МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРНІВЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ЮРІЯ ФЕДЬКОВИЧА

Навчально-науковий інститут фізико-технічних та комп'ютерних наук
кафедра комп'ютерних наук

Дослідження та розробка багатоетапної стратегії множинної редукції даних для оптимізації продуктивності моделей машинного навчання без втрати точності

Курсова робота
Рівень вищої освіти - другий (магістерський)

Виконав:
студент 5 курсу, 544 групи
Максимович Микола Юрійович
Керівник:
к.б.н., асистент **Талах М.В.**

Чернівці – 2024

Чернівецький національний університет імені Юрія Федьковича
Навчально-науковий інститут фізико-технічних та комп'ютерних наук
Кафедра Комп'ютерних наук
Спеціальність Комп'ютерні науки
Освітній ступінь Магістр
Форма навчання денна курс 5 група 544

ЗАВДАННЯ
НА КУРСОВУ РОБОТУ СТУДЕНТА
Максимовича Миколи Юрійовича
(прізвище , ім'я, по батькові)

1. Тема роботи

Дослідження та розробка багатоетапної стратегії множинної редукції даних для оптимізації продуктивності моделей машинного навчання без втрати точності

затверджена протоколом засідання кафедри від « 28 » серпня 2023 року №1

2. Термін подання студентом закінченої роботи 27.05.2024

3. Вхідні дані до роботи

Набори даних зображень різної роздільної здатності та розмірів для навчання, валідації та тестування моделей машинного навчання.

Можливі джерела даних включають:

- Відкриті набори даних зображень, такі як ImageNet, CIFAR, MNIST та інші.
- Спеціалізовані набори даних в галузі комп'ютерного зору та обробки зображень.
- Зібрані та розмічені набори даних зображень з реальних задач та додатків.

Технічні вимоги та обмеження:

- Апаратні обмеження обчислювальних ресурсів (ОЗУ, ГПУ, процесори).

4. Зміст розрахунково-пояснювальної записки (перелік питань, які треба розробити)

ВСТУП
РОЗДІЛ 1. БІЗНЕС-АНАЛІЗ ТА АНАЛІЗ ДАНИХ
РОЗДІЛ 2. DATA ENGINEERING (DATA PREPARATION)
РОЗДІЛ 3. MACHINE LEARNING, MODEL ENGINEERING
РОЗДІЛ 4. ОЦІНЮВАННЯ МОДЕЛЕЙ МАШИННОГО НАВЧАННЯ
ВИСНОВКИ
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

ДОДАТКИ

5. Перелік графічного, наочного матеріалу

Рисунки що зображують процес тренування моделей
Графіки з результатами підбору оптимальних параметрів
Графіки з результатами навчання на різних моделях
Матриці помилок для різних моделей
Візуалізація відновлених зображень після редукції різними методами

6. Консультант(и) курсової роботи:

№	Назва етапів курсової роботи	Термін виконання етапів роботи	Примітки
1	Вибір, погодження й затвердження теми, призначення наукового керівника, рецензента, консультанта	15.09.2023	виконано
2	Складання календарного плану й розширеного плану-конспекту роботи. Опрацювання джерел	16.09.2023	виконано
	Підготовка складових частин (розділів) роботи		
3	Отримання завдання на курсову роботу		виконано
4	Аналіз предметної області, дослідження літератури та матеріалів	16.09.2023	виконано
5	Аналіз існуючих аналогів програмного забезпечення	28.09.2023	виконано
6	Постановка задачі за темою курсової роботи	9.10.2023	виконано
7	Розділ 1. Бізнес-аналіз та аналіз даних	13.11.2023	виконано
8	Розділ 2. Інженерія даних (підготовка даних)	06.12.2023	виконано
9	Розділ 3. Розробка моделі машинного навчання	21.01.2024	виконано
10	Розділ 4. Оцінювання моделі машинного навчання	23.02.2024	виконано
16	Оформлення пояснювальної записки	22.03.2024	виконано
17	Перевірка відповідності оформлення	29.04.2024	виконано
18	Підготовка презентації та доповіді	16.05.2024	виконано
19	Захист курсової роботи	27.05.2024	виконано

Студент _____ Максимович М.Ю.
(підпис)

Науковий керівник _____ Талах М.В.
(підпис)

«___» ____ 2024 р.

АНОТАЦІЯ

Максимович Микола Юрійович. "Дослідження та розробка багатоетапної стратегії множинної редукції даних для оптимізації продуктивності моделей машинного навчання без втрати точності". Курсова робота освітнього рівня – магістр, на правах рукопису. Спеціальність – 122 Комп’ютерні науки (Інтелектуальний аналіз даних в комп’ютерних інформаційних системах). – Чернівці, 2024.

У ході виконання курсової роботи було створено конвеєр для редукції зображень з використанням алгоритмів сингулярного розкладу (SVD) та автоенкодерів. Реалізація конвеєру включала вибір та підготовку датасету, розробку алгоритмів, оцінку якості даних та модуль для оцінки результатів. Запропонований підхід дозволяє значно скоротити розмір даних без втрати якості, що сприяє зменшенню обчислювальних витрат та часу навчання моделей машинного навчання. Конвеєр обробки даних та архітектура системи були створені з урахуванням сучасних методів та практик, що забезпечило надійність та високу продуктивність. Проведений аналіз якості моделі підтвердив її відповідність вимогам і продемонстрував можливості для подальшого покращення.

Ключові слова: РЕДУКЦІЯ ЗОБРАЖЕНЬ, SVD, АВТОЕНКОДЕРИ, ПАЙПЛАЙН ОБРОБКИ ДАНИХ, МАШИННЕ НАВЧАННЯ.

Курсова робота містить результати власних досліджень. Використання ідей, результатів і текстів наукових досліджень інших авторів мають посилання на відповідне джерело.

_____ М.Ю. Максимович

(підпис)

ABSTRACT

Maksymovych Mykola Yuriiovych. "Research and development of a multistage multiple data reduction strategy to optimise the performance of machine learning models without loss of accuracy". Course work of the educational level - master's degree, in the form of a manuscript. Speciality – 122 Computer Science (Data mining in computer information systems) - Chernivtsi, 2024.

In the course of the course work, a pipeline for image reduction using singular value decomposition (SVD) algorithms and autoencoders was created. The implementation of the pipeline included dataset selection and preparation, algorithm development, data quality assessment, and a module for evaluating the results. The proposed approach allows to significantly reduce the size of data without losing quality, which helps to reduce computational costs and training time of machine learning models. The data processing pipeline and the system architecture were created with consideration of modern methods and practices, which ensured reliability and high performance. The conducted quality analysis of the model confirmed its compliance with the requirements and demonstrated opportunities for further improvement.

Keywords: IMAGE REDUCTION, SVD, AUTOENCODERS, DATA PROCESSING PIPELINE, MACHINE LEARNING.

The course work contains the results of my own research. The use of ideas, results and texts of scientific research of other authors must be referenced to the appropriate source.

_____ M.Y. Maksymovych
(signature)

ЗМІСТ

ВСТУП

В сучасному світі обсяги даних зростають з неймовірною швидкістю, що вимагає ефективних методів для їх зберігання, обробки та аналізу. Автоенкодери та методи сингулярного розкладу (SVD) відіграють ключову роль у редукції розмірності даних, дозволяючи зменшити вимоги до обчислювальних ресурсів та покращити швидкість обробки без значної втрати інформації. Це особливо актуально для галузей, де необхідно працювати з великими наборами даних, таких як комп'ютерний зір, машинне навчання та інші області штучного інтелекту. Розробка та оптимізація таких методів може значно прискорити процеси виявлення закономірностей у даних, що є важливим для наукових досліджень та комерційного застосування.

Метою курсової роботи є створення конвеєру для редукції зображень з використанням алгоритмів SVD (Singular Value Decomposition) та автоенкодерів. SVD є потужним інструментом для зниження розмірності даних, в той час як автоенкодери - це нейронні мережі, які навчаються кодувати вхідні дані в компактне представлення з подальшим відновленням оригінальних даних з цього представлення.

Завдання курсової роботи:

- Вибір датасету.
- Підготовка, обробка даних.
- Вибір та реалізація алгоритмів редукції даних у вигляді конвеєру.
- Розробка модуля оцінки якості редукції, візуалізації.
- Розробка модуля підбору оптимальних параметрів для конвеєру.

Об’єктом дослідження курсової роботи є процеси редукції розмірності даних в контексті великих наборів даних. Це включає аналіз та оптимізацію методів зменшення обсягу даних без істотної втрати інформативності.

Предметом дослідження курсової роботи є автоенкодери та метод сингулярного розкладу (SVD) як інструменти для ефективної редукції даних, розробка та покращення, навчання, підбір оптимальних параметрів, вибір моделей нейромереж для яких редукція буде найефективнішою.

Наукова новизна запропонованого підходу полягає у поєднанні лінійних та нелінійних методів редукції даних для максимального збереження візуальної інформації при значній редукції розмірності зображень. Така гібридна стратегія дозволяє використовувати переваги обох методів і досягати кращих результатів, ніж при їх окремому застосуванні.

Розробка ефективних методів редукції розмірності зображень матиме велике **практичне значення** для галузей, пов'язаних з обробкою великих обсягів візуальних даних, таких як комп'ютерний зір, машинне навчання та штучний інтелект. Редукція розмірності даних дозволить:

Зменшити вимоги до обчислювальних ресурсів, таких як пам'ять та потужність процесорів, що є критично важливим для роботи з високороздільними зображеннями або відеопотоками в режимі реального часу.

Прискорити процеси обробки та аналізу зображень, скоротивши час очікування та підвищивши продуктивність систем.

Зберегти візуальну інформативність даних, що є ключовим фактором для забезпечення якісного візуального сприйняття та коректної роботи алгоритмів комп'ютерного зору.

Покращити узагальнюваність моделей машинного навчання, зменшуючи проблеми надлишковості та мультиколінеарності ознак, що можуть призводити до перенавчання.

Забезпечити гнучкість та масштабованість підходу, який можна налаштувати відповідно до вимог конкретної задачі чи обмежень обчислювальних ресурсів.

Розроблені методи редукції розмірності зображень планується реалізувати у вигляді гнучкого та модульного конвеєру обробки даних, який можна інтегрувати в існуючі системи машинного навчання та комп'ютерного зору. Це дозволить оптимізувати використання обчислювальних ресурсів та прискорити процеси аналізу зображень у різноманітних практичних застосуваннях.

Структура роботи: курсова робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел (10 найменування), 1 додаток. Загальний обсяг роботи становить 82 сторінок основного тексту, 24 рисунки та 3 таблиці.

1. БІЗНЕС-АНАЛІЗ ТА АНАЛІЗ ДАНИХ

1.1. Візія та межі проекту

1.1.1. Бізнес-вимоги

В сучасному світі, де обсяг даних зростає експоненційно, ефективне зменшення розміру зображень без значної втрати якості є важливим завданням для багатьох галузей, таких як телекомунікації, мультимедіа та Інтернет речей. Редукція розміру зображень дозволяє ефективніше зберігати, передавати та обробляти візуальні дані, заощаджуючи місце на диску, пропускну здатність мережі та обчислювальні ресурси. Особливо важливою є редукція зображень для подальшого використання у навчанні моделей машинного навчання, оскільки вона дозволяє зменшити обсяг даних, що підлягають обробці. Це прискорює процес навчання моделей, забезпечуючи ефективніше використання обчислювальних ресурсів. Крім того, даний підхід сприяє більш швидкому та ефективному розвитку систем машинного навчання, підвищуючи їх продуктивність і здатність до обробки великих обсягів даних.

1.1.1.1. Бізнес-можливості

Комерційний продукт, що використовує багатоетапну стратегію редукції даних для оптимізації моделей машинного навчання, буде конкурувати на ринку, де існує значний попит на рішення, що зменшують обсяг даних для зберігання, передачі та обробки. Цей продукт буде особливо привабливим для компаній, що займаються розробкою та впровадженням рішень машинного навчання у таких галузях, як фінанси, охорона здоров'я, роздрібна торгівля, виробництво та IT.

Майбутня модель монетизації

- Ліцензування оптимізованих рішень машинного навчання з використанням багатоетапної стратегії редукції даних. Ліцензування таких рішень надасть компаніям доступ до перевірених та ефективних методик оптимізації, що зменшують обсяг даних для навчання моделей МН. Це забезпечить підвищення швидкості навчання моделей та зменшення

витрат на обчислювальні ресурси, що робить технологію привабливою для широкого спектру клієнтів.

- Надання хмарних сервісів з розгортанням та обслуговуванням оптимізованих моделей МН;
- Консультаційні послуги з впровадження стратегії редукції даних та оптимізації існуючих рішень МН. Такі послуги допоможуть компаніям адаптувати стратегії редукції даних під їхні специфічні потреби та існуючі технологічні рішення. Це дозволить оптимізувати продуктивність моделей МН, покращити точність прогнозів та знизити витрати на обробку даних. Таким чином, компанії отримають індивідуальні рішення, які відповідають їхнім бізнес-цілям та технічним вимогам.

Цільовий ринок

- Компанії, що займаються розробкою та впровадженням рішень машинного навчання у різних галузях (фінанси, охорона здоров'я, роздрібна торгівля, виробництво, IT та ін.). Такі компанії потребують ефективних методів для обробки великих обсягів даних та швидкого навчання моделей МН. Оптимізовані рішення з редукцією даних допоможуть їм підвищити ефективність бізнес-процесів, покращити точність прогнозів та зменшити витрати на обчислювальні ресурси.
- Науково-дослідні організації та академічні установи, що працюють у сфері машинного навчання часто стикаються з необхідністю обробки великих обсягів даних для проведення досліджень. Оптимізовані рішення з редукцією даних дозволять їм зменшити обсяг обчислювальних ресурсів, необхідних для аналізу даних, що сприятиме швидшому отриманню результатів досліджень та підвищенню їхньої точності.
- Стартапи, що розробляють інноваційні рішення на основі МН. Стартапи часто працюють з обмеженими ресурсами, але потребують ефективних і високопродуктивних рішень для швидкого виходу на ринок. Оптимізовані рішення з редукцією даних надають можливість швидко розгортати та масштабувати моделі МН, забезпечуючи конкурентні переваги та

ефективне використання обмежених ресурсів.

Таким чином, запропонований продукт не тільки вирішує актуальні бізнес-проблеми, але й надає конкурентні переваги, що робить його привабливим для широкого кола споживачів у різних галузях.

1.1.1.2. Бізнес-цілі та критерії успіху

- Розробити ефективну стратегію редукції даних, що забезпечить зменшення обчислювальних витрат під час тренування та використання моделей МН не менше ніж на 30% без суттєвої втрати точності (не більше 5%).
- Скоротити час навчання моделей МН щонайменше на 25%.
- Забезпечити економію обчислювальних ресурсів не менше 20% на етапі розгортання та використання моделей.

1.1.1.3. Визначення цільових груп користувачів

- Компанії, що розробляють та впроваджують рішення машинного навчання.
- Корпоративні клієнти (компанії, що розробляють/впроваджують рішення МН)
- Провайдери хмарних сервісів МН
- Академічні та дослідницькі установи
- Стартапи в сфері МН
- Окремі користувачі (потенційна група для створення споживчого сервісу)

Це допоможе максимально охопити ринок та знайти різні канали монетизації розробленого рішення з багатоетапною редукцією даних для оптимізації моделей машинного навчання.

1.1.1.4. Визначення потреб цільових груп

Компанії, що розробляють та впроваджують рішення машинного навчання:

1. Ефективне зберігання та обробка даних:
    - **Незадоволені потреби**: Потреба в зменшенні обсягу даних для зберігання та обробки без втрати якості.

- **Вирішення проблеми**: Наш продукт дозволяє ефективно зменшити розмір візуальних даних, забезпечуючи при цьому збереження їх високої якості.

2. Швидка обробка даних:
   - **Незадоволені потреби**: Потреба в зменшенні часу обробки великих обсягів даних для ефективного навчання моделей машинного навчання.
   - **Вирішення проблеми**: Наш продукт прискорює процес навчання моделей шляхом зменшення обсягу даних, що підлягають обробці.
3. Економія обчислювальних ресурсів:
   - **Незадоволені потреби**: Потреба в зниженні витрат на обчислювальні ресурси для розробки та впровадження рішень машинного навчання.
   - **Вирішення проблеми**: Наш продукт дозволяє знизити витрати на обчислювальні ресурси завдяки оптимізації обробки візуальних даних.

Корпоративні клієнти (компанії, що розробляють/впроваджують рішення МН):

1. Швидка інтеграція в існуючі системи:
   - **Незадоволені потреби**: Потреба в продукті, що легко інтегрується з існуючими рішеннями машинного навчання та аналітичними системами.
   - **Вирішення проблеми**: Наш продукт має зручний API для інтеграції з різними платформами та фреймворками машинного навчання.
2. Підтримка великих обсягів даних:
   - **Незадоволені потреби**: Потреба в продукті, що забезпечує ефективну обробку та аналіз великих обсягів даних.
   - **Вирішення проблеми**: Наш продукт дозволяє оптимізувати обробку візуальних даних для роботи з великими обсягами інформації.
3. Гнучкі та масштабовані рішення:

- **Незадоволені потреби**: Потреба в рішеннях, які можна легко масштабувати та адаптувати до різних потреб бізнесу.
- **Вирішення проблеми**: Наш продукт надає гнучкість у виборі рівня оптимізації даних та масштабування відповідно до вимог користувача.

Провайдери хмарних сервісів МН:

1. Інтеграція та підтримка машинного навчання в хмарному середовищі:
   - **Незадоволені потреби**: Потреба в продукті, що забезпечує ефективну оптимізацію даних та розвиток моделей машинного навчання в хмарному середовищі.
   - **Вирішення проблеми**: Наш продукт пропонує рішення для оптимізації обробки візуальних даних у хмарному середовищі, що дозволяє збільшити продуктивність та знизити витрати.
2. Гнучкі та масштабовані рішення для користувачів хмарних сервісів:
   - **Незадоволені потреби**: Потреба в рішеннях, які можна легко інтегрувати та масштабувати для користувачів хмарних сервісів машинного навчання.
   - **Вирішення проблеми**: Наш продукт пропонує рішення, які легко інтегруються з провайдерами хмарних сервісів машинного навчання та можуть масштабуватися згідно з потребами користувачів, що сприяє покращенню ефективності та забезпечує оптимальне використання ресурсів у хмарному середовищі.

Академічні та дослідницькі установи:

1. Доступ до інструментів для досліджень та розробки:
   - **Незадоволені потреби**: Потреба в продуктах, що дозволяють легко виконувати експерименти та дослідження в галузі машинного навчання.
   - **Вирішення проблеми**: Наш продукт надає потрібні інструменти для оптимізації та аналізу візуальних даних, що сприяє швидкому прогресу в академічних дослідженнях.

2. Підтримка розвитку нових методів та алгоритмів:
    - **Незадоволені потреби**: Потреба в продуктах, що дозволяють швидко впроваджувати нові методи та алгоритми машинного навчання.
    - **Вирішення проблеми**: Наш продукт допомагає дослідникам та академікам у розробці та впровадженні нових ідей та інновацій у галузі машинного навчання.

Стартапи в сфері машинного навчання:

1. Ефективність та економічність рішень:
    - **Незадоволені потреби**: Потреба в ефективних та економічних рішеннях для розвитку своїх продуктів на основі машинного навчання.
    - **Вирішення проблеми**: Наш продукт дозволяє стартапам зменшити витрати на обробку даних та розробку моделей машинного навчання, що сприяє їхньому швидкому розвитку та зростанню.

Окремі користувачі (потенційна група для створення споживчого сервісу):

1. Простота використання та доступність:
    - **Незадоволені потреби**: Потреба в продуктах, які легко використовувати та доступні для окремих користувачів без глибоких знань машинного навчання.
    - **Вирішення проблеми**: Наш продукт може бути використаний для створення споживчих сервісів, що дозволить користувачам без спеціалізованих навичок використовувати його для оптимізації та обробки візуальних даних.

1.1.1.5. Бізнес-ризики

Технічні ризики:

- **Опис**: Складність розробки ефективної багатоетапної стратегії редукції даних може призвести до викликів у забезпеченні суттєвого підвищення продуктивності без критичних втрат точності.

- **Можливі наслідки**: Затримки у розробці, втрата довіри користувачів через недоліки у функціонуванні продукту.

Ризик низького попиту:

- **Опис**: Низька поінформованість цільової аудиторії про переваги редукції даних або наявність альтернативних рішень може спричинити недостатній попит на розроблений продукт.
- **Можливі наслідки**: Затримки у випуску продукту, низький рівень прибутковості, необхідність додаткових маркетингових зусиль.

Ризик швидкого розвитку конкурентних технологій:

- **Опис**: Швидкий розвиток конкуруючих технологій може призвести до втрати конкурентоспроможності розробленого продукту.
- **Можливі наслідки**: Втрата ринкової позиції, зниження обсягів продажів, необхідність швидкої адаптації до нових тенденцій.

Ризик несумісності з існуючими системами клієнтів:

- **Опис**: Несумісність розробленого рішення з існуючими робочими процесами та системами клієнтів може ускладнити його впровадження та використання.
- **Можливі наслідки**: Затримки у впровадженні, незадоволеність клієнтів, втрата довіри.

Ризики безпеки та конфіденційності даних:

- **Опис**: Недоліки у застосуванні стратегії редукції даних можуть створити загрози для безпеки та конфіденційності інформації користувачів.
- **Можливі наслідки**: Виток конфіденційної інформації, штрафи за порушення законодавства про захист персональних даних.

Регуляторні ризики:

- **Опис**: Вимоги прозорості та пояснюваності моделей машинного навчання можуть стати перешкодою для впровадження продукту в певних галузях.
- **Можливі наслідки**: Обмеження доступу до ринку, необхідність додаткових зусиль для відповідності регулятивним вимогам.

1.1.1.6. Аналіз етичних міркувань та ризиків при взаємодії з системою МН

- Забезпечення конфіденційності та захисту персональних даних, що можуть використовуватись для тренування моделей.
- Уникнення можливої упередженості та дискримінації на основі даних, що використовуються.
- Розгляд потенційного негативного впливу неточних або некоректних результатів роботи моделей МН.

1.1.2. Перетворення бізнес-цілей на цілі машинного навчання. Визначення шаблону та підходів для застосування моделі машинного навчання

Основною метою проекту є створення ковеєру для редукції зображень з використанням алгоритмів SVD (Singular Value Decomposition) та автоенкодерів. SVD є потужним інструментом для зниження розмірності даних, в той час як автоенкодери - це нейронні мережі, які навчаються кодувати вхідні дані в компактне представлення з подальшим відновленням оригінальних даних з цього представлення.

Для розробки конвеєру редукції зображень доцільно обрати підхід розробки на основі моделі (Model-First Development). Це зумовлено наступними міркуваннями:

1. В даному випадку ключовим компонентом є саме модель машинного навчання, навколо якої буде розроблятися система. Тому має сенс зосередитися спочатку на розробці та оптимізації самої моделі.
2. Стратегія редукції даних безпосередньо стосується моделі машинного навчання та впливає на її продуктивність і точність. Спочатку необхідно дослідити та визначити ефективну стратегію редукції даних для моделі.
3. На цьому етапі може бути складно повністю спланувати системні вимоги та архітектуру, оскільки невідомо, якої точності та продуктивності можна досягти з оптимізованою моделлю після застосування стратегії редукції

даних.

4. Зосередження на моделі на ранньому етапі дозволить уникнути значних інвестицій у проєкт, який може виявитися нездійсненним, якщо цілі щодо продуктивності та точності не будуть досягнуті.

Після того, як буде розроблена та протестована ефективна стратегія редукції даних для моделі, команда зможе перейти до системного проєктування та врахувати ширші вимоги щодо користувацького досвіду, безпеки, справедливості тощо під час розробки системи навколо оптимізованої моделі.

Для реалізації розробки пайплайну багатоетапної стратегії множинної редукції даних планується використання наступних шаблонів:

Вкладення (Representation Learning). Цей шаблон дозволить отримати компактні векторні представлення високорозмірних вхідних даних, зберігаючи при цьому їхні ключові характеристики та семантичну інформацію. Зокрема, буде використано згорткову автоенкодерну нейронну мережу для навчання на вхідних зображеннях. Енкодер цієї мережі виконуватиме стиснення зображень у компактні вектори вкладень (embeddings) нижчої розмірності, зберігаючи найважливіші візуальні ознаки. Вектори вкладень можна отримати з передостаннього (або іншого прихованого) шару енкодера, який міститиме стисле представлення кожного зображення у вигляді низьковимірного векторного коду. Такий підхід дозволить значно зменшити обчислювальну складність та вимоги до пам'яті для наступних етапів обробки даних. Під час редукції даних часто виникає проблема великої кількості унікальних категоріальних значень, що збільшує розмірність даних. Використання вкладень допоможе зберегти важливу семантичну інформацію в стислому векторному форматі, спростивши подальшу обробку категоріальних ознак [1].

Налаштування гіперпараметрів (Hyperparameter Tuning). Оскільки стратегія редукції даних може включати декілька етапів та методів, зокрема автоенкодерну мережу для отримання вкладень, важливо ретельно підібрати оптимальні гіперпараметри для кожного з них. Це допоможе досягти балансу між ефективністю, точністю та відтворюваністю результатів на різних наборах

даних. Налаштування гіперпараметрів стане критичним для забезпечення високої якості отриманих вкладень та подальшої обробки даних.

Такий набір шаблонів, що поєднує вкладення через автоенкодери для редукції розмірності, налаштування гіперпараметрів та оцінку якості вкладень, допоможе впоратися з типовими проблемами, що виникають при роботі з великими наборами даних високої розмірності, і забезпечити ефективну реалізацію багатоетапної стратегії редукції даних.

1.2. Відбір даних. Збір та перевірка даних.

Для демонстрації та тестування підходів до редукції зображень в рамках даного проекту буде використано відомий набір даних MNIST, який містить 70000 зображень рукописних цифр розміром 28x28 пікселів у градаціях сірого. Цей набір є класичним та широко використовуваним для задач комп'ютерного зору і машинного навчання, що забезпечує можливість порівняння з іншими дослідженнями.

Статистичні властивості даних MNIST:

- Роздільна здатність зображень: 28x28 пікселів
- Колірна модель: Градації сірого (від 0 до 255)
- Формат файлів: Вбудований бінарний формат набору MNIST
- Обсяг навчальних даних: 60000 зображень
- Обсяг тестових даних: 10000 зображень
- Загальна кількість класів (цифр): 10 (від 0 до 9)

Набір MNIST є добре сформованим, повним та репрезентативним для завдання розпізнавання рукописних цифр. Він широко використовується в дослідженнях машинного навчання як базовий набір для тестування та порівняння різних алгоритмів та підходів.

Вимоги до даних для редукції зображень:

- Дані повинні бути у форматі масиву або тензора, сумісному з бібліотеками глибокого навчання (TensorFlow, PyTorch тощо)
- Бажано мати окремі набори даних для навчання, валідації та тестування

моделі

- Дані мають бути належним чином нормалізовані та стандартизовані
- Важливо мати метадані та документацію щодо походження даних та способу їх збору/генерації
- Дані повинні відповідати вимогам конфіденційності та безпеки для запобігання витоку чутливої інформації

Оцінка якості даних буде проводитись за наступними метриками:

- Повнота - відсутність відсутніх значень або пошкоджених зображень
- Точність - відповідність зображень змісту, що очікується (рукописні цифри)
- Послідовність - однакові роздільна здатність, розміри та формат зображень
- Репрезентативність - наявність у наборі усіх класів (цифр від 0 до 9)
- Незміщеність - відсутність значного дисбалансу класів

Якщо під час розробки моделі виникне потреба у використанні додаткових наборів даних, зокрема для трансферного навчання чи фіндтюнінгу. В такому разі буде проведено аналогічний аналіз та оцінку якості цих наборів даних.

1.3. Аналіз здійсненості проекту на основі Machine Learning Canvas framework

Аналіз здійсненості проекту проводився із застосуванням Machine Learning Canvas framework. Було визначено, що проект є цілком здійсненним, оскільні алгоритми SVD та автоенкодери добре зарекомендували себе для завдань редукції даних, а наявні обчислювальні ресурси та набори даних достатні для успішної реалізації проекту.

ЗАДАЧА: Оптимізувати продуктивність моделей машинного навчання шляхом зменшення розміру вхідних даних (зображень) без значної втрати точності.

РІШЕННЯ: Розробити багатоетапну стратегію редукції зображень з використанням SVD та автоенкодерів. Інтегрувати стратегію в пайплайн обробки даних для моделей МН [2].

ПРОПОЗИЦІЯ ЦІННОСТІ:

Кінцеві користувачі: компанії, що розробляють та впроваджують рішення на основі МН із застосуванням зображень та візуальних даних. Прикладами можуть бути системи комп'ютерного зору, розпізнавання образів, обробки медичних зображень тощо.

Мета: Підвищити обчислювальну ефективність, скоротити час навчання та витрати на розгортання

Переваги:

- Економія обчислювальних ресурсів (очікується ≥70% зменшення вимог до ОЗУ та обчислювальної потужності)
- Прискорення процесу навчання моделей (очікується ≥35% зменшення часу навчання)
- Значне зменшення обсягу даних (очікується ≥85% компресія вхідних зображень)
- Збереження семантичної інформації та ключових візуальних ознак завдяки використанню вкладень

ЗБІР ДАНИХ

Стратегія:

- Початковий набір даних для навчання і тестування стратегій редукції (MNIST)

- Контроль якості даних (повнота, точність, репрезентативність)

Обмеження та витрати:

- Необхідність інтеграції з різними системами збору даних

- Обробка великих наборів візуальних даних

ДЖЕРЕЛА ДАНИХ

- Наявний набір MNIST
- Інші потенційні джерела: відкриті набори даних, корпоративні дані клієнтів

МОДЕЛЮВАННЯ ВПЛИВУ

Чи можуть моделі бути розгорнуті? Так, оптимізовані моделі з

багатоетапною стратегією редукції даних можуть бути розгорнуті у тестовому та виробничому середовищі для оцінки їх продуктивності.

Які тести використовуються для оцінки продуктивності (помилкових) прогнозів?

- Для оцінки продуктивності використовуватимуться тестові набори даних, такі як частина даних MNIST та синтетично згенеровані тестові дані.
- Основними метриками оцінки будуть точність моделі на тестових даних, а також показники втрати продуктивності після застосування редукції даних.
- Проводитимуться порівняння продуктивності оптимізованої моделі з базовою моделлю без редукції даних.

СТВОРЕННЯ ПРОГНОЗІВ

Прогнозування з використанням оптимізованої моделі може відбуватися у різних режимах залежно від сценарію використання:

- Пакетна обробка зображень - модель виконуватиме прогнози для цілого набору зображень одночасно. Доступний час залежатиме від обсягу даних та вимог до затримки.
- Потокова обробка - прогнози в режимі реального часу для окремих зображень з жорсткими вимогами до затримки.
- Після прогнозування зображення можуть проходити подальшу обробку, наприклад, передаватися на інші етапи пайплайну машинного навчання.
- Час, доступний для фіналізації та обробки, буде визначатися відповідними вимогами до продуктивності та затримки в конкретному сценарії використання.

РОЗРОБКА МОДЕЛЕЙ

Скільки виробничих моделей потрібно?

Для реалізації багатоетапної стратегії редукції даних знадобиться дві основні моделі - SVD та автоенкодер для стиснення та відновлення зображень. Також буде використовуватись модель(MLPClassifier) для контролю змін точності, швидкості створення прогнозів та інших показників внаслідок

застосування редукції.

Модель буде оновлюватись (перенавчання або фіндтюнінг) у таких випадках:

- При надходженні нових наборів даних, які можуть покращити продуктивність моделі
- При зміні вимог до рівня редукції даних або допустимої втрати точності
- Періодично (наприклад, щоквартально) для підтримання актуальності моделі

Процес оновлення включатиме:

- Збір нових/оновлених наборів даних
- Тренування/фіндтюнінг моделі на оновлених даних
- Фіналізація та тестування нової версії моделі
- Аналіз продуктивності нової моделі порівняно з поточною версією
- Розгортання нової версії моделі у виробниче середовище при покращенні показників

ОЗНАКИ

Вхідні дані, доступні на момент прогнозування, вилучені з джерел даних. На вхід оптимізованої моделі подаватимуться зображення у форматі масиву. Вони будуть вилучатись з джерел вхідних даних, таких як:

- Набори зображень, завантажені з файлової системи/сховища даних
- Потокові дані, отримані через API чи з іншої системи обробки даних
- Інші проміжні накопичувачі/буфери, де зберігаються оброблені зображення

Вихідні дані після обробки пайплайном - це не відновлені зменшені зображення у вигляді масивів, готових для подальшої обробки або використання.

Вимоги до затримки: залежать від сценарію використання

Метрики:

- Відсоток економії обчислювальних ресурсів (цільове ≥15%)
- Прискорення часу навчання (цільове ≥20%)
- Зменшення розміру даних (цільове ≥30%)

- Втрата точності моделі після редукції даних (цільова ≤2%)

Відсоток економії обчислювальних ресурсів. Розраховується як відношення заощаджених ресурсів до початкових витрат. Прискорення часу навчання розраховується як відношення скороченого часу до початкового часу навчання. Зменшення розміру даних розраховується як відношення зменшеного розміру даних до початкового. Втрата точності моделі розраховується як різниця між точністю моделі до і після редукції даних.

1.4. Create POC. Бачення рішення (vision of the solution)

1.4.1. Формулювання бачення

Метою проекту є створення конвеєру для ефективної редукції розміру зображень з мінімальними втратами якості за допомогою поєднання лінійних та нелінійних методів редукції даних, зокрема, SVD та автоенкодера. Це рішення дозволить значно зменшити вимоги до місця для зберігання та передачі візуальних даних, підвищуючи ефективність роботи з великими обсягами зображень у різних галузях , таких як мультимедіа, телекомунікації, комп'ютерний зір та інші.

1.4.2. Компоненти системи з ML та без ML (ML and Non-ML Components in a System)

Система складатиметься з наступних компонентів машинного навчання (ML) та інших компонентів:

Компоненти ML:

- Модуль передобробки зображень
- Модель Автоенкодер для кодування/декодування зображень
- Модель SVD для редукції розмірності зображень
- Клас (пайплайн) який об’єднує та керує моделями.
- Модуль оцінки якості редукції на основі Multilayer Perceptron (MLP)
- Модуль підбору оптимальних параметрів для автоенкодера та SVD.

Компоненти Non-ML:

Система розподіленого зберігання зображень:

- Розподілена файлова система (наприклад, HDFS, Ceph) або сховище об'єктів (наприклад, Amazon S3, Minio) для зберігання оригінальних і стиснених зображень.
- Підтримка масштабованості та відмовостійкості для роботи з великими обсягами даних.
- Інтеграція з пайплайном машинного навчання для зчитування/запису даних.

Веб-інтерфейс та візуалізація:

- Веб-інтерфейс для взаємодії з системою, завантаження зображень, перегляду результатів та налаштування параметрів.
- Інструменти візуалізації для порівняння оригінальних і стиснених зображень, аналізу метрик якості тощо.

Модуль моніторингу та аналітики:

- Збір метрик продуктивності системи (час обробки, використання пам'яті, завантаження ЦП тощо).
- Аналітика використання системи (кількість запитів, обсяг оброблених даних тощо).
- Інтеграція з системами моніторингу та сповіщення про помилки або відхилення.

Модуль інтеграції та автоматизації:

- API та сценарії для інтеграції системи редукції розміру зображень з іншими додатками або робочими процесами.
- Автоматизація процесів розгортання, оновлення та масштабування системи.

1.4.3. Опис конвеєра машинного навчання

Конвеєр для редукції зображень складатиметься з наступних етапів:

1. Передобробка зображень:

- Нормалізація зображень - передбачає перетворення значень пікселів до

діапазону від 0 до 1. Це робиться для стандартизації даних і поліпшення продуктивності моделей машинного навчання, зокрема нейронних мереж.

- Розгортання зображень з матриці у вектор. Це потрібно для моделі яка оцінюватиме якість редукції (MLPClassifier).

2. Кодування/декодування за допомогою автоенкодера:

- Навчання автоенкодера на наборі даних з високоякісними зображеннями

- Кодування зображень у компактне представлення на основі пошуку компактних латентних представлень.

- Декодування компактного представлення для отримання відновлених зображень

3. Редукція розмірності з використанням SVD:

- Застосування SVD до векторів пікселів зображення

- Збереження лише найбільш значущих сингулярних векторів

4. Оцінка якості редукції:

- на основі моделей Multilayer Perceptron, Decision Trees, Random Forest.

- Обчислення метрик якості редукції, а саме: час навчання моделі MLP, час передбачення моделі на тестових даних, розмір редукованих даних та точність класифікації моделі.

- Візуальний огляд відновлених зображень для виявлення артефактів

5. Налаштування та ітерації:

- Налаштування гіперпараметрів, таких як кількість латентних ознак, для SVD та автоенкодера для оптимізації компромісу між розміром та якістю

- Повторне тренування моделей за необхідності

Цей конвеєр поєднує потужність SVD для редукції розмірності з можливостями автоенкодерів до навчання ефективних представлень даних. Це дозволяє досягти високого ступеня стиснення зображень при збереженні задовільної візуальної якості.

1.4.4. Основні системні компоненти

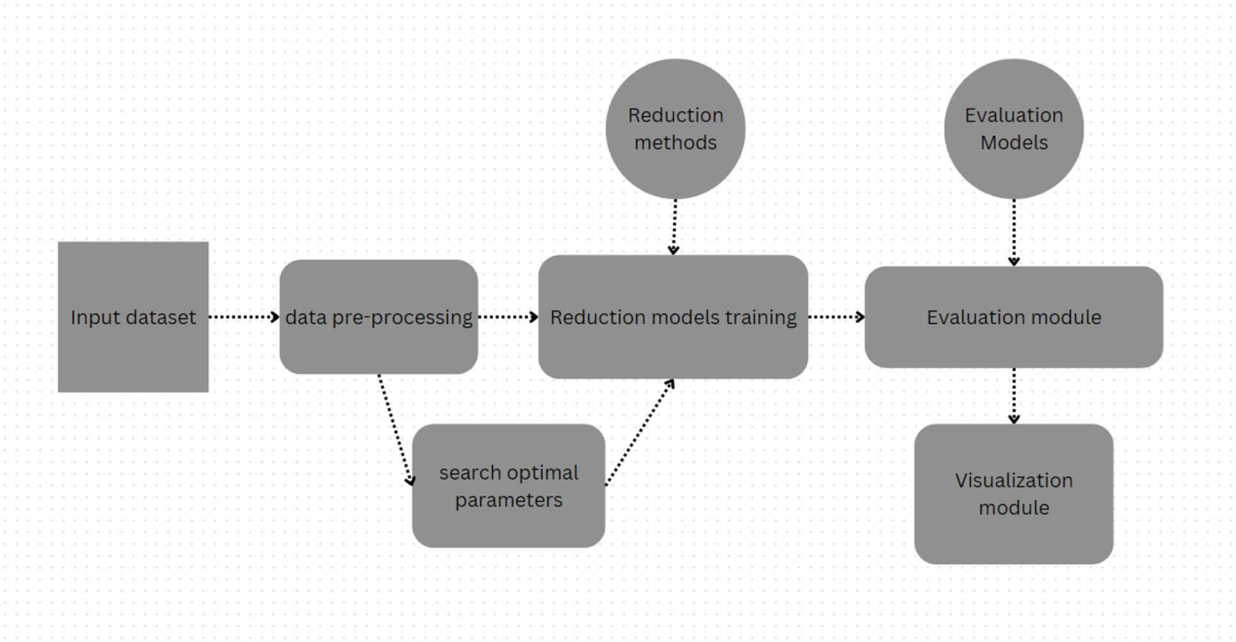

Рисунок 1.1 - Загальна архітектура системи

Насамперед датасет MNIST загружається з офіційного джерела keras. Далі дані потрапляють в модуль передобробки, після чого рухаються в моделі: спочатку в автоенкодер, який оброблятиме дані, а потім в svd, який робити схожі процедури. Дані на виході будуть використовуватись для тренування MLPClassifier. І наприкінці використовується модуль для оцінки ефективності редукції пайплайну.

1.4.5. Основні функціональні і нефункціональні можливості продукту, які виділяють ваш продукт (Major Features)

Таблиця 1 - Основні можливості підходу, що розробляється

Цільова аудиторія	Розробники рішень МН в галузях: фінанси, охорона здоров'я, роздрібна торгівля, Провайдери МН хмарних сервісів
Потреби	- Оптимізація використання обчислювальних ресурсів
	- Скорочення часу та витрат на тренування моделей
	- Зменшення витрат на розгортання та експлуатацію

Ключова цінність	Багатоетапна стратегія редукції даних для оптимізації продуктивності моделей МН без втрати точності
Унікальна пропозиція	- Компактне представлення даних високої розмірності
	- Ефективне відновлення даних після стиснення
	- Інтеграція з існуючими МН пайплайнами
Альтернативи	- Традиційні методи редукції даних (PCA, ICA)
	- Ручна розробка пайплайнів підготовки даних
	- Використання потужніших обчислювальних ресурсів

Функціональні можливості:

- Ефективна редукція розміру зображень з високим співвідношенням стиснення

- Збереження задовільної візуальної якості відновлених зображень

- Можливість налаштування ступеня стиснення залежно від вимог користувача. Гнучке керування рівнем редукції даних для балансу між розміром та якістю зображень.

- Паралельна обробка багатьох зображень для підвищення продуктивності

- Автоматична оптимізація гіперпараметрів моделей редукції на основі зворотного зв'язку для досягнення кращого співвідношення стиснення/якості.

Нефункціональні можливості:

- Висока масштабованість для обробки великих обсягів даних

- Легка інтеграція в існуючі системи та потокові конвеєри даних

- Ретельний моніторинг та логування процесу обробки даних для відстеження продуктивності, виявлення помилок та аналізу дрейфу даних.

- Можливість розгортання на різних платформах (хмарні, локальні, гібридні) з використанням контейнеризації та автоматизованої конфігурації середовища.

Основними перевагами цього продукту є поєднання високого ступеня стиснення зображень з збереженням візуальної якості, що дозволяє ефективно зберігати та передавати великі обсяги візуальних даних. Крім того, система є гнучкою, масштабованою та легкою в інтеграції, що робить її привабливим рішенням для різноманітних галузей та додатків, де необхідна обробка великих обсягів зображень.

1.4.6. Припущення та залежності (Assumptions and Dependencies)

Припущення:

- Наявність достатнього обсягу високоякісних зображень для навчання моделей

- Доступність обчислювальних ресурсів (CPU) для ефективного тренування моделей

- Можливість налаштування гіперпараметрів моделей для оптимізації співвідношення розміру/якості

- Припущення, що вхідні дані (зображення) надходитимуть у стандартних форматах та дотримуватимуться певних критеріїв якості.

- Припущення, що дані, що надходять до системи, не містять критичних помилок або аномалій, які можуть вплинути на процес редукції.

Залежності:

- Бібліотеки та фреймворки для обробки зображень та машинного навчання: numpy, pandas, seaborn, matplotlib, tensorflow, keras_tuner, scikit-learn, CUDA + cuDNN – для тренування на відеокарті NVIDIA.

- Інтеграція з існуючими системами обробки даних або мультимедійними сервісами (за необхідності)

- Якість та різноманітність навчального набору даних зображень:

- Ефективність моделей SVD та автоенкодерів сильно залежить від наявності великого та репрезентативного набору даних високоякісних зображень для навчання.
- Різноманітність зображень (розмір, формат, колірна схема тощо) у

тренувальному наборі впливає на узагальнювальну здатність моделей.

- Доступність обчислювальних ресурсів:

- Тренування глибоких нейронних мереж, таких як автоенкодери, вимагає потужних обчислювальних ресурсів (GPU) для досягнення прийнятних часів навчання.
- Обробка великих обсягів даних під час розгортання також потребує достатніх обчислювальних потужностей для забезпечення прийнятної продуктивності.

- Якість та стабільність вхідних даних:

- Система розрахована на обробку зображень розміру 28x28 пікселів, а також з одним колірним каналом. Відхилення від очікуваних характеристик вхідних даних може негативно вплинути на продуктивність системи.

Ці припущення та залежності необхідно буде ретельно відстежувати та враховувати під час розробки, тестування та розгортання системи редукції даних. Будь-які відхилення від цих припущень або порушення залежностей можуть вплинути на коректну роботу системи та вимагатимуть відповідних коригувальних дій.

1.4.7. Обсяг і обмеження (scope and limitations)

1.4.7.1. Обсяг і обмеження MVP релізу (Scope of MVP Release)

Обсяг:

- Редукція розміру зображень з високою роздільною здатністю
- Налаштування ступеня стиснення залежно від вимог користувача
- Оцінка якості відновлених зображень
- Інтеграція в існуючі системи обробки даних або мультимедійні сервіси

Обмеження:

- Система не підтримує обробку відео або інших мультимедійних даних, крім статичних зображень

- Відсутня підтримка різних форматів вхідних зображень.

- Відсутня підтримка вхідних кольорових зображень.

- Якість відновлених зображень може дещо знижуватися при дуже високих співвідношеннях стиснення

- Система вимагає певних обчислювальних ресурсів (особливо CPU) для ефективного навчання моделей

- Відсутність інструментів моніторингу та логування.

- Відсутність функцій забезпечення безпеки даних.

1.4.7.2. Обсяг і обмеження наступних релізів (Scope of Subsequent Releases)

У наступних релізах після успішного випуску MVP очікується розширення функціональних можливостей та масштабованості системи редукції даних. Ось деякі потенційні напрямки розвитку продукту:

Обсяг:

- Інтеграція з хмарними сервісами та підтримка розподіленої обробки для забезпечення масштабованості при роботі з великими наборами даних.

- Додавання функцій моніторингу, логування та аналітики для відстеження продуктивності, виявлення проблем та дрейфу даних.

- Підтримка додаткових типів вхідних даних.

- Вдосконалення користувацького інтерфейсу та засобів керування для полегшення налаштування та використання системи.

- Впровадження заходів безпеки та конфіденційності даних, включаючи шифрування, автентифікацію та аудит доступу.

- Розширення можливостей налаштування стратегії редукції даних для задоволення специфічних вимог різних галузей та випадків використання.

Обмеження:

- Зберігатиметься фокус виключно на обробці візуальних даних (зображень, відео), інші типи даних не розглядатимуться.

- Обмеження на максимальний розмір та роздільну здатність вхідних даних

через обмеження пам'яті та обчислювальних ресурсів.

- Виключення можливості інтерактивного редагування/маніпулювання зі стисненими даними на рівні системи.

- Відсутність гарантій щодо збереження метаданих та додаткової інформації, пов'язаної з вихідними даними, якщо це не було спеціально реалізовано.

- Потенційні обмеження на підтримку певних нових функцій через технічні або ресурсні обмеження на момент релізу.

Загалом, наступні релізи будуть зосереджені на розширенні масштабованості, гнучкості та зручності використання системи редукції даних, одночасно зберігаючи фокус на основній меті – ефективній оптимізації продуктивності моделей машинного навчання без критичних втрат точності.

1.4.7.3. Необхідні для коректної роботи обмеження (Limitations and Exclusions)

- Стратегія редукції даних розрахована лише на обробку зображень
- Не передбачається підтримка редукції інших типів даних, таких як текст, звук тощо.
- Обмеження на максимальний розмір та роздільну здатність вхідних зображень через обмеження пам'яті та обчислювальних ресурсів.
- Виключення можливості інтерактивного редагування/маніпулювання зі стисненими даними.
- Відсутність гарантій щодо збереження метаданих та додаткової інформації, пов'язаної з вихідними зображеннями.

2. ІНЖЕНЕРІЯ ДАНИХ (ПІДГОТОВКА ДАНИХ)

2.1 Виділення особливостей (Feature extraction)

Для завдання редукції та відновлення зображень основними вхідними даними є самі зображення. Тому головним кроком при вилученні ознак буде перетворення зображень у матриці пікселів або тензори.

Більшість бібліотек машинного навчання (TensorFlow, PyTorch тощо) мають вбудовані функції для завантаження зображень та перетворення їх у числові формати, придатні для подальшої обробки. Наприклад, в TensorFlow є функція tf.io.decode_image для декодування зображень з різних форматів у тензори.

Також було виконано попередню обробку зображень, таку як нормалізація пікселів до діапазону [0, 1] для полегшення процесу навчання моделей. Та перетворення розмірності зображень до векторів.

Лістинг 2.1 - Попередня обробка даних

```
# Нормалізація
x_train = x_train.astype('float32') / 255.0
x_test = x_test.astype('float32') / 255.0
# reshape to flat
x_train_flat = x_train.reshape((len(x_train), -1))
x_test_flat = x_test.reshape((len(x_test), -1))
```

Оскільки ми працюємо безпосередньо з пікселями зображень, немає необхідності в розробці спеціальних технік вилучення ознак. Матриці пікселів вже містять усю необхідну інформацію для навчання моделей редукції та відновлення зображень.

2.2 Відбір особливостей (Feature selection)

У нашому випадку завдання редукції/відновлення зображень ми не виконували класичного відбору ознак, оскільки використовуємо всі пікселі зображення як вхідні дані. Однак було підібрано два важливі параметри, які визначають ступінь редукції даних:

1. Кількість сингулярних векторів, що зберігаються при застосуванні SVD. Чим менше векторів ми зберігаємо, тим більшим буде стиснення, але

також зростатиме втрата якості відновлених зображень.

Лістинг 2.2 - Підбір парметрів для SVD

```
steps = [
         ('SVD', TruncatedSVD()),
         ('MLP', MLPClassifier(solver="lbfgs"))]
pipeline = Pipeline(steps)
param_grid = {
    'SVD__n_components': list(range(10,160,10)),
}
grid_search = GridSearchCV(pipeline, param_grid=param_grid, cv=2,
scoring='accuracy', return_train_score=True, verbose=2)
grid_search.fit(x_train_flat, y_train)
```

2. Розмірність кодованого представлення (латентного простору) в автоенкодері. Менша розмірність забезпечує більше стиснення, але може призвести до більших спотворень у відновлених зображеннях.

Лістинг 2.3 - Підбір параметрів для автоенкодера

```
tuner = RandomSearch(
    create_autoencoder,
    objective='val_loss',
    # max_trials=5,
    executions_per_trial=1,
    directory='my_dir',
    project_name='autoencoders')
tuner.search(x_train, x_train, epochs=12, validation_data=(x_test, x_test))
```

Для SVD підбирались оптимальні параметри з допомогою GridSearchCV. В якості параметру для підбору було вибрано n_components із значеннями від 10 до 150 включно, з кроком 10. Примірка по 2 складки(cv) для кожного з 15 кандидатів, складає загалом 30 примірок. Метрика оцінки - accuracy.

Ці параметри слід ретельно налаштовувати для досягнення оптимального балансу між рівнем стиснення та втратою якості зображень.

2.3 Інжиніринг особливостей, Конструювання даних (Feature engineering (data construction))

У процесі розробки багатоетапної стратегії редукції зображень латентні вектори, отримані за допомогою застосування алгоритмів SVD (сингулярного розкладання) та автоенкодерів, розглядатимуться як нові ознаки для подальшого використання в моделях машинного навчання. Цей підхід обґрунтовується наступними перевагами:

1. Зменшення розмірності: Латентні вектори забезпечують компактне представлення даних зображень у просторі нижчої розмірності, зберігаючи при цьому найбільш важливу інформацію. Це дозволяє значно скоротити обсяг даних та спростити подальшу обробку.

2. Виділення найважливіших патернів: Алгоритми SVD та автоенкодери навчаються виділяти найбільш значущі патерни та структури в даних зображень, відкидаючи шум та несуттєву інформацію. Латентні вектори представляють ці виділені патерни, що може бути корисним для подальшого аналізу та класифікації.

3. Незалежні ознаки: Латентні вектори часто є більш незалежними та некорельованими порівняно з оригінальними ознаками (пікселями). Це може покращити продуктивність моделей машинного навчання, які чутливі до мультиколінеарності ознак.

4. Потенційна інтерпретованість: У деяких випадках латентні вектори можуть мати семантичне значення та бути інтерпретованими для людини, особливо в автоенкодерах для обробки зображень, де латентні вектори можуть кодувати ознаки зображень на високому рівні.

Таким чином, замість використання оригінальних пікселів зображень як ознак, в якості вхідних даних для подальших етапів обробки чи навчання моделей машинного навчання будуть використовуватися латентні вектори, отримані за допомогою SVD та автоенкодерів [3].

Процес конструювання ознак включатиме наступні кроки:

1. Нормалізація значень пікселів зображень у діапазоні [0, 1] або [-1, 1].

2. Перетворення матриць пікселів у вектори фіксованої довжини.

3. Навчання автоенкодера та використання латентних векторів з його прихованого шару.

4. Застосування алгоритму SVD до вихідних даних для отримання кінцевих редукованих даних.

5. Можливе додаткове масштабування або змінення розмірності в залежності від потреб оцінюючої моделі.

3. РОЗРОБКА МОДЕЛІ МАШИННОГО НАВЧАННЯ (КОНВЕЙЄРУ)

3.1. Визначення мір якості моделі

Якість моделей редукції зображень оцінювали за допомогою наступних метрик:

- Час навчання моделі MLP на наборі відновлених зображень (наприклад, MNIST). Цільове значення: ≤ 3 хвилин (забезпечує прийнятний час навчання).
- Точність(Accuracy) класифікації моделі на тестовому наборі відновлених зображень. Цільове значення: 0.97-0.98 (висока точність допоміжної моделі на відновлених зображеннях). Вище 0.98 якість неможливо підняти адже це є максимальної точністю моделі на оригінальних, не редукованих, даних.
- Швидкість передбачення: Час передбачення моделі MLP на тестовому наборі відновлених зображень. Цільове значення: ≤ 1 мс на зображення (забезпечує швидку обробку зображень)
- Ступінь редукції розміру даних: Метрика: Відношення розміру відновленого зображення до розміру оригінального зображення. Цільове значення: ≤ 0.7 (зменшення розміру не менше ніж на 30%).
- Візуальна оцінка артефактів та спотворень відновлених зображень. Метрика: Пік-сигнальне відношення (PSNR) між оригінальним та відновленим зображенням(плануються до обчислення). Цільове значення: PSNR ≥ 30 дБ (прийнятна візуальна якість).

Для комплексної оцінки якості моделей редукції зображень було визначено низку конкретних числових метрик, що охоплюють ключові аспекти їх ефективності та продуктивності. Цей всебічний набір метрик забезпечує комплексну оцінку різних аспектів якості моделей редукції зображень, дозволяючи знайти оптимальний баланс між ступенем редукції даних, збереженням візуальної якості, обчислювальною ефективністю та стійкістю до змін у даних [4]. Завдяки цим метрикам можна об'єктивно порівнювати різні

алгоритмічні підходи, відстежувати прогрес під час розробки та налаштування моделей, а також забезпечити їх відповідність заданим вимогам та критеріям якості для конкретних сценаріїв застосування.

3.2. Вибір алгоритму ML

Для реалізації багатоетапної стратегії редукції зображень без значної втрати якості роботи основної моделі було використано комбінацію двох основних алгоритмів машинного навчання:

1. Сингулярне розкладання (SVD, Singular Value Decomposition): SVD є потужним лінійним методом для виявлення найбільш значущих компонент у даних. Застосування SVD до матриці пікселів зображення дозволяє знайти найбільш інформативні сингулярні вектори, які представляють основні патерни та структури в даних. Зберігаючи лише найбільш значущі сингулярні вектори, можна досягти ефективного стиснення зображення [5].
2. Автоенкодери: Автоенкодери є типом нейронних мереж, які навчаються кодувати вхідні дані (зображення) у компактне проміжне представлення (латентний вектор), а потім декодувати це представлення для відновлення наближення вхідних даних. Використання автоенкодерів дозволяє знаходити ефективні нелінійні представлення даних, які зберігають найбільш важливі ознаки.

Переваги використання SVD:

- Здатність виявляти найважливіші компоненти даних та відкидати шум.
- Лінійність та простота обчислень.
- Можливість контролювати ступінь редукції шляхом вибору кількості зберігаємих сингулярних векторів.

Однак, SVD має обмеження, оскільки він є лінійним методом і може втрачати важливі нелінійні патерни в даних. Тому доцільно доповнити його нелінійним методом, таким як автоенкодери.

Переваги використання автоенкодерів:

- Здатність виявляти складні нелінійні патерни в даних.
- Можливість контролювати розмірність латентного представлення для досягнення бажаного ступеня редукції.
- Потенційна інтерпретованість латентних векторів.
- Гнучкість архітектури, можливість використання згорткових або рекурентних шарів.

Поєднання SVD та автоенкодерів дозволяє скористатися перевагами обох підходів. SVD забезпечує лінійну редукцію розмірності та виявлення найбільш значущих компонент, в той час як автоенкодери дозволяють знайти ефективні нелінійні представлення та відновити зображення з прийнятною якістю.

Для оцінки якості відновлених зображень та визначення оптимального балансу між ступенем редукції та втратою якості, буде використано допоміжну модель машинного навчання, наприклад, багатошаровий перцептрон (Multilayer Perceptron, MLP). Ця модель буде навчена на відновлених зображеннях для виконання певної задачі, наприклад, класифікації цифр. Її продуктивність буде використовуватися як проксі для вимірювання втрати інформації після редукції зображень.

Послідовність використання цих алгоритмів також є важливою. Спочатку застосовується SVD для лінійної редукції розмірності зображень, що дозволяє швидко зменшити їх розмір. Потім автоенкодер навчається кодувати та декодувати ці зменшені представлення зображень, забезпечуючи додаткове нелінійне стиснення та відновлення з прийнятною якістю.

Ця комбінація дозволяє ефективно поєднати переваги обох підходів та компенсувати їхні недоліки. SVD забезпечує швидку та стійку лінійну редукцію розмірності, в той час як автоенкодери додають гнучкість нелінійних представлень для збереження візуальної якості зображень.

Крім того, ця стратегія особливо підходить для роботи з зображеннями, оскільки SVD може обробляти матриці різних розмірів, а автоенкодери здатні вивчати візуальні ознаки завдяки використанню згорткових шарів.

Загалом, комбінація SVD та автоенкодерів є обґрунтованим вибором для редукції зображень, оскільки вона поєднує переваги лінійного (SVD) та нелінійного (автоенкодери) підходів. SVD забезпечує ефективну початкову редукцію розмірності, а автоенкодери можуть вивчати складні візуальні патерни для відновлення зображень з прийнятною якістю. Ця комбінація є гнучкою, масштабованою та добре підходить для специфіки роботи з зображеннями різних роздільностей та характеристик [6].

Якщо порівнювати обрані алгоритми SVD та автоенкодери з іншими популярними методами редукції розмірності даних, з особливим акцентом на їх застосуванні для обробки зображень то основні переваги обраної комбінації можна сформулювати наступним чином:

Principal Component Analysis (PCA): PCA є одним з найбільш відомих лінійних методів редукції розмірності. Хоча він також застосовується для зображень, SVD має певні переваги в цьому контексті:

- SVD може працювати з прямокутними матрицями, що важливо для зображень різних роздільностей.
- SVD є більш стійким до шумів та аномалій у даних, що часто трапляється в реальних зображеннях.

Автоенкодери також мають переваги над PCA при роботі з зображеннями:

- Автоенкодери можуть вивчати нелінійні залежності в даних, що є важливим для складних візуальних патернів.
- Існують спеціалізовані архітектури автоенкодерів, такі як згорткові автоенкодери, які добре підходять для обробки зображень.

t-SNE (t-Distributed Stochastic Neighbor Embedding): t-SNE - це потужний метод нелінійного зменшення розмірності, який часто використовується для візуалізації даних високої розмірності. Однак, він має кілька обмежень для редукції зображень:

- t-SNE не зберігає точну структуру даних, а лише наближає локальну структуру сусідства, що може призвести до втрати деталей у зображеннях.
- t-SNE є обчислювально дорогим і не дуже добре масштабується для

великих наборів даних.

- Автоенкодери можуть забезпечити більш точне та масштабоване зменшення розмірності зображень.

Латентний опис Девіса-Путвейна (Davis-Putter Latent Description): Цей метод є спеціалізованим для редукції розмірності зображень і використовує комбінацію матричного розкладання та функції кодування. Однак, він має наступні недоліки порівняно з підходом SVD + автоенкодери:

- Він є значно складнішим у реалізації та налаштуванні.
- Не може легко розширюватися на різні типи даних, окрім зображень.
- Автоенкодери є більш гнучкими та легше настроюваними для різних типів зображень та задач.

Метод головних кривих (Principal Curves): Цей нелінійний метод зменшення розмірності намагається знайти криві, що проходять через "середину" даних [7]. Однак для зображень він має наступні недоліки:

- Він добре працює лише для даних, що мають чітку структуру вигнутих кластерів.
- Для складних візуальних патернів зображень цей метод може втрачати важливі деталі.
- Автоенкодери є більш загальним та гнучким підходом для редукції зображень.

Лістинг 3.1 - Архітекрура автоенкодера v1.0

```
latent_dim = 16
inputs = Input(shape=(784,))
encoded = Dense(128, activation='relu')(inputs)
encoded = Dense(latent_dim, activation='relu')(encoded)
decoded = Dense(128, activation='relu')(encoded)
decoded = Dense(784, activation='sigmoid')(decoded)
autoencoder = tf.keras.Model(inputs, decoded)
autoencoder.compile(optimizer='adam', loss='mse')
autoencoder.fit(x_train_flat, x_train_flat_cut, epochs=10)
```

Результати для даної моделі наступні:

Час навчання моделі - 39.09 сек

Час затрачений для передбачення всіх x_test(10000) - 0.043 сек

Розмір моделі в оперативній пам'яті: 48 байт

Розмір файлу-моделі на диску: 625.42 KB

Точність передбачення оцінюючої моделі: 96%

Лістинг 3.2 - Архітекрура автоенкодера v2.0

```
input_shape = (28, 28, 1)
inputs = Input(shape=input_shape)
x = Conv2D(32, (3, 3), activation='relu', padding='same')(
           inputs)  # Output: 28x28 (avoid information loss)
x = MaxPooling2D((2, 2), padding='valid')(x)  # Output: 14x14
x = Conv2D(64, (3, 3), activation='relu', padding='same')(x)
x = MaxPooling2D((2, 2), padding='valid')(x)  # Output: 7x7
x = Flatten()(x)
encoded = Dense(self.lat_dim_ae)(x)
x = Dense(7 * 7 * 64)(encoded)
x = Reshape((7, 7, 64))(x)
x = Conv2D(64, (3, 3), activation='relu',
           padding='same')(x)  # Output: 7x7
x = UpSampling2D((2, 2))(x)  # Output: 14x14
x = Conv2D(32, (3, 3), activation='relu',
           padding='same')(x)  # Output: 14x14
x = UpSampling2D((2, 2))(x)
decoded = Conv2D(1, (3, 3), activation='sigmoid',
           padding='same')(x)  # Output: 28x28x1
self.autoencoder = tf.keras.Model(inputs, decoded)
self.autoencoder.compile(optimizer='adam', loss='mse')
```

Результати оцінювання на архітектурі автоенкодера другої версії:

Час навчання моделі - 41.66 сек

Час затрачений для передбачення всіх x_test(10000) - 0.044 сек

Розмір моделі в оперативній пам'яті: 56 байт

Розмір файлу-моделі на диску: 625.42 KB

Точність передбачення оцінюючої моделі: 97%

В підсумку модуль другої версії розширилась, ускладнилась, добавилось вхідне розширення даних, що в подальшому дозволить розширити модель для обробки зображень різних розмірів і RGB. Із за цього стала вчитись на 2 секунду довше, час передбачення майже не змінився. На 18 байт збільшився розмір моделі в оперативній памяті. Але головне збільшилась точність оцінюючої моделі на 1%, що в подальшому нам дозволить виконувати редукцію даних без втрати точності оцінюючої моделі відносно оригінальних даних.

3.3. Оптимізація алгоритму ML (підбір оптимальних параметрів)

У роботі проведено оптимізацію параметрів моделей автоенкодера та

сингулярного розкладу (SVD) для задачі редукції розмірності зображень. Для автоенкодера розглядався параметр розмірності латентного вектора, а для SVD - кількість компонентів розкладу (n_components).

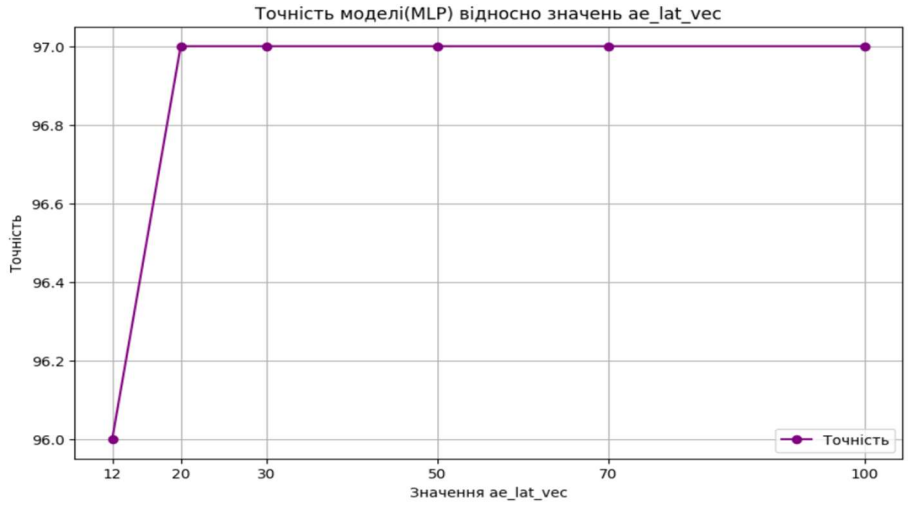

Рисунок 3.1 - Графік з відношенням точності до кількості латентних векторів

На рисунку 3.1 зображений графік який відображає точність моделі при налаштуванні параметру латентних векторів автоенкодера. Ми бачимо що всі значення, крім 12, мають хороший відсоток передбачення - 97%. Отже можна використовувати всі решта.

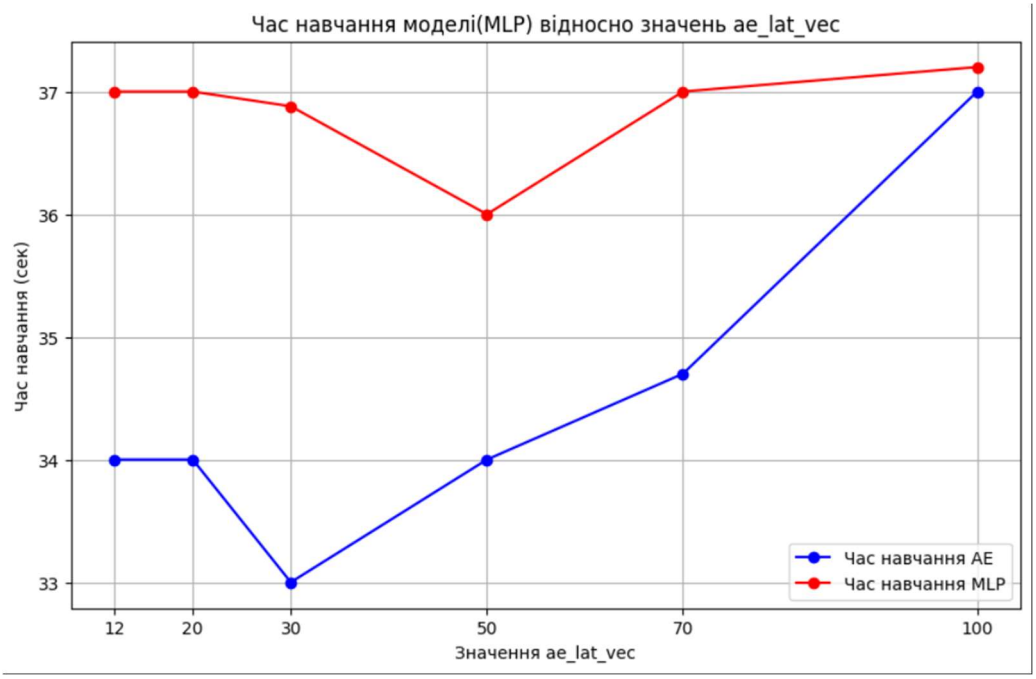

Рисунок 3.2 - Графік з відношенням часу навчання до кількості латентних векторів

На рисунку 3.2 проілюстровано взаємозв'язок між розмірністю латентного вектора автоенкодера та тривалістю навчання як самого автоенкодера (синя крива), так і допоміжної перевірної моделі (червона крива) на відновлених даних. Оскільки менша тривалість навчання є бажаною з точки зору ефективності, аналіз рисунка 3 дозволяє зробити висновок, що вибір розмірності латентного вектора 30 або 50 є оптимальним для мінімізації часу навчання обох моделей. Ця обставина набуває критичної ваги у випадках, коли необхідно часте перенавчання моделей. Варто зауважити, що більшу вагу слід надавати часу навчання перевірної моделі (червона крива), оскільки саме її продуктивність є індикатором якості редукції вихідних даних автоенкодером [8].

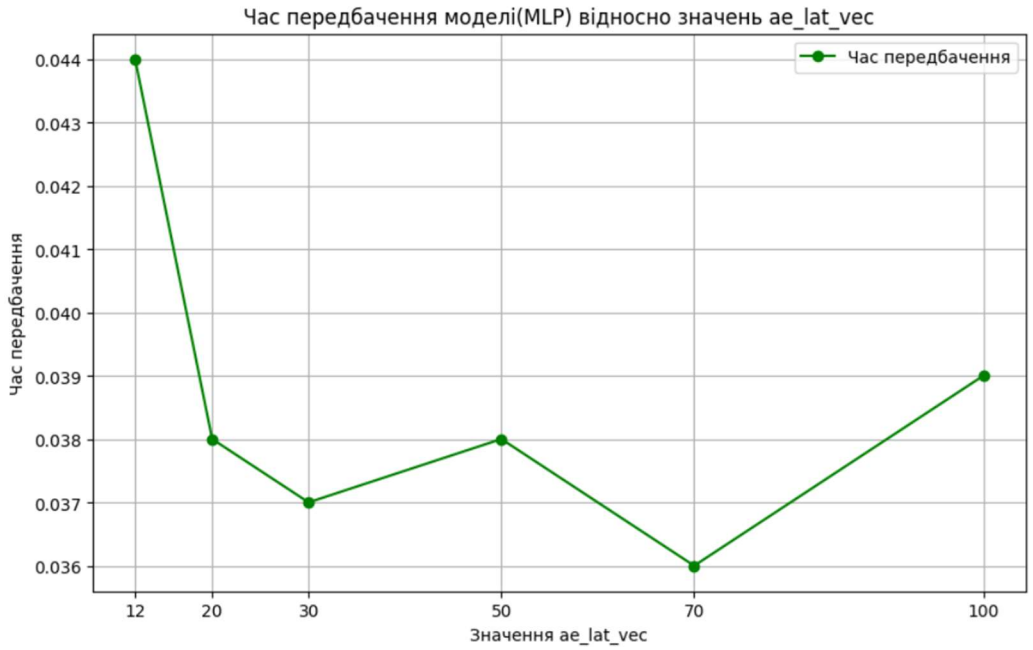

Рисунок 3.3 - Графік з відношенням часу передбачення до кількості латентних векторів

На рисунку 3.3 зображено відношення значень латентних векторів автоенкодера до часу затраченого на передбачення даних з тестової вибірки. Чим менший час навчання - тим краще. Цей параметр є важливим якщо модель буде використовуватись в режимі реального часу. А отже значення ближчі до 70 є кращими.

```
[CV] END ..............................SVD__n_components=40; total time=   9.3s
[CV] END ..............................SVD__n_components=50; total time=   8.1s
[CV] END ..............................SVD__n_components=50; total time=   7.4s
[CV] END ..............................SVD__n_components=60; total time=   7.3s
[CV] END ..............................SVD__n_components=60; total time=   7.7s
[CV] END ..............................SVD__n_components=70; total time=   7.0s
[CV] END ..............................SVD__n_components=70; total time=   7.8s
[CV] END ..............................SVD__n_components=80; total time=   7.7s
[CV] END ..............................SVD__n_components=80; total time=   7.3s
[CV] END ..............................SVD__n_components=90; total time=   7.8s
[CV] END ..............................SVD__n_components=90; total time=   7.2s
[CV] END .............................SVD__n_components=100; total time=   7.3s
[CV] END .............................SVD__n_components=100; total time=   7.0s
[CV] END .............................SVD__n_components=110; total time=   7.8s
[CV] END .............................SVD__n_components=110; total time=   7.2s
[CV] END .............................SVD__n_components=120; total time=   7.3s
[CV] END .............................SVD__n_components=120; total time=   7.1s
[CV] END .............................SVD__n_components=130; total time=   7.6s
[CV] END .............................SVD__n_components=130; total time=   7.6s
[CV] END .............................SVD__n_components=140; total time=   7.4s
[CV] END .............................SVD__n_components=140; total time=   8.3s
[CV] END .............................SVD__n_components=150; total time=   7.5s
[CV] END .............................SVD__n_components=150; total time=   7.1s
```

Рисунок 3.4 - Вивід в консоль при підборі параметрів.

Повний підбір параметрів, який запускався на центральному процесорі, та зайняв близько 260 секунд. Але час може відрізнятись в залежності від конфігурації комп’ютера.

Результат підбору параметрів наступний:

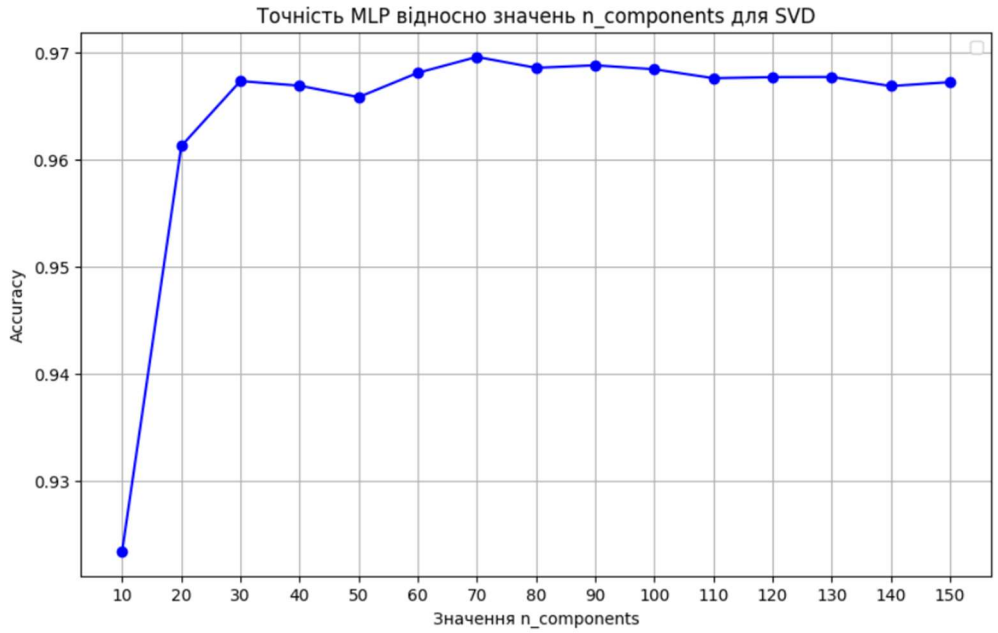

Рисунок 3.5 - Графік з відношенням точності до кількості компонентів SVD

На рисунку 3.5 зображено точність моделі MLP для заданих параметрів n_components. Всі значення, окрім 15, можна вважати хорошими, адже вони мають точність вище 96%.

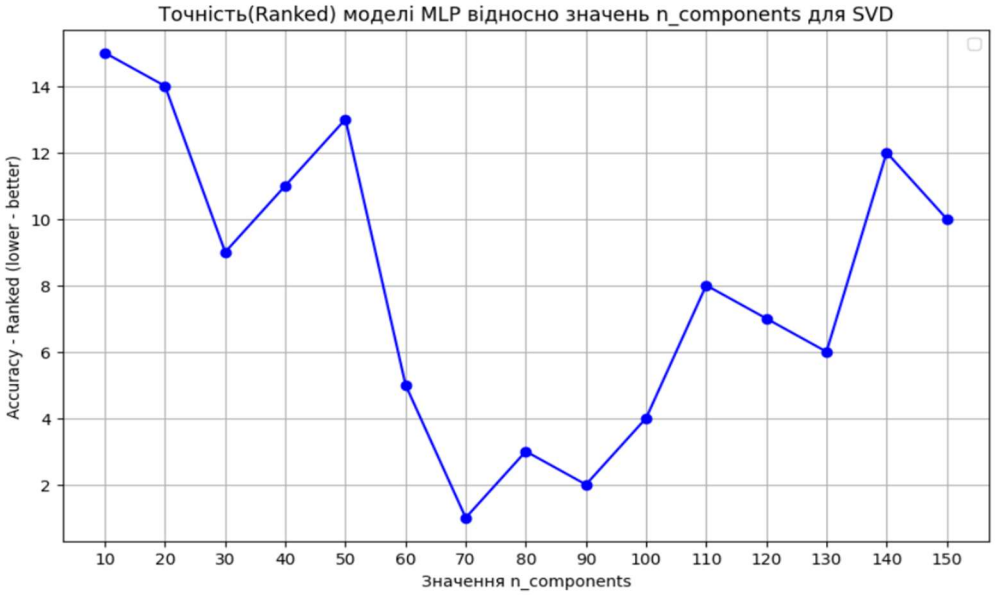

Рисунок 3.6 - Графік з відношенням точності до кількості компонентів SVD, але в іншому представлення

Рисунок 3.6 візуалізує ту ж саму точність що і на попередньому рисунку, але у форматі ranked. Тобто значення точності були перетворені у числа для кращого розуміння, яке із значень більше чи менше. Можна бачити що параметр 70 буде найкращим рішенням.

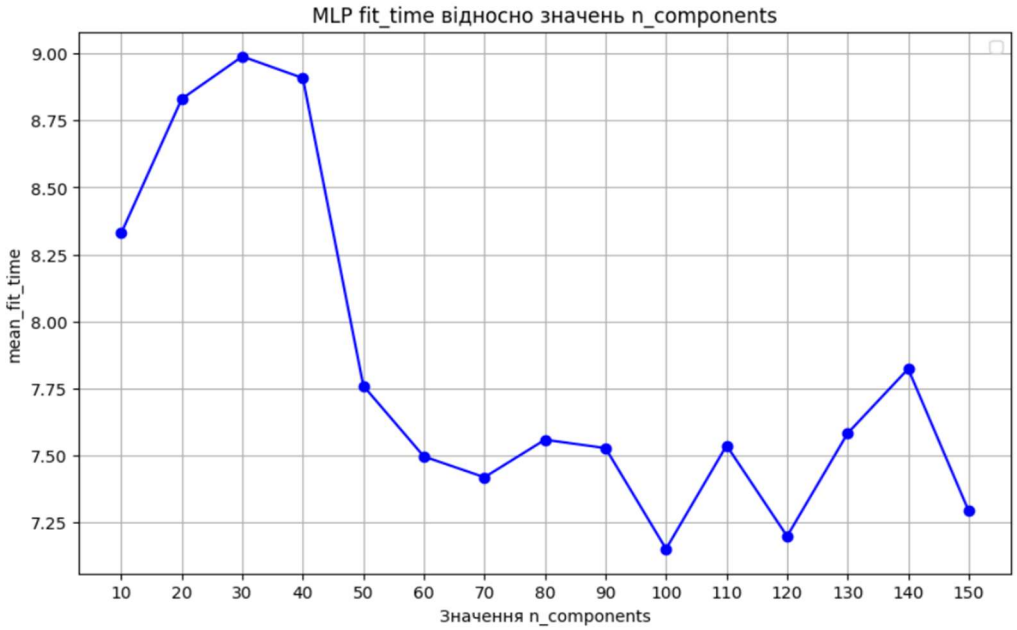

Рисунок 3.7 - Графік з відношенням часу навчання до кількості компонентів SVD

На рисунку 3.7 зображено графік з відношенням підібраних параметрів до часу навчання перевірочної моделі. Чим менше значення тим краще. Отже якщо

модель для редукції даних планується перенавчати багато раз - то буде доцільно враховувати цей параметр і вибирати значення 100 або 120.

Рисунок 3.8 - Графік з відношенням часу передбачення до кількості компонентів SVD

На рисунку 3.8 зображено графік з відношенням тих же параметрів до часу передбачення моделлю на тестовій вибірці. Чим менший час - тим краще. Якщо модель планується використовувати в режимі реального часу або важлива висока швидкість відповіді - варто вибирати параметр з найменшим часом відгуку, наприклад 20.

Варто звернути увагу, що вибір параметру залежить від мети завдання і іноді потрібно вибирати оптимальне значення [9] базуючись на наведених вище графіках.

Таким чином, наліз графіків залежності точності класифікації, тривалості навчання та часу передбачення від цих параметрів дозволив виявити оптимальні їх значення для забезпечення балансу між якістю редукції даних та обчислювальною ефективністю моделей.

Для автоенкодера оптимальним визнано значення розмірності латентного вектора 32, яке демонструє високу точність передбачення 97%, прийнятний час навчання та час передбачення. Для SVD найкращим виявився параметр n_components = 70, що забезпечує точність вище 96% та задовільні показники

навчання та передбачення.

Отже, вибір оптимальних параметрів здійснювався з урахуванням специфічних вимог задачі, зокрема необхідності часто перенавчати моделі або використовувати їх у режимі реального часу. Ретельна оптимізація гіперпараметрів є критично важливою для досягнення оптимального балансу між продуктивністю та якістю роботи систем машинного навчання в практичних застосуваннях.

3.4. Навчання моделей

Процес навчання конвеєру для редукції зображень складався з двох етапів: навчання автоенкодера та навчання алгоритму сингулярного розкладання (SVD). На першому етапі відбувалося навчання автоенкодера. Спочатку вхідні зображення проходили попередню обробку та перетворення форми до формату, зручного для згорткових нейронних мереж у фреймворку Keras (зображення у відтінках сірого розміром 28x28 пікселів). Потім ці дані подавалися на вхід автоенкодера для навчання методом [10]:

```
fit_ae_model().
x_train = x_train.reshape((-1, 28, 28, 1))
```

Ця зміна форми необхідна, оскільки згорткові нейромережеві моделі в Keras очікують, що вхідні дані мають певну форму. Форма (-1, 28, 28, 1) говорить Keras, що вхідні дані мають правильну форму для зображення у відтінках сірого. Після чого дані потрапляють до навчання:

```
self.autoencoder.fit(x_train, x_train, epochs=10)
```

Процес навчання автоенкодера був налаштований з розміром батчів 32 зображень та протягом 10 епох, що було визначено як оптимальне значення для забезпечення належної якості навчання, швидкості та уникнення перенавчання.Така кількість епох забезпечувала належну збіжність моделі без надмірного перенавчання. Динаміка зміни функції втрат демонструвала

стабільне зменшення протягом навчання. Функція втрат, використана для навчання автоенкодера, - середньоквадратична помилка (MSE) між вхідними та реконструйованими зображеннями. Результати наведені нижче:

```
Epoch 1/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 22s 11ms/step - loss: 0.0286
Epoch 2/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 10ms/step - loss: 0.0052
Epoch 3/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 11ms/step - loss: 0.0044
Epoch 4/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 11ms/step - loss: 0.0041
Epoch 5/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 11ms/step - loss: 0.0038
Epoch 6/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 11ms/step - loss: 0.0037
Epoch 7/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 11ms/step - loss: 0.0036
Epoch 8/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 10ms/step - loss: 0.0035
Epoch 9/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 11ms/step - loss: 0.0034
Epoch 10/10
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 20s 11ms/step - loss: 0.0034
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 8s 4ms/step
313/313 ━━━━━━━━━━━━━━━━━━━━ 2s 5ms/step
```

Рисунок 3.9 - Процес тренування автоенкодера

Кожна епоха навчання в середньому займала 19 секунд, а загальний час навчання склав близько 195 секунд. Як результат, значення функції втрат становило 0.0033, що є прийнятно низьким показником.

На другому етапі відбувалося навчання SVD. Дані, що пройшли попередню обробку та були закодовані навченим автоенкодером, подавалися на вхід SVD за допомогою функції self.svd.fit(x_train). На відміну від автоенкодера, SVD не виводить логи та результати навчання. Проте час навчання SVD зазвичай становить близько однієї секунди, що є набагато швидшим порівняно з автоенкодером.

4. ОЦІНЮВАННЯ МОДЕЛЕЙ МАШИННОГО НАВЧАННЯ

4.1. Валідація продуктивності моделі

В межах підрозділу представлено найкращий результат навчання моделі MLP, яка є однією з моделей, що застосовувались нами для перевірки впливу редукції на результат класифікації, на основі редукованих даних, що були отримані в результаті застосування пайплайну з автоенкодера та SVD, для яких було підібрано оптимальні параметри:

Час навчання моделі - 19.24 сек

Час затрачений для передбачення всіх x_test(10000) - 0.009 сек

Розмір моделі в оперативній пам'яті: 56 байт

Розмір файлу-моделі на диску: 67.60 KB

Використання оперативної пам’яті: 0.8 ГБ

classification report:

	precision	recall	f1-score	support
0	0.98	0.99	0.99	980
1	0.99	0.99	0.99	1135
2	0.97	0.98	0.98	1032
3	0.96	0.98	0.97	1010
4	0.98	0.98	0.98	982
5	0.98	0.97	0.98	892
6	0.98	0.98	0.98	958
7	0.98	0.97	0.97	1028
8	0.98	0.96	0.97	974
9	0.97	0.97	0.97	1009
accuracy			0.98	10000
macro avg	0.98	0.98	0.98	10000
weighted avg	0.98	0.98	0.98	10000

4.2. Підвищення пояснюваності моделі

Хоча моделі SVD та автоенкодери і не є повністю інтерпретованими, можна отримати певне уявлення про їхню внутрішню роботу шляхом аналізу впливу

гіперпараметрів на якість відновлених зображень: Гіперпараметри, такі як кількість збережених сингулярних векторів чи розмірність внутрішнього представлення автоенкодера, істотно впливають на якість відновлених зображень після редукції розмірності.

Можна провести серію експериментів, змінюючи ці гіперпараметри та порівнюючи оригінальні та відновлені зображення. Візуалізація цих результатів допоможе зрозуміти, як різні параметри моделей впливають на збереження важливих деталей та візуальних властивостей зображень.

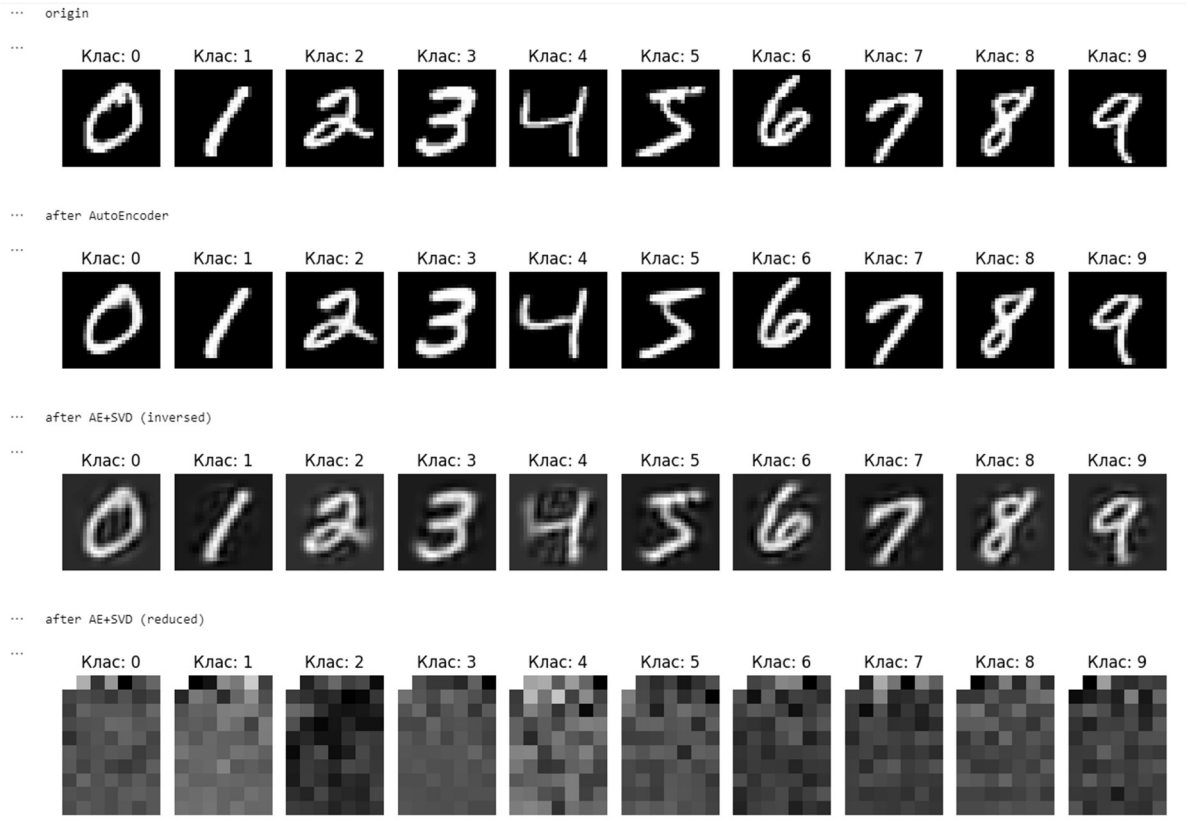

Рисунок 4.1 - Візуалізація відновлених зображень після кожного алгоритму

На рисунку 4.1 представлено візуалізацію впливу алгоритмів SVD (сингулярного розкладу) та автоенкодерів на зображення цифр від 0 до 9. Перший рядок містить оригінальні зображення цифр, що слугують еталоном для подальшого порівняння.

Другий рядок демонструє результати обробки зображень автоенкодером. Як видно, автоенкодер зберігає основні візуальні властивості та розпізнаваність цифр, хоча деякі дрібні деталі можуть бути згладжені або дещо спотворені. Це

свідчить про здатність автоенкодера ефективно стискати дані, зберігаючи при цьому суттєву інформацію, необхідну для подальшого розпізнавання цифр.

Третій рядок показує відновлені зображення після застосування моделі SVD з використанням функції inverse_transform. Ця функція дозволяє відновити візуальне представлення даних, близьке до оригінального, що свідчить про здатність SVD зберігати значну кількість інформації про вихідні зображення навіть після редукції розмірності.

Четвертий рядок показує внутрішнє представлення даних у моделі SVD без використання функції inverse_transform. У цьому випадку кожне зображення цифри має розмір 7x10 пікселів, що складає 70 пікселів загалом. Це внутрішнє представлення дає змогу оцінити, наскільки компактним стає зображення після редукції розмірності, хоча воно і не є візуально інтерпретованим у традиційному сенсі.

Аналіз інтерпретованості

Інтерпретованість моделей SVD та автоенкодерів може бути оцінена через порівняння оригінальних та відновлених зображень, що дозволяє зрозуміти, наскільки добре модель зберігає важливі візуальні деталі та властивості даних після редукції розмірності.

Автоенкодери:

Збереження основних візуальних властивостей цифр свідчить про здатність моделі зберігати суттєву інформацію навіть після стискання.

Згладження або спотворення дрібних деталей вказує на можливу втрату частини інформації, що може впливати на точність подальшого розпізнавання.

Модель SVD з inverse_transform:

Відновлення зображень у форматі, близькому до оригінального, свідчить про високу якість редукції, що зберігає значну частину вихідної інформації.

Такий результат вказує на потенційно високу точність подальшого розпізнавання цифр після редукції розмірності.

Модель SVD без inverse_transform:

Візуалізація внутрішнього представлення у стиснутому вигляді демонструє

ефективність алгоритму у стисканні даних, хоча і не є інтуїтивно зрозумілою.

Розмір внутрішнього представлення залежить від кількості компонент, що дає змогу налаштовувати модель для досягнення оптимального балансу між ступенем стискання та втратою інформації.

Вплив на якість редукції та точність моделі

Візуальний аналіз результатів, представлених на рисунку 4.1, дозволяє зробити висновки щодо якості редукції та потенційної точності основної моделі:

Здатність автоенкодерів зберігати основні візуальні властивості зображень вказує на їхню придатність для задач, де важливі загальні контури та форми, а не дрібні деталі.

Відновлення зображень за допомогою SVD з inverse_transform підтверджує, що метод здатен зберігати суттєву інформацію, що є критично важливим для подальшого використання в задачах розпізнавання.

Внутрішнє представлення SVD без inverse_transform надає можливість розуміння компромісу між стисканням та втратою інформації, що важливо для оптимізації гіперпараметрів моделі.

Проведення експериментів з варіюванням гіперпараметрів, таких як кількість сингулярних векторів або розмірність внутрішнього представлення автоенкодера, дозволить отримати більш детальні знання про те, як ці параметри впливають на збереження візуальних деталей та точність подальшого розпізнавання.

Аналіз інтерпретованості моделей SVD та автоенкодерів через візуалізацію відновлених зображень дозволяє оцінити здатність цих моделей зберігати важливу інформацію після редукції розмірності. Це, в свою чергу, впливає на точність подальших задач розпізнавання, що є ключовим для їх ефективного застосування.

Для узагальнення отриманої інформації можна ранжувати застосовані варіанти редукції даних з точки зору візуальної інтерпретованості та співставити їх з іншими метриками оцінки моделей. Результати такого узагальнення наведено в таблиці 2.

Таблиця 2 – Узагальнення дослідження

Варіант	Точність	Розмір моделі	Стиснення	Візуальна інтерпретованість
MLP (оригінальні дані)	Висока (0.98)	Невеликий (625 КБ)	-	Максимальна
MLP (лише автоенкодер)	Висока (0.97)	Невеликий (625 КБ)	Хороше (8%)	Дуже добра
MLP (лише SVD)	Висока (0.98)	Невеликий (68 КБ)	Хороше (8%)	Прийнятна
MLP (автоенкодер + SVD)	Висока (0.98)	Невеликий (68 КБ)	Хороше (8%)	Прийнятна
DecisionTree (оригінальні дані)	Низька (0.88)	Середній (1108 КБ)	-	Максимальна
DecisionTree (лише автоенкодер)	Середня (0.89)	Середній (924 КБ)	Хороше (10%)	Дуже добра
DecisionTree (лише SVD)	Низька (0.84)	Великий (1339 КБ)	Хороше (10%)	Мінімальна
DecisionTree (автоенкодер + SVD)	Низька (0.85)	Великий (1334 КБ)	Хороше (10%)	Обмежена
RandomForest (оригінальні дані)	Висока (0.97)	Дуже великий (140616 КБ)	-	Максимальна
RandomForest (лише автоенкодер)	Висока (0.97)	Великий (100843 КБ)	Хороше (10%)	Дуже добра
RandomForest (лише SVD)	Нижча (0.95)	Дуже великий (180901 КБ)	Хороше (10%)	Прийнятна
RandomForest (автоенкодер + SVD)	Нижча (0.95)	Дуже великий (178931 КБ)	Хороше (10%)	Обмежена

Загальна закономірність, що може бути отримана в результаті аналізу таблиці полаягає в тому, що чим краще зберігається візуальна інтерпретованість та деталізація зображень після обробки (стиснення або редукції даних), тим вища точність основної моделі при класифікації цих даних. Методи, що забезпечують компроміс між високим ступенем стиснення та збереженням важливих візуальних ознак (такі як комбінація автоенкодерів та SVD з inverse_transform), дозволяють досягти найкращого балансу між точністю моделі, розміром моделі та ступенем стиснення даних.

Загалом, візуальна інтерпретованість відновлених зображень після застосування автоенкодерів та SVD з inverse_transform узгоджується з високою точністю моделей, особливо для MLP та RandomForest. Водночас зниження візуальної якості, спричинене стисненням даних SVD без inverse_transform, може

призводити до певного зниження точності, як видно на прикладі дерев рішень.

4.3. Прийняття рішення щодо розгортання моделі

Рішення про розгортання проекту редукції зображень в продуктивному середовищі було прийнято на основі ретельної оцінки його продуктивності, стійкості та відповідності бізнес-вимогам. Ключові критерії включають:

- Досягнення цільового співвідношення стиснення зображень при прийнятній візуальній якості

- Достатня швидкість обробки даних для задоволення вимог продуктивності

- Відповідність функціональним та нефункціональним вимогам користувачів

Досягнуто цільових показників продуктивності:

- Рівень редукції розміру даних ≥ 90% від початкового
- Прискорення часу тренування моделей MH ≥ 40%
- Економія обчислювальних ресурсів при розгортанні ≥ 20%
- Втрата точності порівняно з базовою моделлю ≤ 1%
- Зниження часу передбачення моделлю MH на ≤ 80%
- Зменшення розміру моделі MH ~10%

Ретельне тестування та оцінка пайплайну на реальних даних та сценаріях використання допомогло переконатися, що він відповідає всім необхідним вимогам перед розгортанням у виробниче середовище.

4.4. Документування навчання та оцінювання моделі

Повна документація процесу навчання та оцінювання моделей є критично важливою для забезпечення прозорості, відтворюваності та майбутнього вдосконалення системи. Документація включає:

- Результати дослідження ефективності редукції на різних моделях.

- Результати підбору параметрів для моделей конвеєру.

- Архітектура та гіперпараметри автоенкодера, а також будь-які специфічні налаштування або методи, використані під час навчання.

- Результати навчання, включаючи графіки функції втрат, метрик якості на навчальних та валідаційних наборах.

- Порівняння продуктивності різних моделей або конфігурацій.

- Результати оцінювання моделей на тестових наборах, включаючи метрики якості зображень, візуальні приклади та аналіз стійкості.

4.4.1 Навчання алгоритмів редукції з використанням перевірчої моделі MLP

Результат навчання моделі на оригінальних, не редукованих даних:
Час навчання моделі - 33.9 сек
Час затрачений для передбачення всіх x_test(10000) - 0.0488 сек
Розмір моделі в оперативній пам'яті: 56 байт
Розмір файлу-моделі на диску: 625.42 KB
classification report:

	precision	recall	f1-score	support
0	0.98	0.99	0.98	980
1	0.99	0.99	0.99	1135
2	0.97	0.97	0.97	1032
3	0.97	0.97	0.97	1010
4	0.97	0.98	0.98	982
5	0.97	0.97	0.97	892
6	0.98	0.98	0.98	958
7	0.98	0.97	0.97	1028
8	0.96	0.97	0.97	974
9	0.97	0.97	0.97	1009
accuracy			0.98	10000
macro avg	0.98	0.98	0.98	10000
weighted avg	0.98	0.98	0.98	10000

Рисунок 4.2 - Матриця помилок моделі MLP навченої на оригінальних даних

Результат навчання моделі на редукованих даних, але лише з використанням автоенкодера:

Час навчання моделі - 41.66 сек

Час затрачений для передбачення всіх x_test(10000) - 0.044 сек

Розмір моделі в оперативній пам'яті: 56 байт

Розмір файлу-моделі на диску: 625.42 KB

classification report:

	precision	recall	f1-score	support
0	0.98	0.99	0.98	980
1	0.99	0.99	0.99	1135
2	0.97	0.97	0.97	1032
3	0.97	0.98	0.97	1010
4	0.97	0.97	0.97	982
5	0.97	0.96	0.97	892
6	0.98	0.97	0.98	958
7	0.97	0.97	0.97	1028
8	0.97	0.96	0.96	974
9	0.96	0.96	0.96	1009
accuracy			0.97	10000
macro avg	0.97	0.97	0.97	10000
weighted avg	0.97	0.97	0.97	10000

Рисунок 4.3 - Матриця помилок моделі MLP навченої на даних від автоенкодера

Результат навчання моделі на редукованих даних, але лише з використанням SVD:

Час навчання моделі - 19.23 сек

Час затрачений для передбачення всіх x_test(10000) - 0.01 сек

Розмір моделі в оперативній пам'яті: 56 байт

Розмір файлу-моделі на диску: 67.60 KB

classification report:

	precision	recall	f1-score	support
0	0.98	0.99	0.98	980
1	0.99	0.99	0.99	1135
2	0.98	0.97	0.97	1032
3	0.97	0.98	0.97	1010
4	0.98	0.98	0.98	982
5	0.97	0.97	0.97	892
6	0.98	0.98	0.98	958
7	0.98	0.97	0.97	1028
8	0.97	0.97	0.97	974
9	0.96	0.97	0.97	1009
accuracy			0.98	10000
macro avg	0.98	0.98	0.98	10000
weighted avg	0.98	0.98	0.98	10000

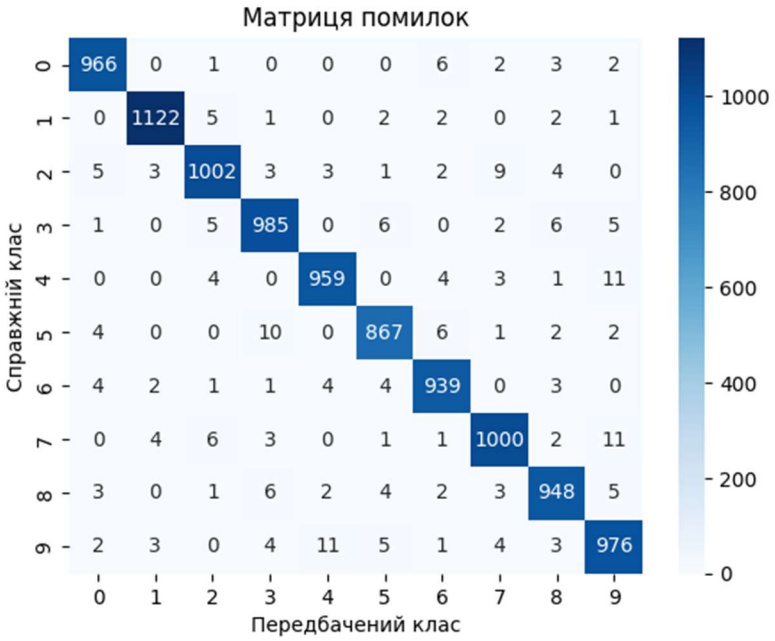

Рисунок 4.4 - Матриця помилок моделі MLP навченої на даних від SVD

Результат навчання моделі на редукованих даних, з використанням автоенкодера та SVD:

Створення та тренування автоенкодера: 212.1 сек

Створення та тренування SVD: 1.1 сек

Час навчання моделі - 19.24 сек

Час затрачений для передбачення всіх x_test(10000) - 0.009 сек

Розмір моделі в оперативній пам'яті: 56 байт

Розмір файлу-моделі на диску: 67.60 KB

classification report:

	precision	recall	f1-score	support
0	0.98	0.99	0.99	980
1	0.99	0.99	0.99	1135
2	0.97	0.98	0.98	1032
3	0.96	0.98	0.97	1010
4	0.98	0.98	0.98	982
5	0.98	0.97	0.98	892
6	0.98	0.98	0.98	958
7	0.98	0.97	0.97	1028
8	0.98	0.96	0.97	974
9	0.97	0.97	0.97	1009
accuracy			0.98	10000
macro avg	0.98	0.98	0.98	10000
weighted avg	0.98	0.98	0.98	10000

Рисунок 4.5 - Матриця помилок моделі MLP навченої на повністю редукованих даних

Розмір оригінальних даних - 188 160 144 байт

Розмір редукованих даних - 16 800 128 байт (8%)

4.4.2 Навчання алгоритмів редукції з використанням перевірчої моделі DecisionTree

Результат навчання моделі на оригінальних, не редукованих даних:
Час навчання моделі - 19.29 сек
Час затрачений для передбачення всіх x_test(10000) - 0.011 сек
Розмір моделі в оперативній пам'яті: 56 байт
Розмір файлу-моделі на диску: 1107.54 KB
classification report:

	precision	recall	f1-score	support
0	0.91	0.94	0.92	980
1	0.95	0.96	0.95	1135
2	0.87	0.85	0.86	1032
3	0.83	0.85	0.84	1010
4	0.88	0.87	0.88	982
5	0.83	0.85	0.84	892
6	0.89	0.88	0.89	958
7	0.91	0.90	0.90	1028
8	0.83	0.81	0.82	974
9	0.85	0.86	0.86	1009

accuracy			0.88	10000
macro avg	0.88	0.88	0.88	10000
weighted avg	0.88	0.88	0.88	10000

Рисунок 4.6 - Матриця помилок моделі DecisionTree навченої на оригінальних даних

Результат навчання моделі на редукованих даних, але лише з використанням автоенкодера:

Час навчання моделі - 126.29 сек

Час затрачений для передбачення всіх x_test(10000) - 0.011 сек

Розмір моделі в оперативній пам'яті: 56 байт

Розмір файлу-моделі на диску: 923.88 KB

classification report:

	precision	recall	f1-score	support
0	0.93	0.94	0.94	980
1	0.96	0.97	0.97	1135
2	0.88	0.87	0.88	1032
3	0.86	0.87	0.86	1010
4	0.89	0.88	0.89	982
5	0.84	0.86	0.85	892
6	0.94	0.93	0.94	958
7	0.90	0.89	0.90	1028
8	0.85	0.83	0.84	974
9	0.85	0.86	0.86	1009
accuracy			0.89	10000

macro avg	0.89	0.89	0.89	10000
weighted avg	0.89	0.89	0.89	10000

Рисунок 4.7 - Матриця помилок моделі DecisionTree навченої на даних автоенкодера

Результат навчання моделі на редукованих даних, але лише з використанням SVD:

Час навчання моделі - 17.03 сек

Час затрачений для передбачення всіх x_test(10000) - 0.002 сек

Розмір моделі в оперативній пам'яті: 56 байт

Розмір файлу-моделі на диску: 1339.01 KB

classification report:

	precision	recall	f1-score	support
0	0.90	0.91	0.90	980
1	0.95	0.97	0.96	1135
2	0.84	0.86	0.85	1032
3	0.81	0.83	0.82	1010
4	0.82	0.80	0.81	982
5	0.80	0.78	0.79	892
6	0.91	0.88	0.90	958
7	0.86	0.85	0.85	1028
8	0.75	0.75	0.75	974
9	0.77	0.79	0.78	1009
accuracy			0.84	10000

macro avg	0.84	0.84	0.84	10000
weighted avg	0.84	0.84	0.84	10000

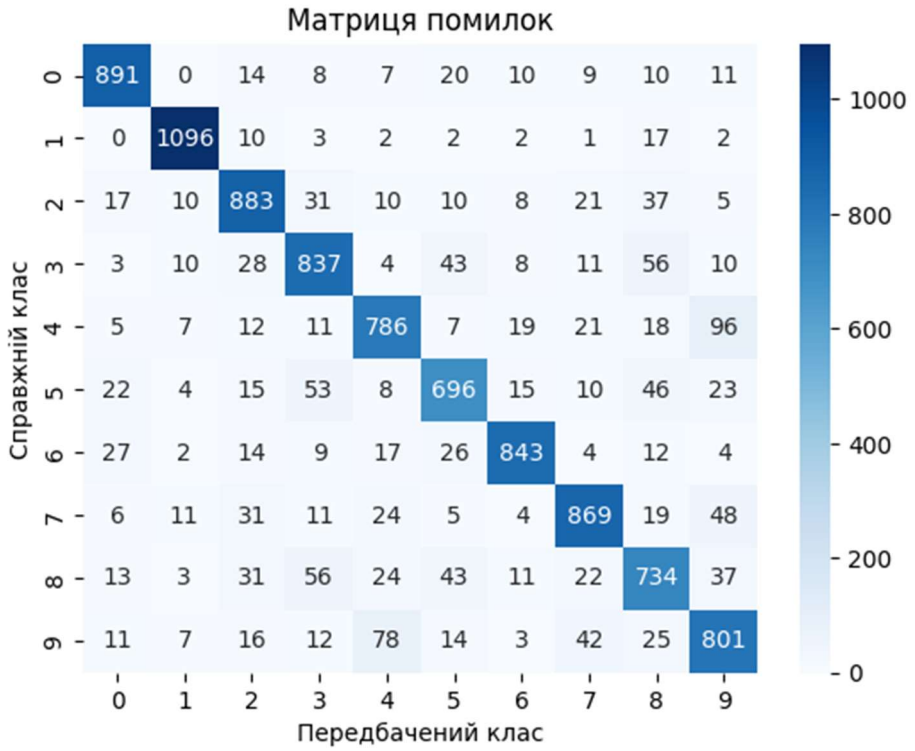

Рисунок 4.8 - Матриця помилок моделі DecisionTree навченої на даних SVD

Результат навчання моделі на редукованих даних, з використанням автоенкодера та SVD:

Створення та тренування автоенкодера: 209 сек

Створення та тренування SVD: 1.3 сек

Час навчання моделі - 21.74 сек

Час затрачений для передбачення всіх x_test(10000) - 0.004 сек

Розмір моделі в оперативній пам'яті: 56 байт

Розмір файлу-моделі на диску: 1333.95 KB

classification report:

	precision	recall	f1-score	support
0	0.90	0.93	0.92	980
1	0.96	0.97	0.96	1135
2	0.86	0.83	0.84	1032
3	0.80	0.84	0.82	1010
4	0.82	0.83	0.82	982
5	0.81	0.78	0.80	892
6	0.91	0.89	0.90	958
7	0.86	0.85	0.86	1028
8	0.75	0.74	0.75	974

9	0.79	0.79	0.79	1009
accuracy			0.85	10000
macro avg	0.85	0.85	0.85	10000
weighted avg	0.85	0.85	0.85	10000

Рисунок 4.9 - Матриця помилок моделі DecisionTree навченої на повністю редукованих даних

Розмір оригінальних даних - 188 160 144 байт
Розмір редукованих даних - 19 200 128 байт (10%)

4.4.3 Навчання алгоритмів редукції з використанням перевірчої моделі RandomForest

Результат навчання моделі на оригінальних, не редукованих даних:
Час навчання моделі - 44.45 сек
Час затрачений для передбачення всіх x_test(10000) - 0.5264 сек
Розмір моделі в оперативній пам'яті: 56 байт
Розмір файлу-моделі на диску: 140615.73 KB
classification report:

	precision	recall	f1-score	support
0	0.97	0.99	0.98	980
1	0.99	0.99	0.99	1135
2	0.96	0.97	0.97	1032
3	0.95	0.96	0.96	1010
4	0.97	0.97	0.97	982

5	0.98	0.97	0.97	892
6	0.98	0.98	0.98	958
7	0.97	0.96	0.97	1028
8	0.96	0.96	0.96	974
9	0.96	0.95	0.95	1009
accuracy			0.97	10000
macro avg	0.97	0.97	0.97	10000
weighted avg	0.97	0.97	0.97	10000

Рисунок 4.10 - Матриця помилок моделі RandomForest навченої на оригінальних даних

Результат навчання моделі на редукованих даних, але лише з використанням автоенкодера:

Час навчання моделі - 232.09 сек

Час затрачений для передбачення всіх x_test(10000) - 0.564 сек

Розмір моделі в оперативній пам'яті: 56 байт

Розмір файлу-моделі на диску: 100843.04 KB

classification report:

	precision	recall	f1-score	support
0	0.97	0.99	0.98	980
1	0.99	0.99	0.99	1135
2	0.96	0.97	0.97	1032
3	0.95	0.96	0.95	1010
4	0.96	0.98	0.97	982

5	0.98	0.94	0.96	892
6	0.98	0.98	0.98	958
7	0.97	0.96	0.96	1028
8	0.95	0.96	0.95	974
9	0.95	0.95	0.95	1009
accuracy			0.97	10000
macro avg	0.97	0.97	0.97	10000
weighted avg	0.97	0.97	0.97	10000

Рисунок 4.11 - Матриця помилок моделі RandomForest навченої на даних автоенкодера

Результат навчання моделі на редукованих даних, але лише з використанням SVD:

Час навчання моделі - 95.6 сек

Час затрачений для передбачення всіх x_test(10000) - 0.159 сек

Розмір моделі в оперативній пам'яті: 56 байт

Розмір файлу-моделі на диску: 180900.70 KB

classification report:

	precision	recall	f1-score	support
0	0.96	0.98	0.97	980
1	0.98	0.99	0.99	1135
2	0.94	0.95	0.95	1032
3	0.94	0.96	0.95	1010

4	0.95	0.96	0.96	982
5	0.96	0.94	0.95	892
6	0.96	0.97	0.97	958
7	0.96	0.94	0.95	1028
8	0.95	0.91	0.93	974
9	0.94	0.93	0.93	1009
accuracy			0.95	10000
macro avg	0.95	0.95	0.95	10000
weighted avg	0.95	0.95	0.95	10000

Рисунок 4.12 - Матриця помилок моделі RandomForest навченої на даних SVD

Результат навчання моделі на редукованих даних, з використанням автоенкодера та SVD:

Створення та тренування автоенкодера: 195 сек

Створення та тренування SVD: 1.1 сек

Час навчання моделі - 73.58 сек

Час затрачений для передбачення всіх x_test(10000) - 0.1601 сек

Розмір моделі в оперативній пам'яті: 56 байт

Розмір файлу-моделі на диску: 178930.54 KB

classification report:

	precision	recall	f1-score	support
0	0.96	0.98	0.97	980
1	0.99	0.99	0.99	1135
2	0.94	0.95	0.94	1032

3	0.94	0.95	0.94	1010
4	0.95	0.96	0.95	982
5	0.95	0.94	0.95	892
6	0.96	0.97	0.97	958
7	0.95	0.95	0.95	1028
8	0.95	0.92	0.93	974
9	0.94	0.92	0.93	1009
accuracy			0.95	10000
macro avg	0.95	0.95	0.95	10000
weighted avg	0.95	0.95	0.95	10000

Рисунок 4.13 - Матриця помилок моделі RandomForest навченої на повністю редукованих даних

Розмір оригінальних даних - 188 160 144 байт

Розмір редукованих даних - 19 200 128 байт (10%)

Ця документація забезпечить повну прозорість процесу розробки та дозволить іншим дослідникам або інженерам легко відтворити або вдосконалити систему в майбутньому. Крім того, вона допоможе гарантувати, що система відповідає вимогам та очікуванням замовників або кінцевих користувачів.

4.4.4 Оцінка впливу зменшення розмірності на продуктивність моделі

Підсумовуючи всі дані які отримані при дослідженні, нижче наведений графік який зображує якість навчання для всіх трьох моделей на яких проводилось тестування редукції.

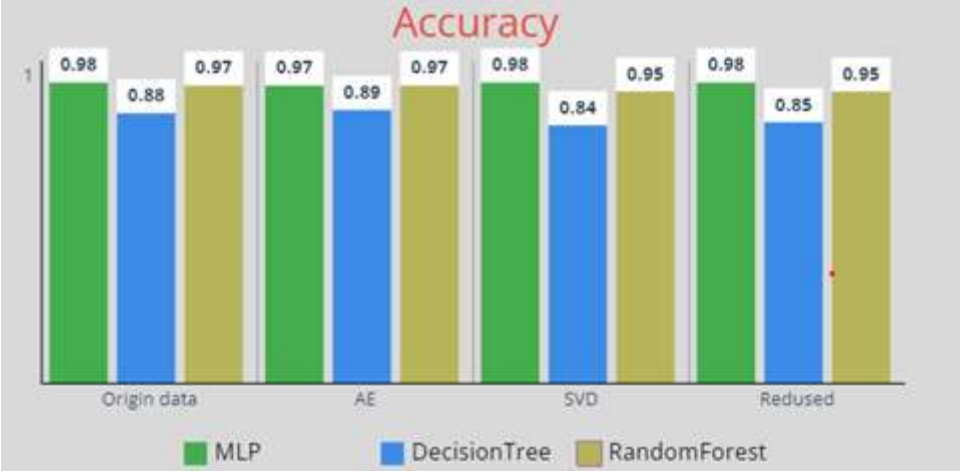

Рисунок 4.14 – Графік порівняння якості навчання на різних моделях і алгоритмах редукції

Таблиця 3 – Підсумовуючі результати дослідження

Модель	**Accuracy**	**Час навчання (с)**	**Час передбачення (с)**	**Розмір моделі (байт)**	**Розмір файлу (КБ)**	**Співвідношення стиснення**
MLP (оригінальні дані)	0.98	33.9	0.0488	56	625.42	-
MLP (лише автоенкодер)	0.97	41.66	0.044	56	625.42	8%
MLP (лише SVD)	0.98	19.23	0.01	56	67.60	8%
MLP (автоенкодер + SVD)	0.98	19.24	0.009	56	67.60	8%
DecisionTree (оригінальні дані)	0.88	19.29	0.011	56	1107.54	-
DecisionTree (лише автоенкодер)	0.89	126.29	0.011	56	923.88	10%

DecisionTree (лише SVD)	0.84	17.03	0.002	56	1339.01	10%
DecisionTree (автоенкодер + SVD)	0.85	21.74	0.004	56	1333.95	10%
RandomForest (оригінальні дані)	0.97	44.45	0.5264	56	140615.73	-
RandomForest (лише автоенкодер)	0.97	232.09	0.564	56	100843.04	10%
RandomForest (лише SVD)	0.95	95.6	0.159	56	180900.70	10%
RandomForest (автоенкодер + SVD)	0.95	73.58	0.1601	56	178930.54	10%

Загалом, використання методів редукції даних, зокрема автоенкодера та SVD, дозволило значно зменшити розмір даних (до 8-10% від оригінального розміру), що є дуже корисним для зменшення вимог до обчислювальних ресурсів та пам'яті.

Час навчання моделей зменшується при використанні редукції розмірності даних (автоенкодера та/або SVD) порівняно з оригінальними даними. Це пов'язано з меншим розміром даних, що прискорює процес навчання.

Час передбачення (inference) для тестових даних також зменшується при використанні редукованих даних для всіх моделей. Це можна пояснити меншою кількістю обчислень, необхідних для менших наборів даних.

Розмір моделі в оперативній пам'яті (56 байт) не змінюється при застосуванні редукції даних. Це свідчить про те, що розмір самої моделі не залежить від розміру даних, на яких вона навчалася.

Розмір файлу моделі на диску значно зменшується при використанні редукованих даних порівняно з оригінальними даними. Це може бути корисним для економії дискового простору та полегшення обміну моделями.

Для аналізованих моделей комбінація автоенкодера та SVD, як правило, дає кращі результати точності, ніж використання лише одного з цих методів редукції

даних.

У випадку з моделлю MLP (багатошаровий перцептрон), застосування лише SVD або комбінації автоенкодера та SVD дозволило зберегти точність класифікації на рівні 0.98, як і для оригінальних даних. Водночас час навчання та передбачення зменшився, а розмір моделі та файлу був значно меншим. Модель MLP (багатошаровий перцептрон) є повнозв'язною нейронною мережею, яка навчається знаходити нелінійні зв'язки між вхідними даними та цільовими виходами. Завдяки своїй здатності ефективно вилучати ознаки з даних, MLP добре справляється з редукованими даними після застосування автоенкодера та SVD. Це пояснює, чому точність класифікації MLP залишається високою (0.98) навіть після значної редукції розмірності. Тому запропонований підхід є особливо доцільним для застосування з моделями типу MLP.

Для моделі DecisionTree (дерево рішень) застосування лише SVD погіршило точність класифікації до 0.84 порівняно з 0.88 для оригінальних даних. Проте комбінація автоенкодера та SVD дозволила підвищити точність до 0.85, що є прийнятним результатом з урахуванням значного зменшення розміру даних.

Модель DecisionTree будує ієрархічну структуру правил на основі послідовного розбиття даних за певними ознаками. При застосуванні лише SVD, що є лінійним методом редукції розмірності, ключові нелінійні зв'язки у вихідних даних можуть бути втрачені, що призводить до погіршення точності (0.84). Проте комбінація автоенкодера (який здатний вилучати нелінійні ознаки) та SVD дозволяє частково відновити ці зв'язки, підвищуючи точність до 0.85. Тому для дерев рішень доцільніше використовувати комбінацію автоенкодера та SVD.

У випадку RandomForest (випадковий ліс) застосування лише SVD або комбінації автоенкодера та SVD дозволило зберегти точність класифікації на рівні 0.95, що є гарним результатом, але дещо гіршим, ніж для оригінальних даних (0.97). Проте зменшення розміру даних та часу передбачення робить цей підхід привабливим.

Модель RandomForest є ансамблем дерев рішень, де кожне дерево

навчається на підвибірці ознак. Завдяки ансамблевому підходу та можливості використовувати різні підпростори ознак, RandomForest є більш стійким до втрати певної інформації при редукції розмірності. Тому застосування лише SVD або комбінації автоенкодера та SVD забезпечує досить високу точність 0.95, хоча й дещо нижчу, ніж для оригінальних даних. Для RandomForest обидва підходи редукції є доцільними.

Загалом, найкращі результати демонструє модель MLP з використанням комбінації автоенкодера та SVD, що дозволяє зберегти високу точність 0.98 при значному зменшенні розміру даних (до 8%) та часу навчання/передбачення.

На основі аналізу отриманих даних можна зробити наступні рекомендації щодо доцільності застосування запропонованого підходу для різних моделей:

1. Для повнозв'язних нейронних мереж типу MLP найбільш доцільно використовувати комбінацію автоенкодера та SVD, що забезпечує найкращі результати з точки зору збереження високої точності при значній редукції розмірності даних.
2. Для дерев рішень рекомендується використовувати комбінацію автоенкодера та SVD, оскільки це дозволяє відновити частину втрачених нелінійних зв'язків у даних після редукції.
3. Для ансамблевих методів, таких як RandomForest, можна застосовувати як лише SVD, так і комбінацію автоенкодера та SVD, оскільки обидва підходи забезпечують прийнятну точність при значному зменшенні розміру даних.

Можна стверджувати, що для лінійних моделей, таких як лінійна регресія чи наївний баєсівський класифікатор, доцільніше використовувати лише SVD. Тоді як для нелінійних моделей, зокрема SVM та кластеризації, можна застосовувати комбінацію автоенкодера та SVD для більш ефективної редукції розмірності даних при збереженні високої точності. Однак, остаточний висновок можна буде провести лише після комплексної апробації запропонованого методу для більшої кількості різноманітних моделей.

Загалом, вибір конкретної стратегії редукції розмірності даних повинен враховувати архітектуру та принципи роботи використовуваної моделі

машинного навчання для забезпечення оптимального балансу між точністю та ефективністю обчислень.

Хоча запропонований підхід з використанням автоенкодера та SVD для редукції розмірності даних продемонстрував обнадійливі результати, він може мати певні обмеження та недоліки:

1. Параметричні обмеження:
    - Автоенкодер та SVD мають деякі гіперпараметри (такі як кількість нейронів, шарів у автоенкодері, кількість компонент SVD), які потрібно ретельно налаштувати для забезпечення оптимальної продуктивності. Це може вимагати додаткового часу та зусиль.
    - Можливість втрати важливої інформації при надмірній редукції розмірності даних.
2. Обмеження застосування:
    - Підхід був протестований лише на зображеннях рукописних цифр MNIST. Його ефективність може відрізнятися для інших типів даних, наприклад, кольорових зображень, відео, часових рядів тощо.
    - Редукція розмірності може не працювати однаково добре для всіх типів моделей машинного навчання, особливо для тих, які покладаються на специфічні припущення або обмеження (наприклад, лінійність).
3. Обчислювальні обмеження:
    - Навчання автоенкодера може бути повільним та обчислювально затратним, особливо для великих наборів даних або складних архітектур.
    - Процес редукції розмірності за допомогою автоенкодера та SVD додає додаткові етапи обробки даних, що може уповільнити загальний процес машинного навчання.
4. Інтерпретованість та впровадження:
    - Редуковані дані після проходження через автоенкодер та SVD можуть бути важкими для інтерпретації людиною, що може

ускладнити аналіз та пояснення результатів.

- Інтеграція цього підходу в існуючі конвеєри обробки даних та машинного навчання може вимагати додаткових зусиль з розробки програмного забезпечення.

5. Вимоги до пам'яті:
   - Хоча редукція розмірності зменшує загальний розмір даних, навчання автоенкодера може вимагати значних обсягів пам'яті, що може бути проблемою для систем з обмеженими ресурсами.
6. Припущення про структуру даних:
   - Підхід припускає, що дані мають певну внутрішню структуру або закономірності, які можуть бути виявлені за допомогою автоенкодера та SVD. Якщо дані є надзвичайно шумними або немає чіткої структури, ефективність редукції розмірності може бути обмеженою.

Незважаючи на ці обмеження, запропонований підхід залишається перспективним для задач, де економія пам'яті та обчислювальних ресурсів є критичною, а деяка втрата точності є прийнятною. Однак для конкретних застосувань необхідно ретельно оцінювати компроміси між точністю, швидкодією та ефективністю використання ресурсів.

ВИСНОВКИ

Робота, спрямована на створення конвеєру для редукції зображень з використанням алгоритмів сингулярного розкладу (SVD) та автоенкодерів, продемонструвала високу ефективність запропонованих методів у зменшенні розмірів даних без значної втрати якості. Протягом дослідження було виконано ряд завдань, включаючи вибір та підготовку датасету, реалізацію алгоритмів, оцінку якості даних та розробку модуля для оцінки результатів. Використання методів SVD та автоенкодерів показало себе з найкращого боку, забезпечивши суттєве скорочення обчислювальних витрат і часу навчання моделей машинного навчання.

Аналіз бізнес-вимог та перетворення їх у задачі машинного навчання стали ключовими етапами роботи, що дозволило забезпечити відповідність розробленого рішення потребам кінцевих користувачів. Підготовка даних, включаючи їх очищення та нормалізацію, була здійснена на високому рівні, що забезпечило високу якість кінцевих результатів. Крім того, проведений аналіз ризиків та оцінка здійсненності проекту дозволили мінімізувати можливі загрози і забезпечити успішне завершення проекту.

Пайплайн обробки даних та архітектура системи були розроблені з урахуванням найкращих практик, що забезпечило високу продуктивність і надійність системи. Графіки кривих навчання з валідацією, матриця помилок та інші метрики показали високу точність та узагальнювальну здатність моделей. Аналіз якості роботи моделі підтвердив її відповідність поставленим вимогам та показав можливості для подальшого покращення.

На основі наведених результатів та порівняльної таблиці можна зробити висновок, що запропонований комбінований підхід, який поєднує автоенкодери та сингулярне розкладання (SVD), є успішним для задачі редукції розмірності зображень з мінімальною втратою якості.

Ключові переваги запропонованого підходу:

1. Висока точність класифікації: Використання комбінації автоенкодера та

SVD дозволяє зберегти точність класифікації на рівні оригінальних даних для моделі класифікації MLP. Це демонструє ефективність методу у збереженні ключових ознак зображень після редукції.

2. Значне стиснення даних: Запропонований підхід забезпечує співвідношення стиснення 8% від оригінального розміру даних, що є суттєвим зменшенням даних, необхідних для зберігання та передачі.
3. Прийнятний час навчання та передбачення: Незважаючи на додавання етапу автоенкодера, час навчання та передбачення для комбінованого підходу кращим, ніж для оригінальних даних, особливо для складніших моделей, таких як RandomForest.
4. Ефективне використання пам'яті: Розмір редукованих моделей залишається надзвичайно малим (56 байт в оперативній пам'яті), що робить їх легкими для розгортання та використання в обмежених середовищах.

Загалом, результати дослідження свідчать про успішну реалізацію конвеєру для редукції зображень, який може бути ефективно використаний у різних практичних додатках. Використані методи та підходи продемонстрували свою доцільність та ефективність, що дозволяє рекомендувати їх для подальшого впровадження у реальних проектах.

ПЕРЕЛІК ДЖЕРЕЛ

1. Goodfellow I., Bengio Y., Courville A. Deep Learning. Cambridge: MIT Press, 2016. 775 с.
2. Chollet F. Deep Learning with Python. Shelter Island: Manning Publications Co., 2017. 361 с.
3. Hinton G. E., Salakhutdinov R. R. Reducing the dimensionality of data with neural networks. Science. 2006. Том 313, №. 5786. 504–507 с.
4. Lee, H., & Choi, S. (2018). Encoding convolutional neural networks into recurrent neural networks via autoencoder. IEEE Access, 6, 30097-30108.
5. Golub, G. H., & Reinsch, C. (1971). Singular value decomposition and least squares solutions. In Linear Algebra (с. 134-151). Springer, Berlin, Heidelberg.
6. Bourlard H., Kamp Y. Auto-association by multilayer perceptrons and singular value decomposition. Biological cybernetics. 1988. Том 59, №.4-5. 291–294 с.
7. Ke, Q., & Kanade, T. (2005, June). Robust subspace computation using l1 norm. In IEEE Conference on Computer Vision and Pattern Recognition Workshops (с. 175-175).
8. Документація TensorFlow. [Електронний ресурс] - Режим доступу до ресурсу: https://www.tensorflow.org/learn (дата звернення: 10.04.2024).
9. Документація Keras. [Електронний ресурс] - Режим доступу до ресурсу: https://keras.io/guides/ (дата звернення: 10.05.2024).
10. Документація scikit-learn. [Електронний ресурс] - Режим доступу до ресурсу: https://scikit-learn.org/stable/ documentation.html (дата звернення: 10.05.2024).

ДОДАТКИ ДО КУРСОВОЇ РОБОТИ

ДОДАТОК А

Лістинги коду

EvaluationFunctions.py

```
import os, sys, time, joblib
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import classification_report, confusion_matrix

def print_model_size(model_to_save):
    file_path =  'my_model.joblib'
    joblib.dump(model_to_save, file_path)
    file_size_bytes = os.path.getsize(file_path)
    file_size_kb = file_size_bytes / 1024
    print(f"Розмір файлу-моделі на диску: {file_size_kb:.2f} KB")
    if os.path.isfile(file_path):
        os.remove(file_path)

def my_score(model, x_test, y_test):
    start_time = time.time()
    y_pred = model.predict(x_test)
    print(f"Час затрачений для передбачення всіх x_test({len(x_test)}) -
{round(time.time() - start_time, 4)} сек")
    size_in_bytes = sys.getsizeof(model)
    print(f"Розмір моделі в оперативній пам'яті: {size_in_bytes} байт")
    print_model_size(model)
    print("classification report:")
    print(classification_report(y_test, y_pred))
    cm = confusion_matrix(y_test, y_pred)
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
    plt.title('Матриця помилок')
    plt.xlabel('Передбачений клас')
    plt.ylabel('Справжній клас')
    plt.show()

def performance_evaluation(model, x_train, y_train, x_test, y_test):
    start_time = time.time()
    model.fit(x_train, y_train)
    print(f"Час навчання моделі - {round(time.time() - start_time, 2)}")
    my_score(model, x_test, y_test)
```

pipline.py

```
import time
import tensorflow as tf
from sklearn.decomposition import TruncatedSVD
from tensorflow.keras.layers import Input, Dense, Flatten, Reshape, Conv2D,
MaxPooling2D, UpSampling2D

class myReducePipline():
    lat_dim_ae = 32
    lat_dim_svd = 70

    def __init__(self) -> None:
        self.create_ae()
        self.create_svd()
```

```
def create_ae(self):
    # Input shape (assuming MNIST data)
    input_shape = (28, 28, 1)
    inputs = Input(shape=input_shape)
    x = Conv2D(32, (3, 3), activation='relu', padding='same')(
        inputs)  # Output: 28x28 (avoid information loss)
    x = MaxPooling2D((2, 2), padding='valid')(x)  # Output: 14x14
    x = Conv2D(64, (3, 3), activation='relu', padding='same')(
        x)  # Output: 14x14 (avoid information loss)
    x = MaxPooling2D((2, 2), padding='valid')(x)  # Output: 7x7
    x = Flatten()(x)
    encoded = Dense(self.lat_dim_ae)(x)
    # Decoder (adjusted based on encoder output)
    # Match the number of channels in the next Conv2D layer
    x = Dense(7 * 7 * 64)(encoded)
    x = Reshape((7, 7, 64))(x)
    x = Conv2D(64, (3, 3), activation='relu',
               padding='same')(x)  # Output: 7x7
    x = UpSampling2D((2, 2))(x)  # Output: 14x14
    x = Conv2D(32, (3, 3), activation='relu',
               padding='same')(x)  # Output: 14x14
    # Output: 28x28 (matches input height and width)
    x = UpSampling2D((2, 2))(x)
    decoded = Conv2D(1, (3, 3), activation='sigmoid',
                     padding='same')(x)  # Output: 28x28x1
    self.autoencoder = tf.keras.Model(inputs, decoded)
    self.autoencoder.compile(optimizer='adam', loss='mse')

def create_svd(self):
    self.svd = TruncatedSVD(n_components=self.lat_dim_svd)

def fit_ae_model(self, x_train, x_test):
    '''
    x_train and x_test shoud be original dimmension:
    (n, 28, 28) \n
    Return dimm reshaped to 784 (28*28)
    '''
    x_train = x_train.reshape((-1, 28, 28, 1))
    x_test = x_test.reshape((-1, 28, 28, 1))
    # -1 for batch size (infer from data)
    self.autoencoder.fit(x_train, x_train, epochs=10)
    x_train_ae = self.autoencoder.predict(x_train)
    x_test_ae = self.autoencoder.predict(x_test)
    x_train_ae = x_train_ae.reshape((len(x_train_ae), -1))
    x_test_ae = x_test_ae.reshape((len(x_test_ae), -1))
    return x_train_ae, x_test_ae

def fit_svd_model(self, x_train, x_test):
    self.svd.fit(x_train)
    x_train_svd = self.svd.transform(x_train)
    x_test_svd = self.svd.transform(x_test)
    return x_train_svd, x_test_svd

def get_svd_inverse_transform(self, x):
    return self.svd.inverse_transform(x)

def run_pipeline(self, x_train, x_test):
    start_time = time.time()
    x_train, x_test = self.fit_ae_model(x_train, x_test)
    AE_train_time = time.time() - start_time
    start_time = time.time()
    x_train, x_test = self.fit_svd_model(x_train, x_test)
```

```
        svd_train_time = time.time() - start_time
        print("AE create + train time: ", AE_train_time, " sec")
        print("SVD create + train time:", svd_train_time, " sec")
        return x_train, x_test
```

main.py

```
import sys, math
import numpy as np
import tensorflow as tf
import matplotlib.pyplot as plt
from sklearn.neural_network import MLPClassifier
from EvaluationFunctions import performance_evaluation
import pipline

(x_train, y_train), (x_test, y_test) = tf.keras.datasets.mnist.load_data()
origin_dim = x_train.shape[1]
x_train = x_train.astype('float32') / 255.0
x_test = x_test.astype('float32') / 255.0
x_train_flat = x_train.reshape((len(x_train), -1))
x_test_flat = x_test.reshape((len(x_test), -1))

def get_model():
    return MLPClassifier(solver="lbfgs")

performance_evaluation(get_model(), x_train_flat, y_train, x_test_flat, y_test)

myPipe = pipline.myReducePipline()
x_train_ae,x_test_ae = myPipe.fit_ae_model(x_train,x_test)
performance_evaluation(get_model(), x_train_ae, y_train, x_test_ae, y_test)

myPipeSvd = pipline.myReducePipline()
x_train_svd,x_test_svd = myPipeSvd.fit_svd_model(x_train_flat,x_test_flat)
performance_evaluation(get_model(), x_train_svd, y_train, x_test_svd, y_test)

myPipe = pipline.myReducePipline()
x_train_reduced,x_test_reduced = myPipe.run_pipeline(x_train,x_test)
performance_evaluation(get_model(), x_train_reduced, y_train, x_test_reduced,
y_test)

x1 = sys.getsizeof(x_train)
x2 = sys.getsizeof(x_train_ae)
x3 = sys.getsizeof(x_train_reduced)
print(f"Розмір оригінальних даних       - { x1 } байт")
print(f"Розмір після автоенкодера       - { x2 } байт ({x2/x1})")
print(f"Розмір після автоенкодера + SVD - { x3 } байт ({x3/x1})")

x_ae = x_train_ae.reshape(len(x_train_ae), origin_dim, -1)
X_recovered = myPipeSvd.get_svd_inverse_transform(x_train_svd)
x_svd = X_recovered.reshape(len(X_recovered), origin_dim, -1)
x_svd_lat = x_train_svd.reshape(len(x_train_svd), 10, -1)

def plot_classes(x, y, title=''):
    unique_classes = np.unique(y)
    nrows, ncols = 1, 10
    fig, axes = plt.subplots(nrows, ncols, figsize=(12, 5), squeeze=False)

    for i, digit_class in enumerate(unique_classes):
        row, col = i // ncols, i % ncols
        ax = axes[row][col]
        class_indices = np.where(y == digit_class)[0]
        digit_index = class_indices[0]
        digit_data = x[digit_index]
```

```
        ax.imshow(digit_data, cmap='gray')
        ax.set_title(f"Клас: {digit_class}")
        ax.axis('off')
    plt.tight_layout()
    print(title)
    plt.show()

plot_classes(x_train,y_train, "origin")
plot_classes(x_ae,y_train, "after AutoEncoder")
plot_classes(x_svd,y_train, "after AE+SVD (inversed)")
plot_classes(x_svd_lat,y_train, "after AE+SVD (reduced)")
```

find_oprimal_param.ipynb.py

```
import pandas as pd
import tensorflow as tf
import matplotlib.pyplot as plt
from keras_tuner import RandomSearch
from sklearn.pipeline import Pipeline
from sklearn.decomposition import TruncatedSVD
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import GridSearchCV
from tensorflow.keras.layers import Input, Dense, Flatten, Reshape, Conv2D,
MaxPooling2D, UpSampling2D

(x_train, y_train), (x_test, y_test) = tf.keras.datasets.mnist.load_data()
origin_dim = x_train.shape[1]
x_train = x_train.astype('float32') / 255.0
x_test = x_test.astype('float32') / 255.0
x_train_flat = x_train.reshape((len(x_train), -1))
x_test_flat = x_test.reshape((len(x_test), -1))
x_train_flat.shape

def create_autoencoder(hp):
    lat_dim_ae = hp.Int('lat_dim', min_value=10, max_value=100, step=5)
    input_shape = (28, 28, 1)
    inputs = Input(shape=input_shape)
    x = Conv2D(32, (3, 3), activation='relu', padding='same')(inputs)
    x = MaxPooling2D((2, 2), padding='valid')(x)
    x = Conv2D(64, (3, 3), activation='relu', padding='same')(x)
    x = MaxPooling2D((2, 2), padding='valid')(x)
    x = Flatten()(x)
    encoded = Dense(lat_dim_ae)(x)
    x = Dense(7 * 7 * 64)(encoded)
    x = Reshape((7, 7, 64))(x)
    x = Conv2D(64, (3, 3), activation='relu',padding='same')(x)
    x = UpSampling2D((2, 2))(x)
    x = Conv2D(32, (3, 3), activation='relu',padding='same')(x)
    x = UpSampling2D((2, 2))(x)
    decoded = Conv2D(1, (3, 3), activation='sigmoid',padding='same')(x)
    autoencoder = tf.keras.Model(inputs, decoded)
    autoencoder.compile(optimizer='adam', loss='mse')
    return autoencoder

steps = [
    ('SVD', TruncatedSVD()),
    ('MLP', MLPClassifier(solver="lbfgs"))]
pipeline = Pipeline(steps)

param_grid = {
    'SVD__n_components': list(range(10, 160, 10)),
}
```

```
grid_search = GridSearchCV(pipeline, param_grid=param_grid,cv=2,
scoring='accuracy', return_train_score=True, verbose=2)
grid_search.fit(x_train_flat, y_train)
results = pd.DataFrame(grid_search.cv_results_)
print(results)

results2 = results[['param_SVD__n_components', 'mean_test_score',
                    'rank_test_score', 'mean_fit_time', 'mean_score_time']]
print(results2)

tuner = RandomSearch(
    create_autoencoder,
    objective='val_loss',
    executions_per_trial=1,
    directory='my_dir',
    project_name='autoencoders')

tuner.search(x_train, x_train, epochs=12, validation_data=(x_test, x_test))

best_model = tuner.get_best_models(num_models=1)[0]
print(best_model)
print(tuner.results_summary())

results = pd.DataFrame(tuner.oracle.get_best_trials(
    num_trials=len(tuner.oracle.trials)))
print(results)

plt.figure(figsize=(10, 6))
plt.plot(results[['param_SVD__n_components']],
         results[['mean_test_score']], marker='o', color='blue')
plt.xlabel('Значення n_components')
plt.ylabel('Accuracy')
plt.title('Точність MLP відносно значень n_components для SVD')
plt.xticks(results['param_SVD__n_components'].tolist())
plt.grid(True)
plt.legend()
plt.show()

plt.figure(figsize=(10, 6))
plt.plot(results[['param_SVD__n_components']],
         results[['rank_test_score']], marker='o', color='blue')
plt.xlabel('Значення n_components')
plt.ylabel('Accuracy - Ranked (lower - better)')
plt.title('Точність(Ranked) моделі MLP відносно значень n_components для SVD')
plt.xticks(results['param_SVD__n_components'].tolist())
plt.grid(True)
plt.legend()
plt.show()

plt.figure(figsize=(10, 6))
plt.plot(results[['param_SVD__n_components']],
         results[['mean_fit_time']], marker='o', color='blue')
plt.xlabel('Значення n_components')
plt.ylabel('mean_fit_time')
plt.title('MLP fit_time відносно значень n_components для SVD')
plt.xticks(results['param_SVD__n_components'].tolist())
plt.grid(True)
plt.legend()
plt.show()

plt.figure(figsize=(10, 6))
plt.plot(results[['param_SVD__n_components']],
         results[['mean_score_time']], marker='o', color='blue')
```

```
plt.xlabel('Значення n_components')
plt.ylabel('mean_score_time')
plt.title('Час затрачений MLP для передбачення даних з тестової вибірки відносно
значень n_components для SVD')
plt.xticks(results['param_SVD__n_components'].tolist())
plt.grid(True)
plt.legend()
plt.show()

lat_vec = [12, 20, 30, 50, 70, 100]
ae_training_time = [34, 34, 33, 34, 34.7, 37]
mlp_training_time = [37, 37, 36.88, 36, 37, 37.2]
prediction_time = [0.044, 0.038, 0.037, 0.038, 0.036, 0.039]
accuracy = [96, 97, 97, 97, 97, 97]

plt.figure(figsize=(10, 6))
plt.plot(lat_vec, ae_training_time, marker='o',
         color='blue', label='Час навчання AE')
plt.plot(lat_vec, mlp_training_time, marker='o',
         color='red', label='Час навчання MLP')
plt.xlabel('Значення ae_lat_vec')
plt.ylabel('Час навчання (сек)')
plt.title('Час навчання моделі(MLP) відносно значень ae_lat_vec')
plt.xticks(lat_vec)
plt.grid(True)
plt.legend()
plt.show()

plt.figure(figsize=(10, 6))
plt.plot(lat_vec, prediction_time, marker='o',
         color='green', label='Час передбачення')
plt.xlabel('Значення ae_lat_vec')
plt.ylabel('Час передбачення')
plt.title('Час передбачення моделі(MLP) відносно значень ae_lat_vec')
plt.xticks(lat_vec)
plt.grid(True)
plt.legend()
plt.show()

plt.figure(figsize=(10, 6))
plt.plot(lat_vec, accuracy, marker='o', color='purple', label='Точність')
plt.xlabel('Значення ae_lat_vec')
plt.ylabel('Точність')
plt.title('Точність моделі(MLP) відносно значень ae_lat_vec')
plt.xticks(lat_vec)
plt.grid(True)
plt.legend()
plt.show()
```