ грн.},$$

де $K_{\text{ТМ}}$ – коефіцієнт, який враховує витрати на транспортування та монтаж приладу у користувача; $K_{\text{А}}$ – річна норма амортизації; $\text{Ц}_{\text{ПР}}$ – договірна ціна приладу.

Витрати на ремонт та профілактику розраховуємо як:

$$C_P = K_{\text{ТМ}} \cdot \text{Ц}_{\text{ПР}} \cdot K_{\text{Р}} = 1.3 \cdot 35000 \cdot 0.05 = 2275 \text{ грн.},$$

де $K_{\text{Р}}$ – відсоток витрат на поточні ремонти.

Ефективний годинний фонд часу ПК за рік розраховуємо за формулою:

$$T_{\text{ЕФ}} = (\text{Д}_{\text{К}} - \text{Д}_{\text{В}} - \text{Д}_{\text{С}} - \text{Д}_{\text{Р}}) \cdot t_{\text{З}} \cdot K_{\text{В}} = (365 - 104 - 12 - 16) \cdot 8 \cdot 0.8 = \\ = 1491.2 \text{ години},$$

де $\text{Д}_{\text{К}}$ – календарна кількість днів у році; $\text{Д}_{\text{В}}, \text{Д}_{\text{С}}$ – відповідно кількість вихідних та святкових днів; $\text{Д}_{\text{Р}}$ – кількість днів планових ремонтів устаткування; t – кількість робочих годин в день; $K_{\text{В}}$ – коефіцієнт використання приладу у часі протягом зміни.

Витрати на оплату електроенергії розраховуємо за формулою:

$$C_{\text{ЕЛ}} = T_{\text{ЕФ}} \cdot N_C \cdot K_{\text{З}} \cdot \text{Ц}_{\text{ЕН}} = 1491.2 \cdot 0.3 \cdot 0.3 \cdot 9.43 = 1265.6 \text{ грн.},$$

де N_C – середньо-споживча потужність приладу; $K_{\text{З}}$ – коефіцієнт зайнятості приладу; $\text{Ц}_{\text{ЕН}}$ – тариф за 1 КВт-годин електроенергії.

Накладні витрати розраховуємо за формулою:

$$C_{\text{Н}} = \text{Ц}_{\text{ПР}} \cdot 0.67 = 35000 \cdot 0.67 = 23450 \text{ грн.}$$

Тоді, річні експлуатаційні витрати будуть:

$$C_{\text{ЕКС}} = C_{\text{ЗП}} + C_{\text{ВІД}} + C_{\text{А}} + C_{\text{Р}} + C_{\text{ЕЛ}} + C_{\text{Н}}, \qquad (5.17)$$

$$C_{\text{ЕКС}} = 66240 + 14572.8 + 11375 + 2275 + 1265.6 + 23450 = 119178.4 \text{ грн.}$$

Собівартість однієї машино-години ЕОМ дорівнюватиме:

$$C_{\text{М}-\text{Г}} = \frac{C_{\text{ЕКС}}}{T_{\text{ЕФ}}} = \frac{119178.4}{1491.2} = 79.92 \text{ грн/год.}$$

Оскільки в даному випадку всі роботи, які пов’язані з розробкою програмного продукту ведуться на ЕОМ, витрати на оплату машинного часу, в залежності від обраного варіанта реалізації, складає:

$$C_M = C_{\text{М}-\text{Г}} \cdot T, \tag{5.18}$$

$$\text{I. } C_M = 79.92 \cdot 1252.8 = 100123.78 \text{ грн.}$$

$$\text{II. } C_M = 79.92 \cdot 1269.7 = 101474.42 \text{ грн.}$$

Накладні витрати складають 67% від заробітної плати:

$$C_H = C_{\text{ЗП}} \cdot 0.67, \tag{5.19}$$

$$\text{I. } C_H = 211778.3 \cdot 0.67 = 141891.46 \text{ грн.}$$

$$\text{II. } C_H = 214635.2 \cdot 0.67 = 143805.58 \text{ грн.}$$

Отже, вартість розробки ПП за варіантам становить:

$$C_{\text{ПП}} = C_{\text{ЗП}} + C_{\text{ВІД}} + C_{\text{М}} + C_{\text{Н}}, \tag{5.20}$$

$$\text{I. } C_{\text{ПП}} = 211778.3 + 46591.23 + 100123.78 + 141891.46 = 500384.77 \text{ грн.}$$

$$\text{II. } C_{\text{ПП}} = 214635.2 + 47219.74 + 101474.42 + 143805.58 = 507134.94 \text{ грн.}$$

5.7 Вибір кращого варіанту ПП техніко-економічного рівня

Розрахуємо коефіцієнт техніко-економічного рівня за формулою:

$$K_{TEPj} = \frac{K_{Kj}}{K_{\Phi j}}, \tag{5.21}$$

$$K_{TEP1} = \frac{5.19}{500384.77} = 1.037 \cdot 10^{-5},$$

$$K_{TEP2} = \frac{3.22}{507134.94} = 6.35 \cdot 10^{-6}.$$

Як бачимо, найбільш ефективним є перший варіант реалізації програми з коефіцієнтом техніко-економічного рівня $K_{TEP1} = 1.037 \cdot 10^{-5}$.

Після виконання функціонально-вартісного аналізу програмного комплексу що розроблюється, можна зробити висновок, що з альтернатив, що залишились після першого відбору двох варіантів виконання програмного комплексу оптимальним є перший варіант реалізації програмного продукту. У нього виявився найкращий показник техніко-економічного рівня якості $K_{TEP} = 6.35 \cdot 10^{-6}$.

Цей варіант реалізації програмного продукту має такі параметри:

- мова програмування – Python;
- використаний фреймворк – PyTorch;
- середовище розробки – PyCharm.

Даний варіант виконання програмного комплексу дає користувачу зручний інтерфейс, швидку реалізацію програми та доступний функціонал для роботи.

Висновки до розділу 5

У п’ятому розділі виконано функціонально-вартісний аналіз розробленого програмного продукту. Спершу виокремлено ключові функції системи, сформовано перелік параметрів, що найбільш повно характеризують її якість, та організовано експертне оцінювання кожного варіанта реалізації. Розраховані коефіцієнти дозволили кількісно порівняти альтернативи й визначити їх техніко-економічний рівень. Далі проведено детальний економічний

розрахунок: визначено трудомісткість етапів, витрати на заробітну плату, машинний час і накладні витрати.

Комплексні показники засвідчили перевагу першого варіанта реалізації, який забезпечує максимальний коефіцієнт техніко-економічного рівня та найнижчу вартість розробки.

ВИСНОВКИ

У ході роботи досліджено проблему нестачі збалансованих гіперспектральних даних для точного розрізнення рослинності. Показано, що висока вартість зйомки та мала кількість прикладів окремих класів суттєво гальмують якість сегментації.

У другому розділі проаналізовано існуючі підходи до генерації гіперспектральних даних, використовуючи WGAN-GP, серед них виділено AC-WGAN-GP як основу для подальшої архітектури. Виконано постановку задачі та досліджено гіперспектральний набір Indian Pines.

Для подолання нестачі даних розроблено схему генерації синтетичних спектрів на основі AC-WGAN-GP, названу AC-VAE-WGAN-GP. У базову архітектуру інтегровано варіаційний автокодувальник замість лінійного PCA. До вхідних даних застосовано нормалізацію та гауссове згладжування.

Під час експериментів встановлено, що згенеровані розподіли спектрів у більшості класів майже збігаються з реальними, а відхилення спостерігається лише в поодиноких малочисельних категоріях. Додавання штучних зразків у співвідношенні 1:3 (реальні : синтетичні) збільшило середню точність класифікації до ~84 % і підвищило коефіцієнт Каппа до 82 %, покращивши сегментаційні карти всередині класів.

Реалізація виконана на Python / PyTorch, що дало змогу мінімізувати витрати та скористатися широким набором доступних інструментів. Отримані результати демонструють практичну цінність підходу для точного землеробства та інших сфер, де потрібний детальний аналіз гіперспектральних даних, а також окреслюють напрямки подальшого удосконалення моделей генерації для рідкісних або складних класів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Учасники проектів Вікімедіа. Функція втрат. *Вікіпедія*. URL: https://uk.wikipedia.org/wiki/Функція_втрат (дата звернення: 17.05.2025).
2. AC-WGAN-GP: generating labeled samples for improving hyperspectral image classification with small-samples / C. Sun та ін. *Remote sensing*. 2022. Т. 14, № 19. С. 4910. URL: https://doi.org/10.3390/rs14194910.
3. Arjovsky M., Chintala S., Bottou L. Wasserstein generative adversarial networks. *International conference on machine learning* : матеріали Міжнар. наук. конф., м. Сідней, 6 серп. 2017 р. 2017. С. 214–223. URL: https://dl.acm.org/doi/10.5555/3305381.3305404.
4. Autoencoding beyond pixels using a learned similarity metric / A. B. L. Larsen та ін. *International conference on machine learning* : матеріали Міжнар. наук. конф., м. Нью-Йорк, 19 черв. 2016 р. 2016. С. 1558–1566. URL: https://dl.acm.org/doi/10.5555/3045390.3045555.
5. Contributors to Wikimedia projects. Hyperspectral imaging. *Wikipedia, the free encyclopedia*. URL: https://en.wikipedia.org/wiki/Hyperspectral_imaging (дата звернення: 17.05.2025).
6. Deep convolutional neural networks for hyperspectral image classification / W. Hu та ін. *Journal of sensors*. 2015. Т. 2015. С. 1–12. URL: https://doi.org/10.1155/2015/258619.
7. Dilmegani C. 12+ data augmentation techniques for data-efficient ML. *AIMultiple*. URL: https://research.aimultiple.com/data-augmentation-techniques/ (дата звернення: 17.05.2025).
8. Doshi K. Batch Norm Explained Visually–How it works, and why neural networks need it. Towards Data Science. URL:

https://towardsdatascience.com/batch-norm-explained-visually-how-it-works-and-why-neural-networks-need-it-b18919692739.

9. Gamaya case study. *Google Cloud*. URL: https://cloud.google.com/customers/gamaya#:~:text=So,%20by%20capturing%20every%20last,needs%20in%20a%20sustainable%20manner (дата звернення: 10.06.2025).
10. Generative adversarial networks / I. Goodfellow та ін. Communications of the ACM. 2020. Т. 63, № 11. С. 139–144. URL: https://doi.org/10.1145/3422622.
11. Hyper-CycleGAN: a new adversarial neural network architecture for cross-domain hyperspectral data generation / Y. He та ін. *Applied sciences*. 2025. Т. 15, № 8. С. 4188. URL: https://doi.org/10.3390/app15084188.
12. Hyperspectral imaging for agriculture. *imec*. URL: https://www.imechyperspectral.com/en/applications/hyperspectral-imaging-agriculture (дата звернення: 17.05.2025).
13. Hyperspectral remote sensing scenes. *Inicio - UPV/EHU*. URL: https://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes (дата звернення: 17.05.2025).
14. Improved training of wasserstein GANs / I. Gulrajani та ін. *NIPS: neural information processing systems* : матеріали Міжнар. наук. конф., м. Лонг-Біч, Каліфорнія, 4–9 груд. 2017 р. Ред-Гук, Нью-Йорк, 2017. С. 5769–5779. URL: http://dl.acm.org/doi/10.5555/3295222.3295327.
15. Jaiswal S. What is normalization in machine learning? A comprehensive guide to data rescaling. *datacamp*. URL: https://www.datacamp.com/tutorial/normalization-in-machine-learning (дата звернення: 17.05.2025).
16. Li Y., Zhang H., Shen Q. Spectral–Spatial classification of hyperspectral imagery with 3D convolutional neural network. *Remote sensing*. 2017. Т. 9, № 1. С. 67. URL: https://doi.org/10.3390/rs9010067.
17. Liu W., You J., Lee J. HSIGAN: a conditional hyperspectral image synthesis method with auxiliary classifier. *IEEE journal of selected topics in applied*

earth observations and remote sensing. 2021. Т. 14. С. 3330–3344. URL: https://doi.org/10.1109/jstars.2021.3063911.

18. Malyshev O. Використовуємо CNN для обробки зображень. Частина перша. *DOU*. URL: https://dou.ua/forums/topic/48368/ (дата звернення: 11.06.2025).
19. Mirza M., Osindero S. Conditional generative adversarial nets. *ArXiv*. 6 лист. 2014 р. URL: https://doi.org/10.48550/arXiv.1411.1784.
20. Palsson B., Ulfarsson M. O., Sveinsson J. R. Synthesis of synthetic hyperspectral images with controllable spectral variability using a generative adversarial network. *Remote sensing*. 2023. Т. 15, № 16. URL: https://doi.org/10.3390/rs15163919.
21. Rana S., Gatti M. Comparative evaluation of modified wasserstein GAN-GP and state-of-the-art GAN models for synthesizing agricultural weed images in RGB and infrared domain. *MethodsX*. 2025. Т. 14. URL: https://doi.org/10.1016/j.mex.2025.103309.
22. Sun Q., Bourennane S. Unsupervised feature extraction based on improved Wasserstein generative adversarial network for hyperspectral classification. *Multimodal sensing and artificial intelligence: technologies and applications*, м. Munich, Germany, 24–27 черв. 2019 р. / ред.: S. Negahdaripour та ін. 2019. URL: https://doi.org/10.1117/12.2527466.
23. Synthetic hyperspectral reflectance data augmentation by generative adversarial network to enhance grape maturity determination / H. Lyu та ін. *Computers and electronics in agriculture*. 2025. Т. 235. URL: https://doi.org/10.1016/j.compag.2025.110341.
24. Utilizing wasserstein generative adversarial networks for enhanced hyperspectral imaging: a novel approach to predict soluble sugar content in cherry tomatoes / J. Cui та ін. *Lwt*. 2024. URL: https://doi.org/10.1016/j.lwt.2024.116585.

25. What is hyperspectral imaging?: a comprehensive guide. *Specim*. URL: https://www.specim.com/technology/what-is-hyperspectral-imaging/ (дата звернення: 17.05.2025).
26. Zammit A. Crop variety classification through hyperspectral imagery - pixxel. *Pixxel | Hyperspectral Imagery and Space Data Company*. URL: https://www.pixxel.space/blogs/crop-variety-classification-through-hyperspectral-imagery#:~:text=Each%20crop%20variety%20emits%20a,green,%20giving%20plants%20their%20colour (дата звернення: 10.06.2025).

ДОДАТОК А ЛІСТИНГ ПРОГРАМИ

Лістинг коду розміщено за посиланням https://github.com/RomanAharkov/AC-VAE-WGAN-GP-network.