Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп’ютерної інженерії та управління

Кафедра Комп’ютерних інтелектуальних технологій та систем

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

рівень вищої освіти другий (магістерський)

Нейромережева кластеризація даних на основі нейронних мереж прямого поширення
(тема)

Виконав:

студент 2 курсу, групи КІТм-21-1

Глюза А.П.
(прізвище, ініціали)

Спеціальність 123 Комп’ютерна інженерія

Тип програми освітньо-професійна

Освітня програма Комп’ютерні інтелектуальні технології

Керівник проф. О.Г. Руденко
(посада, ініціали, прізвище)

Допускається до захисту _____
(підпис)

Зав.кафедри _____ О.Г. Руденко
(підпис)

2022 р.

Харківський національний університет радіоелектроніки

Факультет	Комп'ютерної інженерії та управління
Кафедра	Комп'ютерних інтелектуальних технологій та систем
Рівень вищої освіти	другий (магістерський)
Спеціальність	123 Комп'ютерна інженерія
Тип програми	освітньо-професійна
Освітня програма	Комп'ютерні інтелектуальні технології

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)
«___16___» листопада 2022 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Глюзи А. П.
(прізвище, ініціали)

1. Тема роботи (проекту) «Нейромережева кластеризація даних на основі нейронних мереж прямого поширення»
затверджена наказом університету від «16» листопада 2022р. № _____
2. Термін подання студентом роботи до екзаменаційної комісії ___ _____2022р.
3. Вихідні дані до роботи (проекту) _____
4. Перелік питань, що потрібно опрацювати в роботі 1. Основні задачі нейромережевої кластеризації. 2. Особливості застосування нейронних мереж для кластеризації даних. 3. Реалізація нейронної мережі

5. Перелік графічного матеріалу із зазначенням кресленików, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням кафедри)
Слайд-презентація, 12 слайдів, Додаток А

6. Консультанти розділів роботи (п.6 включається до завдання за наявністю консультантів згідно до наказу, зазначеному у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Огляд літератури за темою роботи		вик
2	Розгляд основних задач нейромережевої		
	кластеризації даних		
3	Розгляд особливостей застосування		
	нейронних мереж для кластеризації даних		
4	Реалізація нейронної мережі		
5	Оформлення матеріалів кваліфікаційної		
	роботи		
6	Подання кваліфікаційної роботи керівникові		
7	Подання кваліфікаційної роботи на рецензу-		
	вання		

Дата видачі завдання 7 листопада 2022р.

Студент _____
(підпис)

Керівник роботи _____ проф. О.Г.Руденко
(підпис) (посада, ініціали,прізвище)

РЕФЕРАТ

Магістерська кваліфікаційна робота містить 76 сторінок, на яких розміщено 26 рисунків та 3 таблиці. Перелік використаної літератури містить 21 найменування.

ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ, НЕЙРОННА МЕРЕЖА, МЕНЕДЖЕР, ШТУЧНИЙ ІНТЕЛЕКТ, JAVA, SPRING.

Тема роботи: Нейромережева кластеризація даних на основі нейронних мереж прямого поширення.

Мета роботи: дослідження методів нейромережевої кластеризації даних на основі нейронних мереж прямого поширення.

Предмет дослідження – нейронна мережа для кластеризації даних.

Об’єкт дослідження – процес кластеризації даних із застосуванням нейронних мереж.

Методи дослідження: системний аналіз, порівняння, експеримент.

В роботі розглянуто можливість кластеризації даних на основі нейронних мереж прямого поширення. Розглянуто основні задачі нейромережевої кластеризації. Визначено основні типи нейронних мереж та алгоритми їх навчання. Наведено результати експериментальних досліджень кластеризації даних за допомогою нейронної мережі прямого поширення. Доведено ефективність застосування нейронних мереж прямого поширення для задач кластеризації.

Новизна роботи полягає у вдосконаленні алгоритмів кластеризації даних за допомогою нейронних мереж.

Практична значимість отриманих результатів полягає у можливості використання розробленої нейронної мережі для кластеризації.

ABSTRACT

Master's qualification work contains 76 pages, which contains 26 figures and 3 tables. The list of references contains 21 titles.

INTELLIGENT DECISION SUPPORT SYSTEMS, NEURAL NETWORK, MANAGER, ARTIFICIAL INTELLIGENCE, JAVA, SPRING.

Theme of the work: Neural network clustering of data based on feed forward neural networks.

Purpose of work: research of methods of neural network data clustering based on feed forward neural networks.

Subject of research – neural network for data clustering.

Object of study – the process of data clustering using neural networks.

Research methods: analysis, comparison, experiment.

The paper considers the possibility of data clustering based on feed forward neural networks. The main tasks of neural network clustering are considered. The main types of neural networks and algorithms for their training are defined.

Experimental studies of data clustering using feed forward neural network are shown. The effectiveness of the use of feedforward neural networks for clustering problems is proved.

The novelty of the work lies in the improvement of data clustering algorithms using neural networks.

The practical significance of the obtained results lies in the possibility of using the developed neural network for clustering.

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління
Кафедра Комп'ютерних інтелектуальних технологій та систем

АНОТАЦІЯ
КВАЛІФІКАЦІЙНОЇ РОБОТИ

рівень вищої освіти другий (магістерський)

Нейромережева кластеризація даних на основі нейронних мереж прямого поширення
(тема)

Виконав:
студент 2 курсу, групи КІТм-21-1
Глюза А.П.
(прізвище, ініціали)

Спеціальність 123 Комп'ютерна інженерія
Тип програми освітньо-професійна
Освітня програма Комп'ютерні інтелектуальні технології
Керівник проф. О.Г.Руденко
(посада, ініціали, прізвище)

2022 р.

АНОТАЦІЯ

Глюза А.П. Тема магістерської кваліфікаційної роботи – Нейромережева кластеризація даних на основі нейронних мереж прямого поширення. Магістерська кваліфікаційна робота.

У магістерській кваліфікаційній роботі вирішено актуально проблему кластеризації даних з застосування нейронних мереж прямого поширення.

Метою кваліфікаційної роботи є дослідження методів нейромережевої кластеризації даних на основі нейронних мереж прямого поширення.

Об’єктом дослідження цієї роботи є процес кластеризації даних із застосуванням нейронних мереж.

Предмет дослідження – нейронна мережа для кластеризації даних.

Кластеризація займає центральне місце у багатьох дослідженнях у галузі біоінформатики, заснованих на даних, і є потужним обчислювальним методом. Зокрема, кластеризація допомагає аналізувати неструктуровані та багатовимірні дані у вигляді послідовностей, виразів, текстів та зображень. Актуальним є застосування нейронних мереж для кластеризації даних.

Результатом написання **першого розділу** магістерської кваліфікаційної роботи є огляд літератури з теми дослідження та розгляд основних задач нейромережевої кластеризації даних.

Встановлено, що кластеризація – це фундаментальне завдання, що зазвичай застосовується в дослідницькому аналізі даних, аналізі зображень, пошуку інформації, стисненні даних, розпізнаванні образів, кластеризації тексту та, наприклад, біоінформатики.

Кластеризація використовується для пошуку структури в нерозмічених даних. Це найпоширеніша форма навчання без учителя.

Також встановлено, що кластерний аналіз даних важливий при роботі з даними, тому що він дозволяє користувачам впорядковувати дані відповідно до критеріїв вирішуваних завдань, а використання методів кластеризації в

інтелектуальном аналізі даних систематизує та угруповує дані згідно ознак. За допомогою кластеризації можна досліджувати взаємозв'язок як між внутрішніми факторами (наприклад, ціноутворення, позиціонування продукту, навички персоналу), так і зовнішніми факторами, такими як конкуренція та демографічні характеристики. Наприклад, використання одного з методів кластеризації під час кластерного аналізу даних може допомогти бізнесу ідентифікувати окремі групи у своїй клієнтській базі.

Встановлено, що кластеризація великих наборів даних є найбільш цінним застосуванням цього інструменту аналізу завдяки обсягу роботи, який він вимагає. Як і у випадку з іншими інструментами навчання без вчителя, кластеризація може брати великі набори даних і без інструкцій швидко організовувати їх у щось корисніше.

Одне з найбільш цінних застосувань кластеризації полягає в тому, що через чутливість багатьох алгоритмів до точок викидів даних вони можуть бути ідентифікаторами аномалій даних. Дійсно, алгоритми кластерного аналізу, такі як просторова кластеризація додатків із шумом на основі щільності призначені для пошуку окремих кластерів, які розташовані близько один до одного, і позначають викиди в наборах даних. Розуміння аномальних даних може допомогти оптимізувати існуючі інструменти збору даних та призвести до більш точних результатів у довгостроковій перспективі.

Отже, застосування кластерного аналізу надає багато переваг.

Особливості кластеризації:

1. Масштабованість кластеризації.

2. Висока розмірність.

3. Зручність використання алгоритму з кількома типами даних: різні типи даних можна використовувати з алгоритмами кластеризації.

4. Робота з неструктурованими даними.

5. Інтерпретованість.

Другий розділ роботи присвячений особливостям застосування нейронних мереж для кластеризації даних. Розглянуто структуру і архітектуру нейронних мереж. Встановлено, що нейронні мережі є функціональною одиницею глибокого навчання і, як відомо, імітують поведінку людського мозку на вирішення складних завдань, керованих даними.

Вхідні дані обробляються за допомогою різних шарів штучних нейронів, складених разом, для отримання бажаного результату.

Штучний нейрон – математична модель біологічного нейрона. А нейромережа – це зв'язка нейронів, кожен нейрон отримує та обробляє отриману інформацію, потім передає її наступному нейрону. Проте, всі нейрони обробляють інформацію однаково, тому найважливішу роль грають синапси, яка з'єднують нейрони. Саме синапси відповідають за правильність результату роботи нейромережі. Вони послаблюють або посилюють сигнал, що проходить між нейронами.

Головними компонентами нейронної мережі є:

– її архітектура – скільки у мережі прихованих шарів, вхідних та вихідних параметрів і як вони пов'язані, а також які активаційні функції застосовуються. Ці питання вирішує розробник;

– навчені ваги – спочатку ваги задаються випадковим чином та підбираються у процесі навчання. Зберігаються вони як сукупності матриць дійсних чисел. Таким чином, нейронну мережу можна зберегти шляхом збереження ваги та архітектури.

Далі встановлено, що основними типами навчання нейронних мереж є:

Спершу треба розглянути основні типи навчання нейронних мереж.

1. З учителем. Навчання з учителем – це одне з найпопулярніших і найчастіших завдань. У цьому випадку маємо вибірку і знаємо по ній правильні відповіді. Навчання з учителем краще підходить для завдань, у яких є досить великий набір достовірних даних для навчання алгоритму. Але

так буває далеко не завжди. Недолік даних – проблема, що найбільш часто зустрічається.

2. Навчання без учителя. Найчастіше ідеально розмічені та чисті дані дістати нелегко. Тому перед алгоритмом стоїть завдання знайти наперед не відомі відповіді. У таких ситуаціях використовують навчання без учителя. У навчанні без вчителя модель має набір даних без явних вказівок, що з нею робити. Нейронна мережа намагається самостійно знайти кореляції даних, намагаючись витягти корисні ознаки та аналізуючи їх.

3. Навчання із підкріпленням. Навчання з підкріпленням – один із способів машинного навчання, у ході якого випробувана система (агент) навчається, взаємодіючи з деяким середовищем. Коли агент робить дії, що сприяють досягненню мети, він отримує нагороду. Глобальна мета – передбачати такі кроки, щоб заробити максимальну нагороду зрештою.

Основними алгоритмами навчання нейронної мережі є:

1. Метод зворотного поширення. Цей метод також називають Backpropagation. Він є одним із основних способів навчання та містить у своїй основі алгоритм обчислення градієнтного спуску.

2. Метод Rprop. Він був запропонований як альтернатива попередньому способу навчання, який потребує занадто багато часу і стає незручним, якщо результати потрібно отримати в короткий термін. Для збільшення швидкості операцій було розроблено багато допоміжних алгоритмів.

3. Генетичний алгоритм навчання. За своїм принципом він схожий з еволюційними процесами природи, що ґрунтуються на комбінуванні (схрещуванні) результатів.

Визначено, що для навчання нейронної мережі ефективно застосувати алгоритм зворотного розповсюдження помилки. Метод *k*–середніх буде використовуватися на початковому етапі для формування навчальної вибірки.

В **третьому розділі** роботи представлена реалізація нейронної мережі прямого поширення для кластеризації даних.

Визначено, що для завдання кластеризації даних доцільно застосовувати середовище Scilab. За допомогою програми можна зображувати процес навчання у вигляді графіка, а отже, можна визначити, як проходив процес навчання та скільки циклів ітерацій навчання знадобилося, щоб помилка навчальної множини досягла потрібного рівня.

Важливою перевагою використання програми Scilab є можливість переглядати та коригувати ваги штучної нейронної мережі. Більшість програм нейромережевого моделювання позбавлені цієї важливої функції.

В експериментальній частині було реалізовано паралельну версію алгоритму *k*–середніх, проведено дослідження показників ефективності паралельного алгоритму на кластері Jet.

Проведені тести наочно демонструють, що значних прискорень можна досягти за рахунок використання паралельної реалізації алгоритму k–середніх для гіперспектральних зображень. Для великого файлу (600 МБ) найбільше прискорення досягнуто під час запуску паралельної програми на 14 обчислювальних вузлах кластера Jet. Для ближчого до типового розміру (140 МБ) найбільше прискорення отримано під час запуску паралельної програми на 9 обчислювальних вузлах кластера Jet. Експериментально встановлена ефективність застосовування даного підходу для нейромережевої кластеризації.

ДАНІ, КЛАСТЕРИЗАЦІЯ, АРХІТЕКТУРА, ЕФЕКТИВНІСТЬ, МЕТОД K–СЕРЕДНІХ, МЕРЕЖА ПРЯМОГО ПОШИРЕННЯ, АЛГОРИТМИ НАВЧАННЯ.

Публікації здобувача за темою роботи:

1. Глюза А. П. Класифікація нейронних мереж / Анастасія Павлівна Глюза. // Вінниця: *ГО "Європейська наукова платформа"*. – 2022. – С. 33–35.

Використані в роботі публікації керівника та співробітників кафедри:

1. Руденко О. Г. Искусственные нейронные сети: архитектуры, обучение, применения / О. Г. Руденко, Е. В. Бодянский. – Харьков: ТЕЛЕТЕХ, 2004. – 370 с.

ЗМІСТ

ВСТУП

Кластеризація займає центральне місце у багатьох дослідженнях у галузі біоінформатики, заснованих на даних, і є потужним обчислювальним методом. Зокрема, кластеризація допомагає аналізувати неструктуровані та багатовимірні дані у вигляді послідовностей, виразів, текстів та зображень.

Оскільки якість кластеризації залежить не тільки від розподілу точок даних, але і від представлення даних, нейронні мережі можуть бути ефективним засобом перетворення відображень з багатовимірного простору даних на низькорозмірний простір ознак, що призводить до покращення результатів кластеризації.

Метою написання магістерської дисертації є дослідження методів нейромережевої кластеризації даних на основі нейронних мереж прямого поширення.

При написанні роботи були поставлені наступні задачі:

1. Виконати огляд літератури з теми дослідження. Розглянути задачі нейромережевої кластеризації даних та існуючі підходи.

2. Дослідити особливості застосування нейронних мереж для кластеризації даних.

3. Навести приклад нейромережевої кластеризації даних на основі нейронних мереж прямого поширення.

Об’єктом дослідження магістерської дисертації є процес кластеризації даних із застосуванням нейронних мереж.

Предметом дослідження є нейронна мережа для кластеризації даних.

При написанні роботи були застосовані методи аналізу, порівняння, експерименту.

Практична значимість отриманих результатів полягає у можливості використання розробленої нейронної мережі для кластеризації.

1 ЗАДАЧІ НЕЙРОМЕРЕЖЕВОЇ КЛАСТЕРИЗАЦІЇ ДАНИХ ТА ІСНУЮЧІ ПІДХОДИ

1.1 Методи представлення даних

У широкому сенсі дані є фактами, текстом, графіками, картинками, звуками, аналоговими або цифровими відео–сегментами. Дані можуть бути отримані в результаті вимірів, експериментів, арифметичних та логічних операцій [1].

Сьогодні дані оброблюються скрізь у кожній області. Важливо правильно обробляти та зберігати їх без помилок. При роботі з даними важливо знати типи даних для їх обробки та отримання правильних результатів.

В загальному випадку виділяється два типи даних: якісні та кількісні дані, які далі поділяються на чотири типи даних: номінальні, порядкові, дискретні та безперервні.

Основні типи даних наведені на рисунку 1.1.

Дані мають бути подані у формі, придатній для зберігання, передачі та обробки. Іншими словами, дані – це необроблений матеріал, який надається постачальниками даних та використовується споживачами для формування інформації на основі даних [1].

Представлення даних – це метод аналізу даних. Відношення між фактами, ідеями, інформацією та концепціями зображуються у вигляді представлення даних [2].

Розглянемо деякі методи представлення даних.

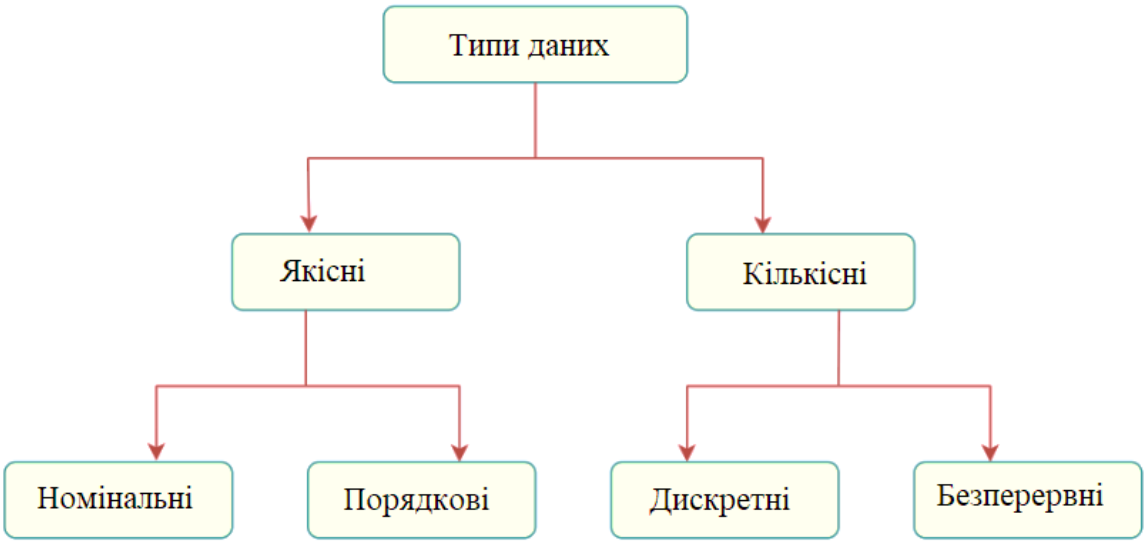

Рисунок 1.1 – Основні типи даних

Наприклад, методи представлення даних при передачі даних:

– десяткові числа;

– двійкові числа;

– шістнадцяткові числа;

– текст;

– графіка.

Будь–яка зібрана інформація може бути організована у вигляді таблиці частотного розподілу, потім показана за допомогою піктограм або гістограм.

Гістограма – це уявлення чисел, що складається зі смуг однакової ширини, довжина яких визначається обраною частотою та масштабом.

Подання даних визначається як методи, що використовуються для представлення інформації [2].

Приклади методів представлення даних у графічній формі:

- гістограма;
- таблиця розподілу частот;
- кругова діаграма;
- лінійний графік.

Гістограма візуально представляє якісні дані.

Інформація відображається горизонтально або вертикально та порівнює такі елементи, як кількість, характеристики, час та частота.

Таблиця частот або розподіл частот – це метод представлення необроблених даних, в якому можна легко зрозуміти інформацію, що міститься в необроблених даних.

Таблиця частотного розподілу будується з допомогою контрольних позначок. Рахункові мітки є формою числової системи з вертикальними лініями, використовуваними для підрахунку [3].

Кругова діаграма використовується для представлення пропорцій числових набору даних. Цей графік включає розподіл кола на різні сектори, де кожен з секторів представляє частку певного елемента в цілому.

Графік, який використовує точки та лінії для подання змін з плином часу, визначається як лінійний графік.

Іншими словами, це діаграма, яка показує лінію, що з'єднує кілька точок, або лінію, що показує зв'язок між точками.

Діаграма ілюструє кількісні дані між двома змінними за допомогою прямої лінії або кривої, яка з'єднує ряд послідовних точок даних. Лінійні діаграми порівнюють дві змінні по вертикальній та горизонтальній осях.

Очевидно, що даний список не є вичерпним та в ньому виділено основні методи представлення даних у графічному вигляді.

1.2 Кластерний аналіз та особливості його застосування

Кластеризація – це фундаментальне завдання, що зазвичай застосовується в дослідницькому аналізі даних, аналізі зображень, пошуку інформації, стисненні даних, розпізнаванні образів, кластеризації тексту та, наприклад, біоінформатики [4].

Кластеризація використовується для пошуку структури в нерозмічених даних. Це найпоширеніша форма навчання без учителя. Враховуючи набір даних, про який користувачу нічого не відомо, алгоритм кластеризації може

виявити групи об'єктів, у яких середні відстані між елементами кожного кластера менші, ніж між елементами в інших кластерах. Приклад кластеризації наведено на рисунку 1.2.

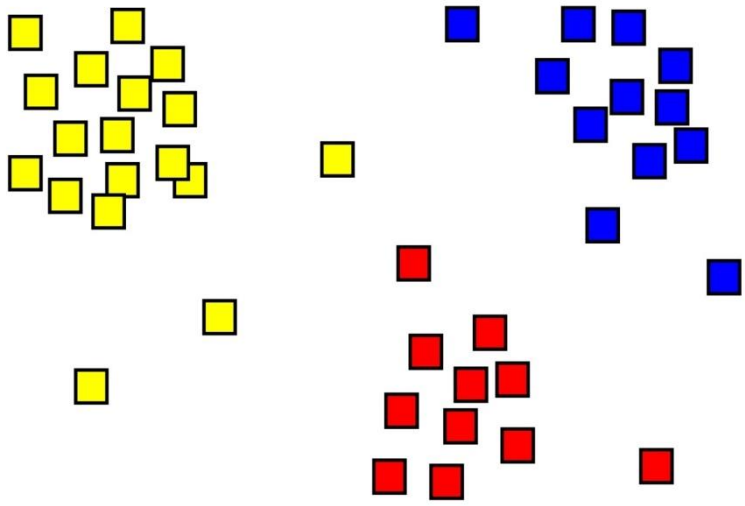

Рисунок 1.2 – Приклад кластеризації даних

Кластеризація має багато практичних застосувань [4].

Основною метою кластеризації є угруповання даних у кластери на основі подібності, щільності, інтервалів або конкретних показників статистичного розподілу простору даних [5–7].

Наприклад, у біоінформатиці кластеризація експресії генів (GE) може виявити групи функціонально пов'язаних генів, в яких гени з невеликою відстанню мають однакові патерни експресії і можуть бути схожими.

Такий аналіз визначає, які гени «вмикаються» або «вимикаються» за певних умов.

Іншим прикладом є кластеризація зображень шляхом вивчення прихованих патернів з немаркованого набору даних. Крім того, візуалізація, інтерпретація та аналіз багатомірних та великомасштабних даних у всій їх неструктурованій цілісності можуть викликати труднощі, якщо дані не організовані в кластери [8].

Оскільки кластерний аналіз сам по собі не є одним конкретним алгоритмом, можуть застосовуватися різні методи, які різняться з точки зору розуміння того, що є кластером і як їх ефективно знаходити. На практиці різні завдання вимагають різних заходів подібності та методів поділу. Крім того, пошук оптимального алгоритму кластеризації для конкретної задачі є складним завданням і може бути сформульований як багатокритеріальне завдання оптимізації.

Кластерний аналіз даних важливий при роботі з даними, тому що він дозволяє користувачам «заглиблюватися» в дані, а використання методів кластеризації для аналізу даних може допомогти їм отримати додаткову інформацію з даних, які вони мають.

Виходячи з цього вони можуть досліджувати взаємозв'язок як між внутрішніми факторами (наприклад, ціноутворення, позиціонування продукту, навички персоналу), так і зовнішніми факторами, такими як конкуренція та демографічні характеристики. Наприклад, використання одного з методів кластеризації під час кластерного аналізу даних може допомогти бізнесу ідентифікувати окремі групи у своїй клієнтській базі. Вони можуть поєднувати різні типи клієнтів в одну групу на основі різних факторів, наприклад моделей покупок. Фактори, проаналізовані за допомогою кластеризації, можуть вплинути на продаж і задоволеність клієнтів, що робить її безцінним інструментом для збільшення доходів, скорочення витрат, а іноді навіть того й іншого [9].

Тобто, кластеризація – це простий спосіб виконання багатьох поверхневих аналізів, які можуть дати швидкі результати в різних областях.

Кластеризація великих наборів даних, можливо, є найбільш цінним застосуванням цього інструменту аналізу завдяки обсягу роботи, який він вимагає. Як і у випадку з іншими інструментами навчання без вчителя, кластеризація може брати великі набори даних і без інструкцій швидко організовувати їх у щось корисніше [10].

Окрім цього, одне з найбільш цінних застосувань кластеризації полягає в тому, що через чутливість багатьох алгоритмів до точок викидів даних вони можуть бути ідентифікаторами аномалій даних. Дійсно, алгоритми кластерного аналізу, такі як просторова кластеризація додатків із шумом на основі щільності (DBSCAN), призначені для пошуку окремих кластерів, які розташовані близько один до одного, і позначають викиди в наборах даних. Розуміння аномальних даних може допомогти оптимізувати існуючі інструменти збору даних та призвести до більш точних результатів у довгостроковій перспективі.

Все вищенаведене показує те, що застосування кластерного аналізу надає багато переваг.

Особливості кластеризації:

1. Масштабованість кластеризації. В даний час існує величезна кількість даних, і доводиться мати справу з величезними базами даних. Для роботи з великими базами даних алгоритм кластеризації має бути масштабованим. Дані мають бути масштабованими, якщо вони не масштабуються, неможливо отримати відповідний результат, що призведе до неправильних результатів.

2. Висока розмірність. Алгоритм повинен вміти обробляти багатовимірний простір разом із даними невеликого розміру.

3. Зручність використання алгоритму з кількома типами даних: різні типи даних можна використовувати з алгоритмами кластеризації. Алгоритм кластеризації має бути здатний працювати з різними типами даних, такими як дискретні, категоріальні та інтервальні дані, двійкові дані тощо.

4. Робота з неструктурованими даними. Деякі бази даних містять пропущені значення, зашумлені або хибні дані. Якщо алгоритми є чутливими до таких даних, це може призвести до неякісних кластерів. Таким чином, алгоритм кластеризації повинен мати можливість обробляти неструктуровані дані та надавати деяку структуру даним, організуючи їх у групи схожих об'єктів даних.

5. Інтерпретованість. Результати кластеризації мають бути інтерпретованими, зрозумілими та придатними для використання. Інтерпретованість відбиває легкість розуміння даних.

Варто відмітити, що і кластеризація, і класифікація є методами ідентифікації шаблонів, які застосовуються у машинному навчанні, і використовуються для поділу об'єктів різні класи з урахуванням їх характеристик. Між цими двома є подібність, але основна відмінність у тому, що метод класифікації використовує зумовлені класи, яким присвоюються об'єкти, тоді як метод кластеризації групує об'єкти з урахуванням виявлення подібності з–поміж них. Класифікація використовується з позначеними даними та орієнтована на навчання з учителем, а кластеризація використовується з немаркованими даними та орієнтована на навчання без вчителя [11].

1.3 Огляд основних алгоритмів кластерного аналізу. Ієрархічна та ітеративна кластеризація

Алгоритми кластеризації зазвичай ділять на ієрархічні–ті, які видають ієрархію даних на виході для зручнішої інтерпретації даних, і неієрархічні–алгоритми, у яких вибір інтерпретації відбувається без вибору ієрархії.

Ієрархічні поділяються на агломеративні та дивізимні. Перші спочатку привласнюють кожен об'єкт у кластер, а потім об'єднують кластери. Дивізимні алгоритми спочатку визначають всі об'єкти в один кластер, а потім поділяють, доки кожен об'єкт не буде у своєму власному кластері. На виході ієрархічного алгоритму маємо дендрограму–схему послідовності злиття об'єктів у кластер або їх поділ.

Неієрархічні алгоритми в свою чергу поділяються на ітеративні (ітеративно розподіляють об'єкти за кластерами), щільнісні (визначають кластер як купність об'єктів), модельні (є структура кластера і

максимізуються подібності між структурою та даними), концептуальні та мережеві.

Підходи кластерного аналізу, такі як ієрархічна кластеризація (HC), кластеризація на основі центроїду (CC), кластеризація на основі розподілу (DC), кластеризація на основі щільності (DC1) та карт, що самоорганізуються (COM) пропонуються в літературі [9,10].

Інші підходи включають імовірнісну кластеризацію, кластеризацію на основі сітки, спектральну кластеризацію та невід'ємну матричну факторизацію [11].

Алгоритми HC (наприклад, агломеративні) включають створення кластерів, що мають заздалегідь певний порядок (зверху вниз або знизу вгору), де кластери нижчого рівня об'єднуються в ще більші кластери на більш високих рівнях, створюючи ієрархію кластерів.

В агломеративній кластеризації (AC) спочатку кожна точка даних вважається окремим кластером. Подібні кластери потім поєднуються з іншими кластерами до тих пір, поки на кожній ітерації не буде сформовано один або K кластерів.

Переваги алгоритмів HC полягають у їх простоті та візуальній привабливості, і залежно від бажаного ступеня деталізації можна вибрати «розрізати» ієрархію на бажаному рівні для отримання відповідної кластеризації.

Однак якість кластеризації (CQ) чутлива до шуму, що ускладнює інтерпретацію ієрархії. Крім того, точки даних групуються з локальними рішеннями, що базуються на детермінованих атрибутах, без можливості переглянути кластеризацію.

Навпаки, алгоритми CC (наприклад, K–means, розбиття навколо медоїдів (K–medoids)) пропонують кілька переваг порівняно з COM і HC, наприклад, часто вони забезпечують чудову продуктивність з точки зору точності щільності точок, збереження топології та обчислювальних вимог.

Однак алгоритми CC не здатні знаходити неопуклі кластери. Алгоритми постійного струму (наприклад, змішана модель Гауса (GMM)) засновані на моделях розподілу, де кластери визначаються як точки даних, що належать виявленому розподілу.

Загалом підходи DC створюють складні моделі для кластерів, тому можна зафіксувати кореляцію та залежності між атрибутами. Недоліком є те, що не можна розробити чітко визначену модель, якщо розподіл Гаусса заснований на сильному припущенні про дані.

Незважаючи на це, алгоритми працюють досить добре для середніх та малорозмірних даних, точність та ефективність для багатовимірних наборів даних з величезною кількістю вибірок різко погіршуються.

Детальніше розглянемо деякі прикладки кластеризації.

Методи ієрархічної кластеризації. Як раніше зазначалося, вони поділяються на алгомеративні та дивізимні.

Алгомеративний алгоритм полягає у наступних кроках:

1. Обчислення матриці близькості.

2. Застосування умови, що на початковому етапі кожна точка даних буде кластером.

3. Об'єднання двох найближчих кластери та оновлення матриці близькості.

4. Поки не залишиться лише один кластер, ключовою операцією є обчислення близькості двох кластерів.

Припустимо, що маємо шість точок даних {A, B, C, D, E, F}.

Крок 1: На початковому етапі необхідно обчислити близькість окремих точок та розглянути всі шість точок даних як окремі кластери, як показано на рисунку 1.3.

Крок 2: На другому етапі схожі кластери поєднуються і формуються як єдиний кластер. Припускається, що B, C та D, E — схожі кластери, об'єднані на другому етапі. Тепер залишилося чотири кластери: A, BC, DE, F.

Крок 3: Знову обчислюється близькість нових кластерів і поєднуються схожі кластери, щоб сформувати нові кластери A, BC, DEF.

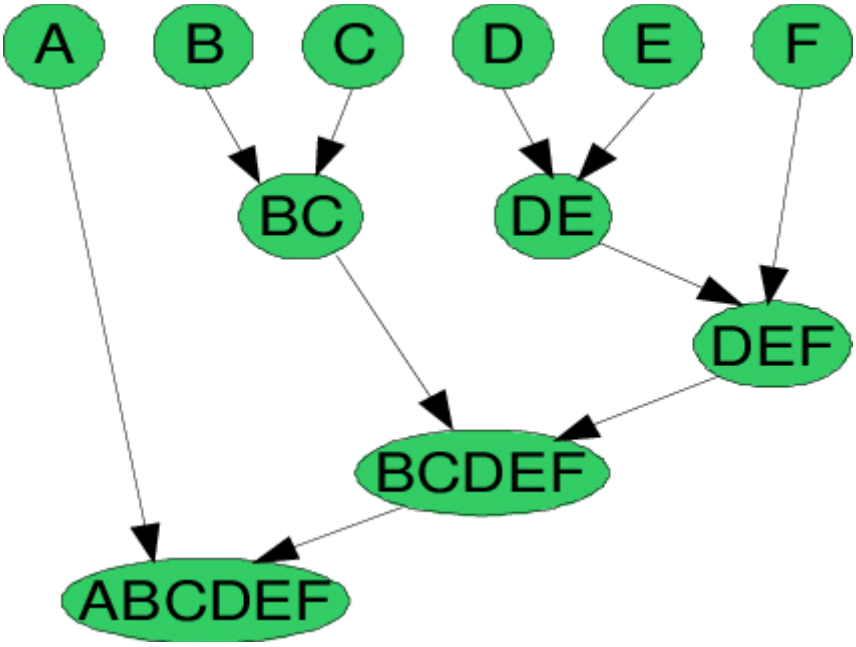

Рисунок 1.3 – До методу агломеративної класифікації даних

Крок 4: Розраховується близькість нових кластерів. Кластери DEF та BC схожі та об'єднані разом, щоб сформувати новий кластер. Тепер залишилося два кластери A, BCDEF.

Крок 5: Всі кластери об'єднуються і утворюють єдиний кластер.

Техніку ієрархічної кластеризації можна візуалізувати за допомогою дендрограми.

Дендрограма – це деревоподібна діаграма, на якій записані послідовності злиття та розбиття. Приклад дендограми наведений на рис. 1.4.

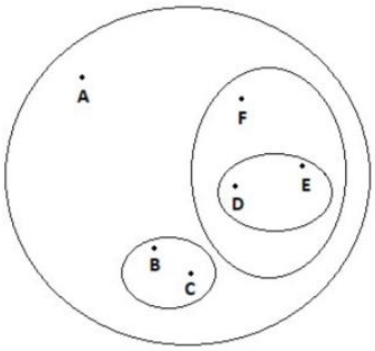

Рисунок 1.4 – Приклад дендограми

Дивізимний метод розподільчої ієрархічної кластеризації мало використовується у нинішній час, тому його в межах магістерської дисертації не розглядаємо.

Обчислення подібності між двома кластерами є важливим для об'єднання або поділу кластерів. Існують певні підходи, які використовуються для розрахунку подібності між двома кластерами:

– за мінімумом;

– за максимумом;

– за середнім за групою;

– за відстанню між центроїдами;

– за методом Уорда.

Серед ітеративних методів кластеризації найбільш поширеним є алгоритм k–середніх.

Алгоритм кластеризації k–means є взагалі найпопулярнішим серед методів кластерного аналізу. Алгоритм відноситься до алгоритмів кластеризації з ітеративним підходом до вирішення задачі та широко застосовується для обробки різних наборів даних і, як наслідок, застосовується у багатьох наукових галузях. Найчастіше кластеризація методом k–середніх застосовується для розпізнавання зображень. Розпаралелювання алгоритму має дозволити обробляти зображення значно швидше шляхом обробки кластерів точок в окремих процесах.

Метод *k*–середніх – це метод кластерного аналізу, метою якого є поділ m спостережень (з простору R^n) на k кластерів, при цьому кожне спостереження відноситься до кластера, до центру (центроїду) якого воно найближче.

В якості міри близькості використовується Евклідова відстань:

$$\rho(x,y) = \|x-y\| = \sqrt{\sum_{p=1}^{n}(x_\rho - y_\rho)^2}, \quad (1.1)$$

де $x, y \in R^n. x, y \in R^n$

Розглянемо низку спостережень:

$$\left(x^{(1)}, x^{(2)}, \dots, x^{(m)}\right), x^{(i)} \in R^n. \quad (1.2)$$

Метод *k*–середніх поділяє *m* спостережень на *k* груп (або кластерів) ($k \leq m$) $S=\{S_1,S_2,\dots,S_k\}$, щоб мінімізувати сумарне квадратичне відхилення точок кластерів від центроїдів цих кластерів:

$$\min\left[\sum_{i=1}^{k}\sum_{x^{(j)}\in S_i}\left\|x^{(i)}-\mu_i\right\|^2\right], \quad (1.3)$$

де $x^{(i)} \in R^n, \mu_i \in R^n$;

μ_i– центроїд для кластера S_i [8].

Алгоритм *k*–середніх виконується так (рисунок 1.5): заздалегідь маючи кількість кластерів, позначаємо їх центри. Випадкові точки розподіляються кластерами залежно від близькості до центру. На виході отримуємо групи (кластери) точок, зібраних навколо кожного із центрів. При цьому кластер містить точки, зібрані за загальною ознакою, а точки між кластерами мають відмінності.

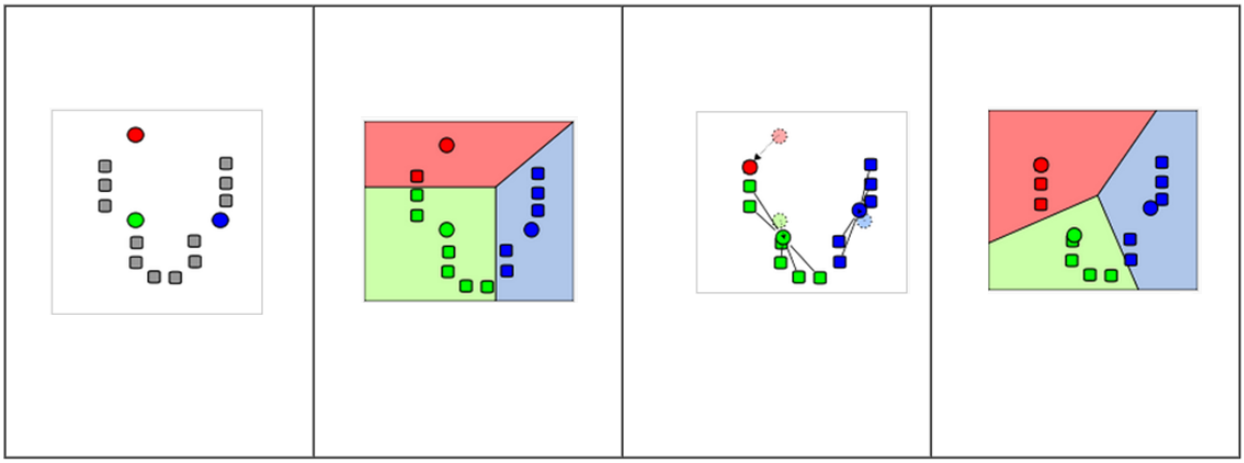

Рисунок 1.5 – Етапи виконання алгоритму кластеризації *k*–середніх

Таким чином, існує безліч алгоритмів кластеризації. Вибір оптимального алгоритму та алгоритму навчання нейронної мережі буде наведено у наступному розділі роботи.

2 ОСОБЛИВОСТІ ЗАСТОСУВАННЯ НЕЙРОННИХ МЕРЕЖ ДЛЯ КЛАСТЕРИЗАЦІЇ ДАНИХ

2.1 Структура і архітектура нейронних мереж

З моменту науково–технічної революції 20 століття людство прагне автоматизувати не тільки виробництво, а й багато інших галузей. Люди навчилися машинізувати та спрощувати фізичну працю.

Нейронна мережа – сучасний тренд, мета якого спростити роботу людини і допомагати їй у математичних розрахунках. А її одне з найважливіших завдань – приймати рішення самостійно, без участі людини.

В роботі [12] стверджується, що термін «нейронна мережа» виник ще в середині 20 століття. А перші роботи в цьому напрямі були виконані американськими вченими Уорреном Мак–Каллоком та Уолтером Піттсом. У 1943 вони розробили комп'ютерну модель нейронної мережі в основі якої закладені математичні алгоритми і теорія діяльності головного мозку. Таким чином, з'явилася перша математична модель біологічного нейрона – нейрон Маккаллока–Піттса. Вони припустили, що така мережа може мати всі риси інтелекту: навчатися, узагальнювати і розпізнавати образи [12].

У 1949 році Дональд Хебб – канадський фізіолог і психолог висловив припущення, що навчання полягає в зміні сили синаптичних зв'язків.

У 1957 році великий крок у бік розвитку ідеї нейронної мережі зробив інший американський вчений Френк Розенблатт. Він розробив математичну та комп'ютерну модель сприйняття інформації мозком людини на основі двошарової нейронної мережі. Для навчання модель використовувала арифметичне додавання та віднімання. А в 1960 році було продемонстровано перший діючий електронний пристрій, який вміло вчитися розпізнавати літери на картках за допомогою пристроїв схожих на кінокамери [12].

Подальший розвиток технології нейронних мереж припинився після публікації у 1969 році книги Мінський М., Пейперт С. "Персептрони". Одніє та найважливішою проблемою було обмеження обчислювальної потужності комп'ютерів тих років, вони не могли ефективно обробляти величезну кількість інформації для навчання великих мереж.

Дослідження в області нейромереж практично не проводилися доти, поки комп'ютери не досягли великих обчислювальних потужностей. Тоді, 1975 року одним із кроків стимулюючих розвиток у цій галузі стала розробка вченим із США Полом Вербсом методу зворотного поширення помилки. У тому ж році японський вчений Куніхіка Фукусіма було запропоновано нову модель нейромережі, названу когнітрон. Когнітрон став однією з перших багатошарових нейронних мереж з архітектурою, яка заснована на будові зорової кори людини. [12]

Перевагами нейронних мереж, які зумовлюють їх широке використання у багатьох сферах життєдіяльності в нинішній час, є:

1. Розв'язання задач при невідомих закономірностях. Використовуючи здатність навчання на безлічі прикладів, нейронна мережа здатна вирішувати завдання, в яких невідомі закономірності розвитку ситуації та залежності між вхідними та вихідними даними. Традиційні математичні методи та експертні системи у таких випадках не є ефективними.

2. Стійкість до шумів у вхідних даних. Можливість роботи за наявності великої кількості неінформативних, шумових вхідних сигналів. Немає необхідності робити їх попереднє відсівання, нейронна мережа сама визначить їх малопридатність для вирішення завдання і відкине їх.

3. Адаптування до змін навколишнього середовища. Нейронні мережі мають здатність адаптуватися до змін навколишнього середовища. Зокрема, нейронні мережі, навчені діяти у певному середовищі, можуть бути легко перевчені для роботи в умовах незначних коливань параметрів середовища. Більше того, для роботи в нестаціонарному середовищі (де статистика змінюється з часом) можуть бути створені нейронні мережі, що

переучуються в реальному часі. Чим вище адаптивні можливості системи, тим стійкішою буде її робота в нестаціонарному середовищі. У цьому слід зазначити, що адаптивність завжди веде до стійкості; іноді вона призводить до абсолютно протилежного результату. Наприклад, адаптивна система з параметрами, що швидко змінюються в часі, може швидко реагувати і на сторонні збудження, що викликає втрату продуктивності.

4. Потенційна надвисока швидкодія. Нейронні мережі мають потенційну надвисоку швидкодію за рахунок використання масового паралелізму обробки інформації.

5. Відмовостійкість при апаратній реалізації нейронної мережі. Нейронні мережі потенційно стійкі до відмови. Це означає, що за несприятливих умов їхня продуктивність падає незначно. Наприклад, якщо пошкоджено якийсь нейрон або його зв'язок, витягання запам'ятованої інформації неможливе. Однак, беручи до уваги розподілений характер зберігання інформації в нейронній мережі, можна стверджувати, що лише серйозні пошкодження структури нейронної мережі суттєво вплинуть на її працездатність. Тому зниження якості роботи нейронної мережі відбувається повільно.

Структура штучної нейронної мережі повторює структуру біологічної нейронної мережі у мозку людини, але зі спрощеннями. Завдяки цій структурі можна отримати модель, здатну аналізувати, запам'ятовувати та відтворювати з пам'яті вхідну інформацію. По суті, нейромережа – це набір нейронів пов'язаних синапсами.

Мережі бувають різними як за структурою, так і за виконуваним завданням. Найпоширенішими застосуваннями нейронних мереж є [13]:

1. Класифікація – розподіл даних за класами. Наприклад, на вхід подається група людей і потрібно вирішити, кому з них схвалити кредит, а кому– ні. Цю роботу може зробити нейронна мережа, аналізуючи такі дані як: вік, платоспроможність, кредитна історія тощо.

2. Прогнозування – можливість передбачати наступний крок. Наприклад, зростання чи падіння котирувань, виходячи із ситуації на біржі.

3. Розпізнавання – нині, найчастіше і найширше застосування нейронних мереж. Використовується в Google, коли виконується пошук за фото або в камерах телефонів, коли воно визначає положення особи та виділяє її та багато іншого.

4. Генерація – генерація контенту, так звана, машинна творчість. Створення аудіо, фото, відео та текстового контенту.

Області застосування нейромереж досить різноманітні, і мабуть у кожній предметній області їм можна знайти застосування. Короткий список областей, де їхнє застосування дозволяє результативно вирішувати поставлені завдання [13]:

– в'язок: стиснення переданого відео та аудіо інформації, швидке кодування–декодування, оптимізація стільникових мереж та схем маршрутизації пакетів;

– робототехніка: розпізнавання сцени, об'єктів та перешкод, що знаходяться в полі зору робота, встановлення оптимального маршруту руху;

– безпека та охоронні системи: розпізнавання осіб; ідентифікація особи за біометричними даними, голосом та підписом; аналіз даних, отриманих з відеокамер та інших зовнішніх сенсорів;

– медицина та охорона здоров'я: постановка діагнозу, обробка медичних зображень та іншої інформації, що отримується з медичних приладів.

– авіоніка: автопілоти, що навчаються, автономні літальні апарати (дрони);

– комп'ютерні та настільні ігри: створення нейрогравців у шашки, шахи та інші популярні ігри [13].

Нейронні мережі є функціональною одиницею глибокого навчання і, як відомо, імітують поведінку людського мозку на вирішення складних завдань, керованих даними.

Вхідні дані обробляються за допомогою різних шарів штучних нейронів, складених разом, для отримання бажаного результату.

Штучний нейрон – математична модель біологічного нейрона. Нейрон у нейронній мережі представлений на рисунку 2.1.

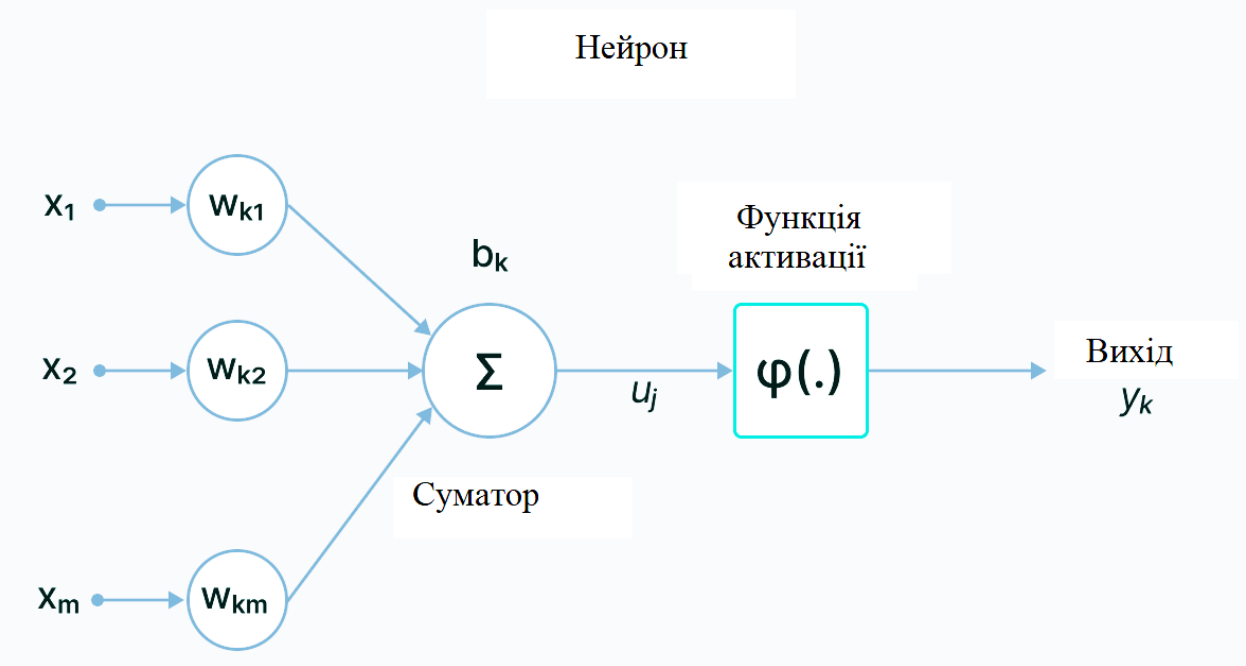

Рисунок 2.1 – Математичний опис нейрона

Вхідні дані– це набір функцій, які вводяться в модель процесу навчання. Наприклад, вхідними даними для виявлення об'єктів може бути масив значень пікселів, що належать до зображення.

Вага. Його основна функція полягає в тому, щоб надати важливості тим функціям, які більшою мірою сприяють навчанню. Робиться це, вводячи скалярне множення між вхідним значенням та матрицею ваг. Наприклад, негативне слово вплине на рішення моделі аналізу настроїв більше, ніж пара нейтральних слів.

Передавальна функція. Завдання передавальної функції полягає в тому, щоб об'єднати кілька вхідних даних в одне вихідне значення, щоб можна було застосувати активацію. Це робиться простим підсумовуванням всіх вхідних даних передавальної функції [14].

Функція активації – вводить нелінійність у роботу персептронів, щоб враховувати різну лінійність вхідних даних. Без цього вихідні дані були б лінійною комбінацією вхідних значень і не могли б внести нелінійність в мережу.

Усунення. Роль усунення полягає у зміщенні значення, отриманого функцією активації. Його роль аналогічна ролі константи у лінійній функції.

Коли кілька нейронів складені разом у ряд, вони становлять шар, а кілька шарів, складених поруч один з одним, називаються багатошаровою нейронною мережею.

Отже, нейромережа – це зв'язка нейронів, кожен нейрон отримує та обробляє отриману інформацію, потім передає її наступному нейрону. Проте, всі нейрони обробляють інформацію однаково, тому найважливішу роль грають синапси, яка з'єднують нейрони. Саме синапси відповідають за правильність результату роботи нейромережі. Вони послаблюють або посилюють сигнал, що проходить між нейронами.

Приклад багатошарової нейронної мережі наведений на рисунку 2.2.

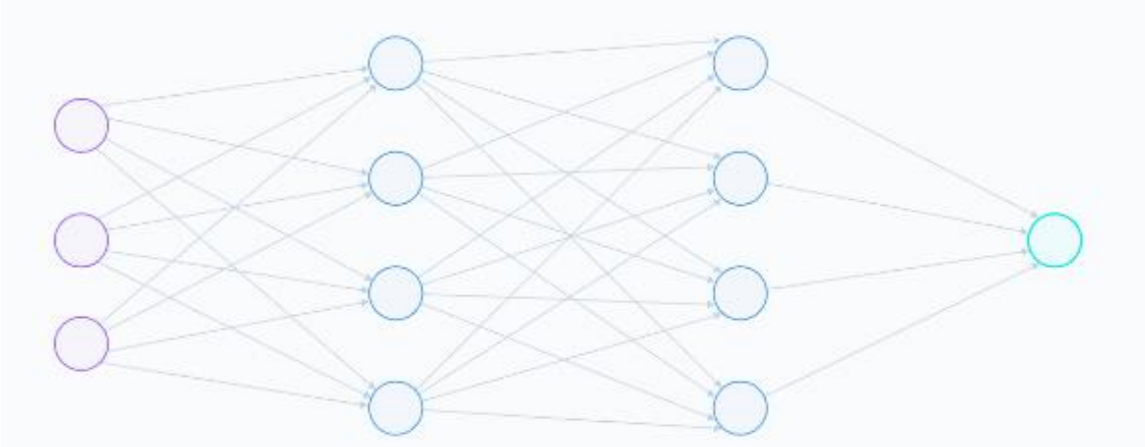

Рисунок 2.2 – Приклад багатошарової нейронної мережі

Вхідний шар. Це дані, які надаємо у модель і які завантажуються у вхідний шар із зовнішніх джерел, таких як CSV–файл або веб–служба. Це

єдиний видимий шар у повній архітектурі нейронної мережі, який передає повну інформацію із зовнішнього світу без жодних обчислень.

Приховані шари. Приховані шари – це те, що робить глибоке навчання тим, чим воно є сьогодні. Це проміжні шари, які виконують всі обчислення та витягують функції даних.

Може бути кілька взаємозалежних прихованих шарів, які враховують пошук різних прихованих функцій даних. Наприклад, при обробці зображень перші приховані шари відповідають за функції вищого рівня, такі як краї, форми або межі. З іншого боку, пізніші приховані шари виконують складніші завдання, такі як ідентифікація цілих об'єктів (автомобіль, будинок, людина).

Вихідний шар. Вихідний шар отримує вхідні дані від попередніх прихованих шарів та робить остаточний прогноз на основі знань моделі. Це найважливіший шар, де отримується кінцевий результат.

У разі моделей класифікації/регресії вихідний шар має один вузол. Тим не менш, це повністю залежить від проблеми і залежить від того, як було побудовано модель.

Персептрон – це найпростіша архітектура нейронної мережі. Це тип нейронної мережі, який приймає ряд вхідних даних, застосовує певні математичні операції до цих вхідних даних та виробляє вихідні дані. Він приймає на вхід вектор дійсних значень, виконує лінійну комбінацію кожного атрибуту з відповідною вагою, яка присвоєна кожному з них.

Зважене введення підсумовується в одне значення і передається через функцію активації.

Ці блоки персептрону об'єднуються, щоб сформувати більшу архітектуру штучної нейронної мережі.

Таким чином, алгоритм дії простого штучного нейрона такий:

– на вхід нейрона надходять сигнали;

– сигнал перемножується на коефіцієнт синапсу і підсумовується з усіма сигналами, що надходять аналогічним чином;

– підсумкове значення сигналу надходить на функцію активації, яка перетворює інформацію на вихідний сигнал;

– вихідний сигнал надходить на вихід нейрона;

– функція активації використовується для нормалізації сигналу.

Наприклад, якщо сигнали на кількох нейронах сильно відрізняються, для коректності виведення необхідно привести значення певного діапазону.

Часто використовувані функції активації представлені на рисунку 2.3.

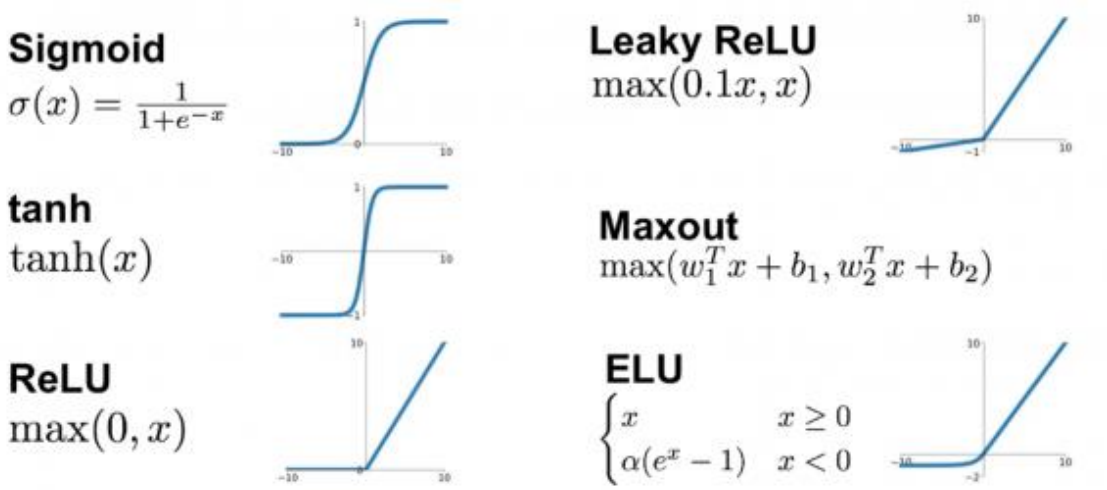

Рисунок 2.3 – Часто використовувані функції активації нейронів

Сигмоїда – одна з найпопулярніших функцій активації, яка використовується, якщо необхідно нормалізувати дані в діапазоні [0,1]. Працювати з від’ємним значеннями з цією функцією не можливо, для цього можна скористатися гіперболічним тангенсом. Його діапазон значень [–1,1]. Функція ReLU, яку часто називають випрямляч, негативну зважену суму перетворює на 0, а позитивну не змінює.

Таким чином, персептрон показує, як працює один нейрон. Серія персептронів, складених у різні шари і в тому випадку, коли інформація передається в одному напрямку, називається мережею прямого поширення.

При прямому проході інформація надходить усередину моделі через вхідний шар, проходить через ряд прихованих шарів і нарешті надходить на

вихідний шар. Ця архітектура нейронних мереж є прямою за своєю природою – інформація не зациклюється на двох прихованих шарах.

Пізніші шари не дають зворотного зв'язку з попередніми шарами. Основний процес навчання у мережах із прямим зв'язком залишається таким самим, як і в персептроні.

Також відомі залишкові мережі ResNets. Основна ідея ResNet полягає в тому, що глибша мережа може бути створена з дрібної мережі шляхом копіювання ваги з дрібних аналогів з використанням зіставлення ідентифікаторів.

Рекурентні нейронні мережі. Базова архітектура глибокого навчання має фіксований розмір вхідних даних, і це діє як блокувальник у сценаріях, де розмір вхідних даних не фіксований. Крім того, рішення, прийняті моделлю, ґрунтуються на поточних даних без пам'яті минулого.

Рекурентні нейронні мережі дуже добре працюють із послідовностями даних як вхідні дані. Їх функціональність можна побачити у вирішенні завдань NLP, таких як аналіз емоцій, спам–фільтри, проблеми часових рядів, такі як прогнозування продажів, прогнозування фондового ринку тощо.

Рекурентні нейронні мережі здатні запам'ятовувати те, чого вони навчилися у минулому, і застосовувати це у майбутніх прогнозах.

Вхід знаходиться у формі послідовних даних, які подаються в RNN, яка має прихований внутрішній стан, який оновлюється щоразу, коли мережа зчитує наступну послідовність даних на вході.

Внутрішній прихований стан буде повернуто моделі. RNN робить деякий вивід для кожної мітки часу.

У RNN кожен із прогнозів дивиться назад лише на одну тимчасову мітку, і в нього дуже короткочасна пам'ять. Він не використовує жодної інформації з далекого минулого.

Приклад архітектури рекурентної нейронної мережі представлений рисунку 2.4.

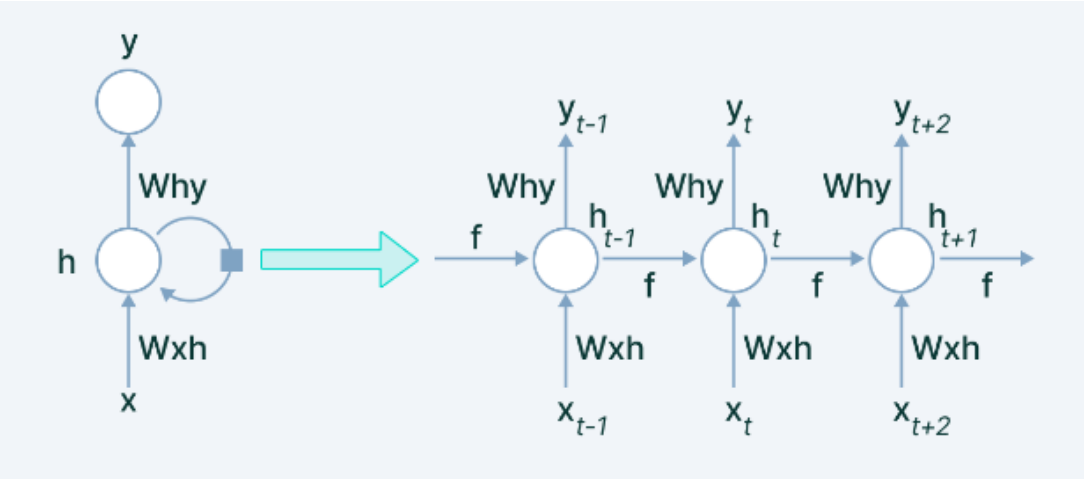

Рисунок 2.4 – Приклад архітектури RNN

Щоб виправити це, можна взяти структуру рекурентних нейронних мереж і розширити її, додавши до неї ще кілька частин.

Критична частина, яку додаємо до цієї рекурентної нейронної мережі, – це пам'ять. Необхідно, щоб нейрон міг пам'ятати, що сталося багато тимчасових міток тому. Для цього потрібно додати до структури штучної нейронної мережі додаткові структури, які називаються гейтами.

Така мережа називатиметься мережею LSTM.

Архітектура мережі LSTM представлена на рисунку 2.5.

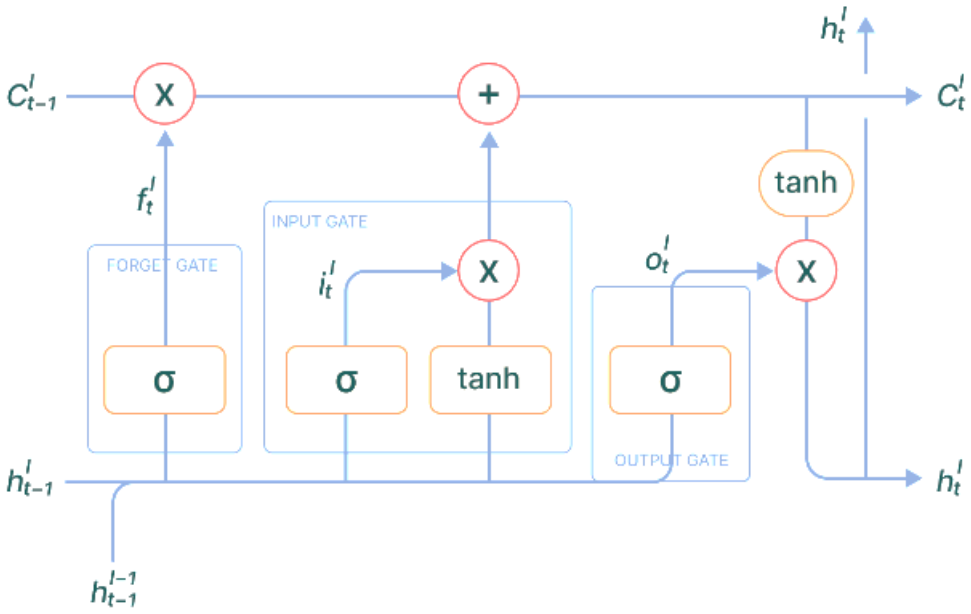

Рисунок 2.5 – Архітектура мережі LSTM

Відстань (C_t'):відповідає вмісту довгострокової пам'яті мережі.

Forget Gate: деяка інформація у стані осередку більше не потрібна і стирається. Гейт отримує два входи: поточну часову мітку і попередній стан комірки, помножені на відповідні вагові матриці до додавання зміщення.

Результат відправляється у функцію активації, яка виводить двійкове значення, що визначає, чи буде інформація збережена чи забута.

Input Gate: цей елемент вирішує, яка частина нової інформації має бути додана до стану осередку. Це схоже на елемент Forget Gate, що використовує поточну тимчасову мітку і попередній стан комірки, з тією різницею, що множиться на інший набір ваг [14].

Output Gate: даного елемента полягає в тому, щоб витягувати значну інформацію з поточного стану осередку і надавати її як вихід.

Згорткові нейронні мережі (CNN) – це тип нейронних мереж з прямим зв'язком, що використовуються в таких завданнях, як аналіз зображень, обробка природної мови та інші складні завдання класифікації зображень.

CNN має приховані шари, які становлять основу ConvNets.

На більш високому рівні згорткові шари виявляють шаблони даних зображення за допомогою фільтрів. Про деталі вищого рівня «дбають» перші кілька згорткових шарів.

Чим глибше йде мережа, тим складнішим стає пошук шаблонів. При додаванні шару згортки в мережу потрібно вказати кількість фільтрів.

Фільтр можна розглядати як відносно невелику матрицю, для якої необхідно визначити кількість рядків та стовпців у цій матриці.

Значення цієї матриці ознак ініціалізується довільними числами. Коли цей згортковий шар отримує значення пікселів вхідних даних, фільтр виконуватиме згортку по кожній ділянці вхідної матриці.

Вихідні дані згорткового шару зазвичай проходять через функцію активації ReLU, щоб надати моделі нелінійності. Як вже зазначалося, даний метод бере карту об'єктів та замінює всі негативні значення нулями.

Об'єднання в пул є дуже важливим кроком у ConvNets, оскільки даний тип скорочує обсяг обчислень і робить модель стійкою до спотворень та варіацій. Повні пов'язані щільні нейронні мережі будуть використовувати плоску матрицю ознак і прогнозувати відповідно до варіанта використання.

Ознаки на виході є зваженими сумами вхідних ознак, що знаходяться приблизно в тому ж місці, що вихідний піксель на вхідному шарі. Незалежно від того, чи потрапляє вхідна ознака в те саме місце, вона визначається залежно від того, знаходиться чи знаходиться вона у зоні ядра, що створює вихідні дані, чи ні.

Це означає, що розмір ядра згортки відповідає за кількість ознак, об'єднаних для отримання на виході нової ознаки.

Приклад операції двовимірної згортки наведений на рисунку 2.6.

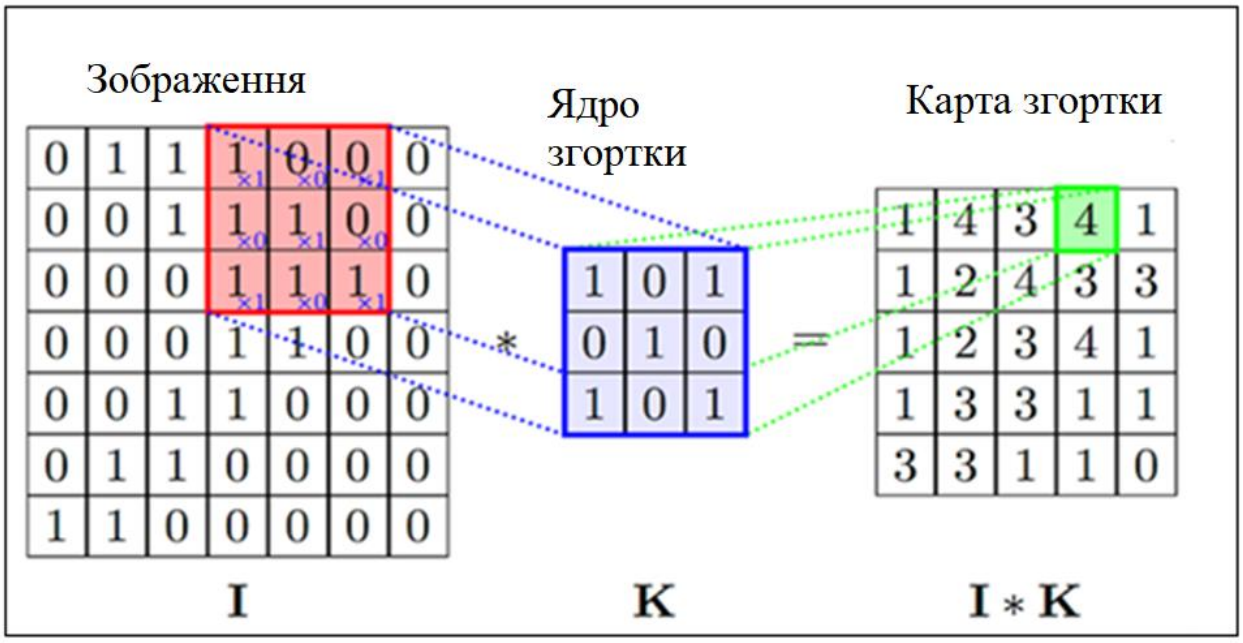

Рисунок 2.6 – Приклад операції двовимірної згортки

Крім іншого, використання згорткових шарів зменшує складність обчислень, що проводяться мережею.

Наприклад, на рисунку 2.6 маємо 7x7=49 ознак на вході та 5x5=25 на виході за допомогою ядра згортки, що складається з 3x3=9 вагових коефіцієнтів.

Використовуючи повнозв'язковий шар, для такого ж результату знадобилася б 49x25 = 1225 ваг і кожна вихідна ознака була б зваженою сумою всіх ознак на вході. Згортковий шар дозволяє аналізувати ознаки в межах однієї частини зображення [14].

Отже:

– ядра згортки поєднують пікселі лише з невеликої локальної області для формування виходу. Виходить, що вихідні ознаки можуть бачити лише вхідні ознаки з невеликої локальної області.

– ядро згортки застосовується глобально по всьому зображенню для створення матриці вихідних значень.

Підвибірковий шар, він же субдискретизуючий, служить для зміни розмірності згорткових карт попереднього шару у бік зменшення. Зазвичай, цей шар має ядро 2x2, що дає зменшення карти попереднього шару в 2 рази.

Ця карта ділиться на осередки 2x2 і з кожного такого осередку вибирається більше число, що дає ущільнення зображення від більш детального до менш детального.

Також, до кожного значення застосовується функція активації, найчастіше це ReLU. Приклад операції вибірки наведено на рисунку 2.7.

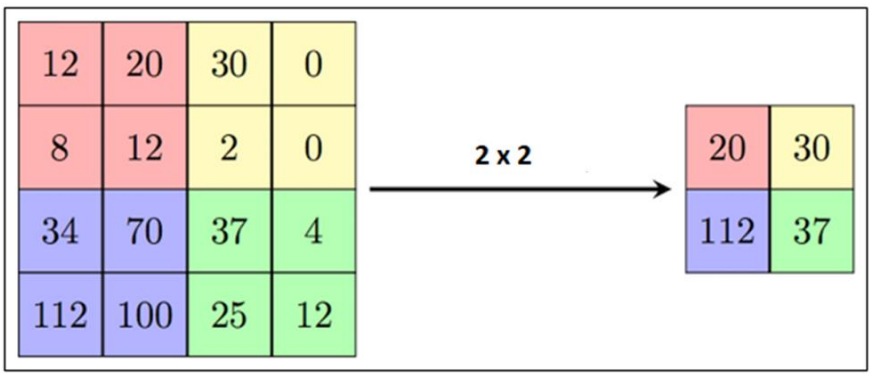

Рисунок 2.7 – Приклад операції вибірки

Таким чином, істотною особливістю архітектури згорткової нейронної мережі є те, що розміри вхідних даних стають все менше і менше від початку до кінця мережі, а кількість каналів більше.

Деконволюційні нейронні мережі – це CNN, які працюють у зворотному порядку.

Коли використовуються згорткові шари та максимальний пул, розмір зображення зменшується. Щоб повернутися до вихідного розміру, необхідно використати підвищення частоти дискретизації та транспонування згорткових шарів.

Підвищення дискретизації не має параметрів, що навчаються — воно просто повторює рядки і стовпці даних зображення з відповідними розмірами [14].

Транспонування згорткового шару означає одночасне застосування згорткової операції та підвищення частоти дискретизації.

Цей процес називається Conv2DTranspose. Якщо встановлюється крок = 1, не буде дискретизації, що підвищує, і отримується вивод того ж розміру на вході.

Також окремо виділимо наступний тип нейронної мережі.

Повнозв'язкова нейронна мережа прямого поширення, вона ж багатошаровий персептрон – мережа в якій кожен нейрон пов'язаний з іншими нейронами сусідніх шарів, а всі зв'язки спрямовані від входу до виходу.

Графічне подання повнозв'язкової нейронної мережі прямого поширення представлене на рисунку 2.8.

Блакитні нейрони – нейрони вхідного шару, зелені – прихований шар та сині – вихідний шар.

Ця мережа здатна вирішувати завдання прогнозування двох параметрів, з урахуванням трьох деяких вхідних даних. Наприклад, прогнозувати температуру повітря і вологість, виходячи з даних про тиск, швидкість вітру і

точку роси. Між вхідним та вихідним шарами розташовуються приховані шари.

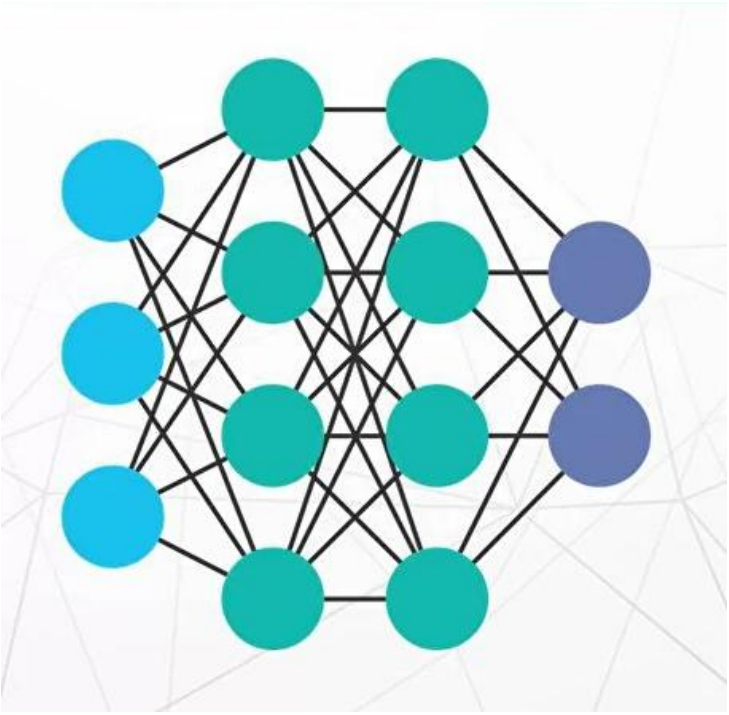

Рисунок 2.8 – Графічне подання повнозв'язкової нейронної мережі прямого поширення

Кожен нейрон цих верств узагальнює інформацію і виводить певні ознаки, якими будується прогноз. Перший нейрон прихованого шару визначає найбільш низькорівневі ознаки, наприклад, якщо низький тиск, значить погода погіршується.

Наступні приховані шари виділяють все більш високорівневі ознаки – якщо погода погіршується, ймовірно, йде дощ або град, отже, вологість підвищена. Якщо тиск її знижено, то нейрон, що відповідає за цю ознаку, буде передавати нульовий сигнал наступним нейронам, щоб не вносити поправок до загальної системи.

Таким чином, на активацію нейрона впливатимуть дані про тиск. Якщо тиск навпаки підвищений, нейрон може передавати від’ємний сигнал. В даному випадку нейронна мережа сама розподіляє відповідальність ознаки.

Загальний принцип виділення властивостей – все більше узагальнення під час переходу з одного шару на інший, від низького рівня до вищого.

Отже, можна дійти до висновку, що головними компонентами нейронної мережі є:

– її архітектура – скільки у мережі прихованих шарів, вхідних та вихідних параметрів і як вони пов'язані, а також які активаційні функції застосовуються. Ці питання вирішує розробник;

– навчені ваги – спочатку ваги задаються випадковим чином та підбираються у процесі навчання. Зберігаються вони як сукупності матриць дійсних чисел. Таким чином, нейронну мережу можна зберегти шляхом збереження ваги та архітектури.

На основі аналізу джерел [12–15] складено таблицю 2.1, у якій наведено основні види нейронних мереж та їх основні сфери застосування.

Таблиця 2.1 – Основні види нейронних мереж та сфери застосування

Тип	Мета
Автоенкодер (AE)	Зазвичай AE використовується для зменшення кількості аналізованих випадкових величин, щоб система могла вивчити подання для набору даних і, отже, обробити генеративні моделі даних.
Двонаправлена рекурентна нейронна мережа (BRNN)	Мета BRNN – збільшити інформаційні входи, доступні для мережі шляхом підключення двох прихованих, спрямованих протилежних шарів до одного і того ж виходу. Використовуючи BRNN, вихідний шар може отримувати інформацію як з минулого, так і майбутнього стану.
Машина Больцмана (BM)	Рекурентна нейронна мережа, цей алгоритм здатний вивчати внутрішні уявлення і може представляти

	та вирішувати складні комбіновані завдання.

Продовження таблиці 2.1

Згорткова нейронна мережа (CNN)	CNN, що найчастіше використовуються для аналізу візуальних образів, являє собою нейронну мережу з прямим зв'язком, призначену для мінімізації попередньої обробки.
Деконволюційна нейронна мережа (DNN)	DNN дозволяють неконтрольовану побудову ієрархічних уявлень зображень. Кожен рівень ієрархії групує інформацію з попереднього рівня додавання складніших функцій.
Мережа глибокого переконання (DBN)	При навчанні на неконтрольованому наборі прикладів DBN може навчитися реконструювати свої вхідні дані, ймовірно, використовуючи шари в якості детекторів ознак. Наслідуючи цей процес, можна навчити DBN виконувати контрольовані класифікації.
Мережа глибокої згорткової зворотної графіки (DCIGN)	Модель DCIGN призначена для вивчення інтерпретованого представлення зображень, які система розділяє відповідно до елементів тривимірної структури сцени. DCIGN використовує безліч рівнів операторів як згорткових, так і деконволюційних.
Глибока залишкова мережа (DRN)	DRN допомагають у вирішенні складних завдань та моделей глибокого навчання. Маючи багато рівнів, DRN запобігає погіршенню результатів.
Автоенкодер із шумозаглушенням (DAE)	DAE використовується для відновлення даних із пошкоджених вхідних даних; алгоритм змушує прихований шар вивчати надійніші функції. В результаті

	на виході виходить точніша версія вхідних даних.
Мережа стану (ESN)	ESN працює з випадковою великою фіксованою рекурентною нейронною мережею, в якій кожен вузол отримує нелінійний сигнал у відповідь. Алгоритм

Продовження таблиці 2.1

	випадковим чином встановлює та призначає ваги та можливості підключення для досягнення гнучкості навчання.
Машина екстремального навчання (ELM)	Цей алгоритм вивчає вихідні ваги прихованих вузлів за крок, створюючи лінійну модель. ELM можуть добре узагальнювати і навчатися набагато швидше, ніж мережі зворотного поширення.
Нейронна мережа прямого поширення (FF або FFNN) та персептрон (P)	Це базові алгоритми нейронних мереж. Нейронна мережа прямого поширення – це штучна нейронна мережа, в якій сполуки вузлів не утворюють цикл; персептрон – це бінарна функція, що має лише два результати (вгору/вниз; так/ні, 0/1).
Закритий рекурентний блок	ЗРБ використовують з'єднання через послідовності вузлів для виконання завдань машинного навчання, пов'язаних із кластеризацією та пам'яттю. ЗРБ уточнюють вихідні дані за допомогою управління інформаційним потоком моделі.
Генеративна змагальна мережа (GAN)	Ця система протиставляє дві нейронні мережі – дискримінаційну та генеративну. Мета полягає в тому, щоб розрізняти реальні та синтетичні результати для моделювання концептуальних завдань високого рівня.
Мережа Хопфілда (HN)	Ця форма рекурентної штучної нейронної мережі є системою асоціативної пам'яті з бінарними пороговими вузлами. Створена для зведення до локального мінімуму,

	HN є модель розуміння людської пам'яті.
Мережа Кохонена (KN)	KN організує проблемний простір у двовимірну картку. Різниця між картами, що самоорганізуються (SOM) та іншими підходами до вирішення проблем полягає в тому,

Продовження таблиці 2.1

	що SOM використовують конкурентне навчання, а не навчання з виправленням помилок.
Машина рідких станів (LSM)	LSM, відомий як машинне навчання третього покоління (або імпульсна нейронна мережа), додає поняття часу як елемента. LSM генерують активацію просторово–часової нейронної мережі, оскільки вони зберігають пам'ять під час обробки.
Довготривала / короткочасна пам'ять (LSTM)	LSTM здатний вивчати чи запам'ятовувати залежність порядку у задачах прогнозування, що стосуються послідовності. Блок LSTM містить комірку, вхідний вентиль, вихідний вентиль та вентиль забування. Осередки зберігають значення протягом довільних інтервалів часу. Кожен блок регулює потоки значень через LSTM з'єднання.
Ланцюг Маркова (MC)	MC – це математичний процес, який описує послідовність можливих подій, в якій ймовірність кожної події залежить виключно від стану, досягнутого в попередній події.
Нейронна машина Тюрінга (NTM)	Ґрунтуючись на роботі фахівця з обробки даних Алана Тюрінга середини 20–го століття, NTM виконує обчислення та розширює можливості нейронних мереж за рахунок зв'язку із зовнішньою пам'яттю. Розробники використовують HTM у роботах і розглядають дану мережу як один із засобів створення штучного людського

	мозку.
Мережі радіальних базисних функцій (мережі RBF)	Розробники використовують мережі RBF для моделювання даних, які становлять основну тенденцію чи функцію. Мережі RBF навчаються апроксимувати

Продовження таблиці 2.1

	основний тренд, використовуючи криві дзвони чи нелінійні класифікатори. Нелінійні класифікатори аналізують глибше, ніж прості лінійні класифікатори, які працюють із векторами меншої розмірності.
Рекурентна нейронна мережа (RNN)	RNN моделюють послідовні взаємодії через пам'ять. На кожному тимчасовому кроці RNN обчислює нову пам'ять або прихований стан, залежно від поточного вхідного та попереднього стану пам'яті.
Обмежена машина Больцмана (RBM)	RBM – це ймовірна графічна модель в неконтрольованому середовищі. RBM складається з видимих та прихованих шарів, а також зв'язків між бінарними нейронами у кожному з цих шарів. RBN корисні для фільтрації та класифікації.

2.2 Алгоритми навчання нейронних мереж

Спершу треба розглянути основні типи навчання нейронних мереж.

1. З учителем. Навчання з учителем – це одне з найпопулярніших і найчастіших завдань. У цьому випадку маємо вибірку, і знаємо по ній правильні відповіді. Навчання з учителем краще підходить для завдань, у яких є досить великий набір достовірних даних для навчання алгоритму. Але так буває далеко не завжди. Недолік даних – проблема, що найбільш часто зустрічається [15].

2. Навчання без учителя. Найчастіше ідеально розмічені та чисті дані дістати нелегко. Тому перед алгоритмом стоїть завдання знайти наперед не відомі відповіді. У таких ситуаціях використовують навчання без учителя. У навчанні без вчителя модель має набір даних без явних вказівок, що з нею робити. Нейронна мережа намагається самостійно знайти кореляції даних, намагаючись витягти корисні ознаки та аналізуючи їх.

3. Навчання із підкріпленням. Навчання з підкріпленням – один із способів машинного навчання, у ході якого випробувана система (агент) навчається, взаємодіючи з деяким середовищем. Коли агент робить дії, що сприяють досягненню мети, він отримує нагороду. Глобальна мета – передбачати такі кроки, щоб заробити максимальну нагороду зрештою. Так як це ітеративний процес, то чим більше зворотного зв'язку отримає агент, тим краще стає стратегія. Цей підхід особливо корисний, наприклад, для навчання роботів, які керують безпілотними та автономними транспортними засобами.

Основними алгоритмами навчання нейронної мережі є [15]:

1. Метод зворотного поширення. Цей метод також називають Backpropagation. Він є одним із основних способів навчання та містить у своїй основі алгоритм обчислення градієнтного спуску. Іншими словами, рухаючись вздовж градієнта, відбувається розрахунок локального максимуму та мінімуму функції.

Для кращого розуміння процесу необхідно перевести функцію в графік, який відображатиме залежність значень помилки від ваги синапсу. На отриманій кривій потрібно визначити точку з найменшим та найбільшим показником. У той же час необхідно графічно відобразити всі ваги і розрахувати для кожного з них глобальний мінімум.

Значення градієнта матиме векторну величину, яка дасть уявлення про напрям і крутість схилу. Пошук значення градієнта здійснюється шляхом обчислення похідної від функції необхідної точки. Така точка матиме значення ваги, розподілене випадковим чином. У ній слід проводити

розрахунок градієнта та визначати спрямованість руху спуску. Обчислення необхідно проводити послідовно у всіх точках, поки не буде досягнуто локального мінімуму, що зупиняє подальший спуск.

Щоб подолати цей етап, потрібно задати таке значення для моменту, який дозволить пройти ділянку графіка і опинитися у потрібній точці. У разі недостатнього значення подолати опуклість не вдасться, а якщо значення буде надто великим, то висока ймовірність «проскоку» глобального мінімуму.

На загальну швидкість навчання нейромережі впливає як момент прискорення, так й значення, що є гіперпараметром і що визначається методом підбору.

Найбільш сприятливе поєднання значень неможливо знати заздалегідь. Воно виявляється в ході кількох навчань та коригування в потрібну сторону.

Сам метод навчання – це процес, у якому дані, що надходять, поширюються між нейронами з допомогою синапсів. Передача здійснюється доти, доки дані не досягнуть шару «виходу», трансформувавшись у відповідь. Ця операція зветься «передача вперед».

Як тільки відповідь отримана, відбувається розрахунок помилки, і відповідно до неї виконується зворотна передача. Мета такої дії – приведення синаптичних ваг до оптимальних значень під час руху від вихідного шару до вхідного.

Для такого алгоритму навчання нейронних мереж необхідно використовувати функції активації, що диференціюються. Це пов'язано з тим, що поширення у зворотному напрямку визначається різницею між відповідями, а також добутком між ним та похідною функцією від вхідного значення.

2. Метод Rprop. Він був запропонований як альтернатива попередньому способу навчання, який потребує занадто багато часу і стає незручним, якщо результати потрібно отримати в короткий термін. Для

збільшення швидкості операцій було розроблено багато допоміжних алгоритмів.

Цей метод є основним під час навчання за принципом epoch (один повний прохід датасету через нейронну мережу).

Якщо на стадії обчислень похідна змінює свій знак на протилежний, то це говорить про надто велику зміну та про упущення локального мінімуму. Отже, потрібно повернути вагу попереднього значення та зменшити шаг зміни. Якщо ж знак залишився тим самим, слід підняти величину зміни ваги для максимальної збіжності.

Якщо закріпити ключові показники підстроювання ваги, то можна не налаштовувати глобальні параметри – це є додатковим плюсом використання методу. Причому є готові значення таких показників. Їх застосування рекомендовано, але жорстких рамок на вибір значень немає [15].

При розрахунку цього значення необхідно дотримуватись деяких правил. Якщо в певній точці похідна змінює свій знак з «+» на «–», то це говорить про зростання помилки. Тому вагу потрібно змінити у менший бік. У протилежній ситуації – вагу треба збільшити.

У цьому випадку порядок операцій буде таким:

– визначення значення корекції;

– розрахунок похідних;

– розрахунок нової величини корекції вагових значень;

– коригування ваг.

Якщо умова зупинки алгоритму не виконується, відбувається повернення до розрахунку похідних, і цикл запускається по новому колу.

3. Генетичний алгоритм навчання. Ще один поширений підхід – це навчання нейронної мережі генетичним алгоритмом (Genetic Algorithm). За своїм принципом він схожий з еволюційними процесами природи, що ґрунтуються на комбінуванні (схрещуванні) результатів.

Іншими словами, відбувається природний відбір, де нове покоління є продуктом комбінації результатів із найкращими властивостями. Якщо

результат такого схрещування не підходить за якимись критеріями, то відбір відбувається знову, поки продукт стане досконалим.

Завершення алгоритму відбувається у момент, коли закінчуються відведені йому спроби чи час на мутацію. При цьому результат може бути недосягнутим. Даний метод використовується для покращення показників ваг за умови, що структура задана за умовчанням. Вага при цьому має бути прописана двійковим кодом, а повний набір ваги сформує підсумковий результат. Розрахунок помилки на виході зумовлює оцінку ефективності.

2.3 Вибір оптимального алгоритму навчання нейронної мережі для кластеризації даних

На підставі проведеного огляду, а, зокрема, роботи [16], де запропоновано алгоритм кластеризації, що поєднує в собі метод k–середніх і багатошаровий персептрон, прийнято рішення використовувати саме це поєднання, тому що воно дає найкращі результати в порівнянні з результатами, представленими в [17,18,19].

Для навчання нейронної мережі буде використовуватися алгоритм зворотного розповсюдження помилки. Метод *k*–середніх буде використовуватися на початковому етапі для формування навчальної вибірки.

3 МОДЕЛЮВАННЯ ПРОЦЕСУ НЕЙРОМЕРЕЖЕВОЇ КЛАСТЕРИЗАЦІЇ ДАНИХ

3.1 Застосування середовища Scilab для кластеризації даних на основі нейронних мереж

В межах даного розділу розглянемо приклад нейромережевої кластеризації даних на основі нейронних мереж прямого поширення.

Для побудови нейронних мереж можливе застосування середовища Scilab [20].

Програмний продукт Scilab полегшує процедуру конструювання, навчання та використання ШНМ. Користувачу не потрібно досконало розуміти механізм створення нейромережі, варто лише керуватися запитами програми.

Для того, щоб почати працювати з нейронними мережами в SciLab, необхідно завантажити відповідний модуль, для цього потрібно перейти по вкладці Інструменти – Керування модулями ATOMS (рисунок 3.1).

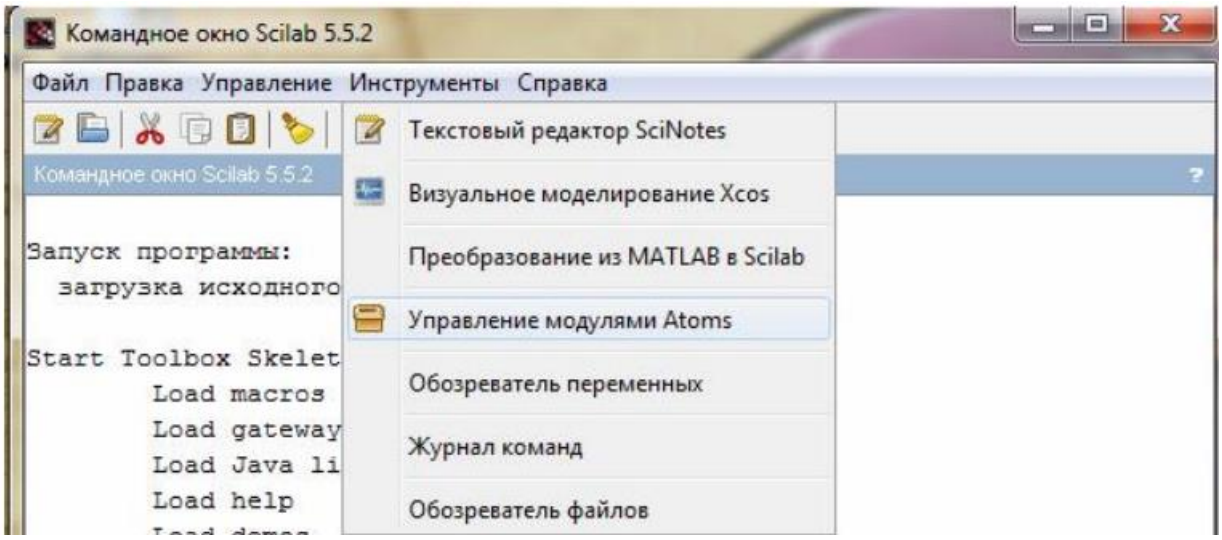

Рисунок 3.1 – Вкладка запуску установки модулів ATOMS

За допомогою програми можна зображувати процес навчання у вигляді графіка, а отже, користувач може визначити, як проходив процес навчання та скільки циклів навчання знадобилося, щоб помилка навчальної множини досягла потрібного рівня.

Важливою перевагою використання програми Scilab є можливість переглядати та коригувати ваги штучної нейронної мережі. Більшість програм нейромережевого моделювання позбавлені цієї важливої функції.

Приклад навчання мережі в середовищі Scilab наведений на рисунку 3.2.

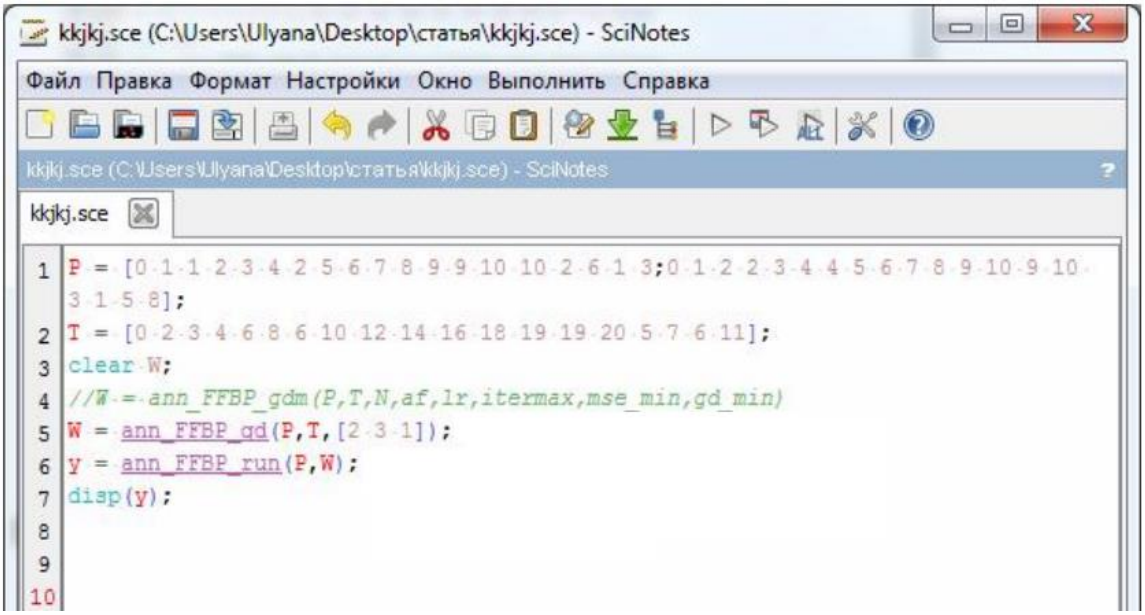

Рисунок 3.2 – Приклад навчання мережі в середовищі Scilab

Таким чином, дане середовище можна використовувати для нейромережевої кластеризації даних.

3.2 Приклад нейромережевої кластеризації в середовищі Scilab

Наведемо приклад нейромережевої кластеризації даних за принципом, наведеним у другому розділі магістерської дисертації.

Визначивши міру близькості до центроїду, розбиваємо об'єкти на кластери шляхом визначення центроїдів цих кластерів. Число кластерів k задається дослідником заздалегідь.

Розглянемо початковий набір k середніх (центроїдів) у кластерах. На першому етапі центроїди кластерів вибираються випадково або за певним правилом (наприклад, вибрати центроїди $\mu_1, \dots, \mu_k$, що максимізують початкові відстані між кластерами $S_1, S_2, \dots, S_k$).

Відносимо спостереження до тих кластерів, чиє середнє (центроїд) до них найближче. Кожне спостереження належить лише одному кластеру, навіть якщо його можна віднести до двох і більше кластерів.

Потім центроїд кожного і кластера перераховується за наступним правилом:

$$\mu_i = \frac{1}{S_j} \sum_{x^{(j)} \in S_i} X^{(j)} \qquad 3.1)$$

Таким чином, алгоритм *k*–середніх полягає у перерахуванні на кожному кроці центроїду для кожного кластера, отриманого на попередньому кроці.

Алгоритм зупиняється, коли значення $\mu_1 \mu_i^{\text{крок } t} = \mu_i^{\text{крок } t+1}$ змінюються.

Варто зазначити, що неправильний вибір початкової кількості кластерів *k* може призвести до некоректних результатів. Саме тому при використанні методу *k*–середніх важливо спочатку провести перевірку відповідного числа кластерів для набору даних.

Припустимо, що файл даних починається зі смуг з *1N* для першого пікселя, за якими слідують смуги з q по *N* для другого пікселя і т. д. Засіб кластеризації ініціалізується шляхом вибору рівномірно розташованих пікселів по діагоналі обробленого зображення. Основна частина обчислення для алгоритму нейромережевої кластеризація присвячена етапам призначення пікселя та оновлення кластера, які будуть розглянуті по черзі.

Код призначення пікселів (спрощено) для нейронної мережі виглядатиме наступним чином:

```
for (i = 0; i<numPixels; i++)
{
pixel = image + i * numBands;
nearestCluster = 0;
minDist = distance(pixel, centers[0], numBands);
for (j = 1; j<numClusters; j++)
{
dist = distance(pixel, centers[j], numBands);
if (dist <minDist)
{
minDist = dist;
nearestCluster = j;
}
}
clusterMap[i] = nearestCluster;
}
if (clusterMap[i] != nearestCluster)
++numChanged;
clusterMap[i] = nearestCluster;}
```

Хоча у наведеному вище коді перераховані лише два вкладені цикли, є третій вкладений цикл, який відбувається у функції distance, яка визиває всю смугу пропускання для підтримки множини *N*–вимірних векторів. Оновлення засобу кластера складається з двох основних циклів: одного для підсумовування пікселів, призначених кожному кластеру, та іншого, щоб ділити на кількість пікселів, що призначаються для кожного рівня, щоб визначити його значення. Підсумовування пікселів та визначення значення кластера наведене у фрагменті лістингу нижче:

```
for (i = 0; i< (int) numPixels; i++)
{
```

```
    addTo(image + i * numBands, centers[clusterMap[i]],
numBands);
    clusterCounts[clusterMap[i]] +=1;
    }
    for (i = 0; i<(int) numClusters; i++)
    {
    for (j = 0; j<numBands; j++)
    centers[i][j] /= clusterCounts[i];}
```

У першому циклі для функції `addTo` накопичується перший аргумент (`pixelvector`), щоб далі підсумувати всі пікселі, пов'язані з кожним кластером. Другий цикл виконує поділ, необхідний створення оновленого засобу кластера.

Перед розпаралелюванням алгоритму корисно оцінити обчислювальне навантаження серійного алгоритму. Для фази призначення пікселів нейронної мережі обчислювальне навантаження керується трьома вкладеними циклами, які перебирають по всіх пікселях, всіх кластерах і діапазонах відповідно.

Для типових гіперспектральних застосувань порядки цих величин представлені у таблиці 3.1.

Таблиця 3.1 – Величини властивостей

Параметри	Величина
numPixels	$O(10^5)$
numClusters(k)	$O(10^1)$
numBands	$O(10^2)$

Кластеризація з обчисленням Евклідової відстані призводить до операцій множення та акумулювання для фази пікселя. Для порівняння, фаза оновлення кластера вимагає лише $O(10^7)$ доповнень до сум вкладів (без множень) та $O(10^3)$ поділів для обробки обчислень оновлених засобів.

Таким чином, на етапі впорядкування кластера значно більше обчислень, ніж на фазі оновлення коштів.

Паралелізація зовнішнього циклу для нейромережевої кластеризації максимізує обсяг обчислень. Додаткові обчислення можуть бути досягнуті шляхом зміни порядку проходження петель таким чином, щоб ітерація пікселів відбувалася у внутрішньому циклі; однак це призведе до частих промахів у кеші, тому цикли будуть збережені, як є. Оскільки кількість обчислень на піксель постійна, статичне планування потоків обиралося встановленням змінного середовища OMP_SCHEDULE = static, 1000.

Оскільки більшість обчислень відбувається під час фази призначення пікселя ітерації k–середніх, і всі пікселі обробляються незалежно протягом цієї фази, для стратегії розпаралелювання MPI обрано декомпозицію домену.

Оскільки MPI є мультипроцесною бібліотекою (на відміну мультипоточної), модифікація вихідного коду для розпаралелювання з MPI значно інвазивніша, ніж OpenMP. Для пікселів numPixels у гіперспектральному зображенні та процесах numProcs MPI кожному процесу призначається блок пікселів numPixels/numProcs. Основна стратегія полягає в тому, що кожен вузол MPI незалежно призначає свій блок пікселів відповідним кластерам.

OpenMP дає більшу продуктивність, ніж MPI.

Незважаючи на те, що кожен процес матиме доступ до повного зображення HSI під час обробки, було прийнято рішення, щоб кожен піксель, який використовується для ініціалізації кластера, зчитувався з файлу даних тільки процесом, що володіє блоком MPI, що відповідає цьому пікселю. Фактично, це було б необхідно в ситуації, коли сам файл даних знаходиться в розподіленому архіві, тому різні вузли зберігання / обробки зберігають різні фрагменти файлу. Середнє значення ініціалізації кластера пікселями було розподілено MPI вузлами, як показано в лістингу нижче:

```
unsigned long pixelID;
```

```
int pixelRank; // <-- rank of node responsible for the pixel
for (i = 0; i<numClusters; i++)
{
pixelID = (i * numRows / numClusters) * numCols + (i * numCols / numClusters);
pixelRank = pixelID * numProcs/numPixels;
if (rank == pixelRank) { // I own this pixel so copy it in my cluster centers buffer.
copy(image + (pixelID – pStart) * numBands, centers[i], numBands);
}
MPI::COMM_WORLD.Bcast(centers[i], numBands, MPI::FLOAT, pixelRank);}
```

Оскільки кожен вузол MPI може визначити, який вузол є власником будь–якого пікселя, тільки власник блоку, який містить певний піксель ініціалізації, копіює його у буфер центрів. Потім всі вузли виконують ідентичну операцію Bcast, щоб скопіювати її у відповідне місце у своєму локальному буфері центрів.

Після ініціалізації кластерних центрів кожен вузол виконує початкове присвоєння всіх своїх пікселів відповідним кластерам:

```
NumChanged = assignPixels (image, clusterMap, centers, numClusters, myNumPixels, numBands);
```

Функція assignPixels виконує таку ж функцію, як і цикл виділення пікселів у послідовному коді.

Кожен крайній цикл алгоритму *k*–середніх складається з оновлення кластерних засобів (центрів), за яким слідує повторне призначення всіх пікселів у зрушені кластери. Для оновлення центрів кластера потрібен міжпроцесний зв'язок, тому що середня позиція кластера вимагає підсумовування пікселів, призначених даному кластеру у всіх вузлах MPI

Кінцевим етапом процесу нейромережевої кластеризації є об'єднання часткових кластерних карток в одну кластерну картку для всього зображення. Це виконується нейронною мережею прямого поширення.

Об'єднання кластерних карток у єдину картку відбувається за допомогою такого набору команд:

```
if (rank == 0)
{
for (i = 1; i <(unsigned int) numProcs; i++)
{
unsigned longiStart = i*numPixels/numProcs;
unsigned longiStop = (i + 1) * numPixels/numProcs;
if (i == (unsigned long) numProcs – 1)
iStop = numPixels;
MPI::COMM_WORLD.Irecv(clusterMap + iStart, iStop – iStart,
MPI::SHORT, i, i);
}
}
else
{
MPI::COMM_WORLD.Send(clusterMap, myNumPixels, MPI::SHORT, 0,
rank);
}
MPI::COMM_WORLD.Barrier().
```

Виконаємо тестування запропонованого рішення за допомогою середовища Scilab.

Для тестування були взяті гіперспектральні знімки. Це потокові лінії, зібрані спектрофотометром Airborne Visible & InfraRed Imaging Spectrum (AVIRIS) над Cuprite, NV. 1.

Сенсорні зображення AVIRIS 224 – спектральні смуги, що охоплюють 0,4–2,5 мкм. Сегменти були рекомбіновані у цілу потокову лінію з 2206 рядками і 614 стовпцями.

Зразкові пікселі зображення зображені на рисунку 3.3. Зображення містить атмосферно скориговані значення рефлексії (помножені на 10000). Глибинні смуги поглинання атмосфери задаються близько 14 і 18 мкм, а також кілька насичених смуг поблизу країв смуг поглинання. Постійне нульове або насичене значення по всьому зображенню не надає несприятливого впливу на кластеризацію, скоріше це призводить до зменшення розмірності даних на одиницю.

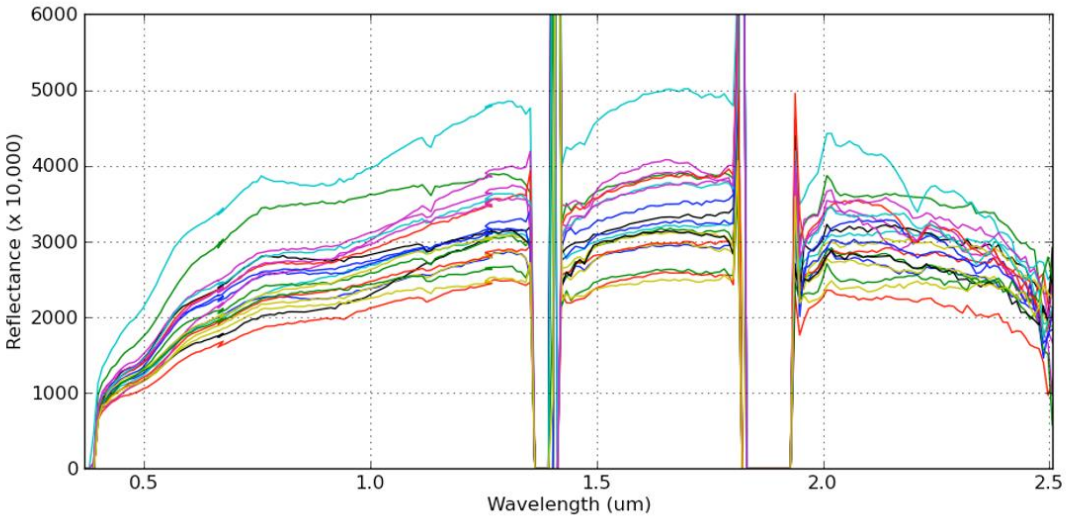

Рисунок 3.3 – Розподіл пікселів зображення

Результати кластеризації з прикладу вихідного зображення можна побачити на рисунку 3.4. На ньому показані канали RGB для вихідного зображення, картка кластера після однієї ітерації та карта кластера після 10 ітерацій. Для тестового зображення поява карти кластера після 1 та 10 ітерацій не радикально відрізняється.

Це в першу чергу пов'язано з тим, що область, покрита зображенням, складається з відносно невеликої кількості подібних по спектру порід і мінералів.

На рисунку 3.5 показана кількість пікселів, призначених новому кластеру, залежно від номера ітерації *k*–засобу.

Конвергенція кластерів очевидна зі зменшення кількості мігруючих пікселів.

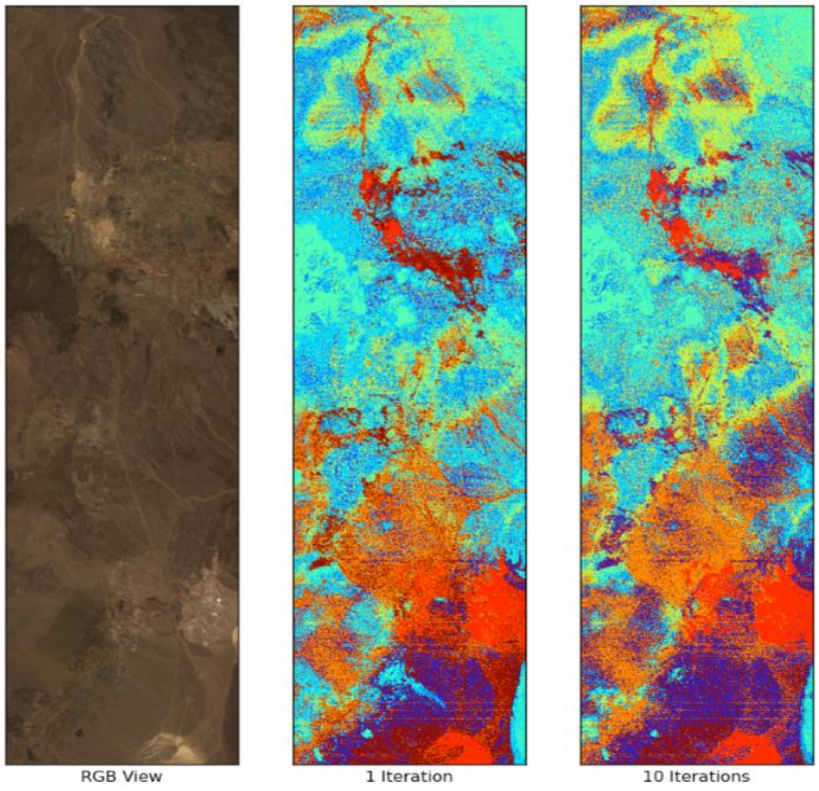

Рисунок 3.4 – Результати кластеризації

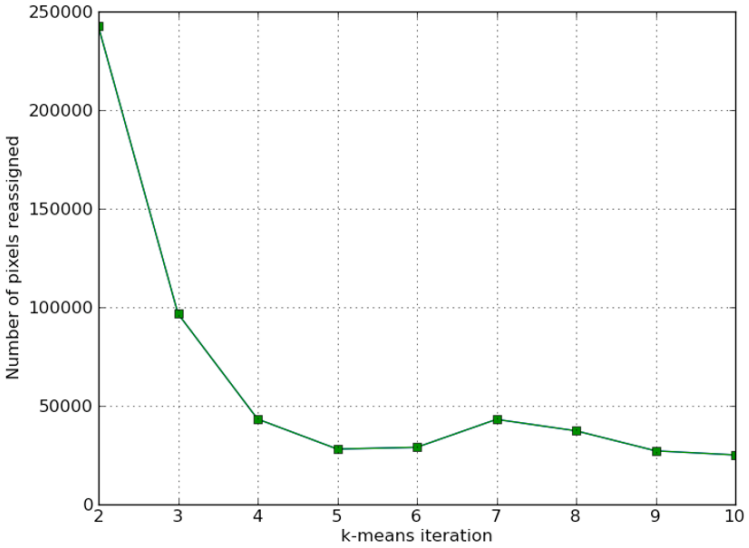

Рисунок 3.5 – Кількість пікселів, призначених кластеру, залежно від ітерації алгоритму

Тестові параметри представлені у вигляді таблиці 3.2.

Таблиця 3.2 – Тестові параметри

Параметри	Значення
Тип програми	Serial, OpenMP, MPI, HybridMPI/OpenMP
Розмір зображення	512x614(140MB),2206x614(607MB)
Кількість кластерів (k)	10, 20

На рисунку 3.6 показано час виконання програми Scilab як функцію кількості потоків (верхній графік) та прискорення (нижній графік).

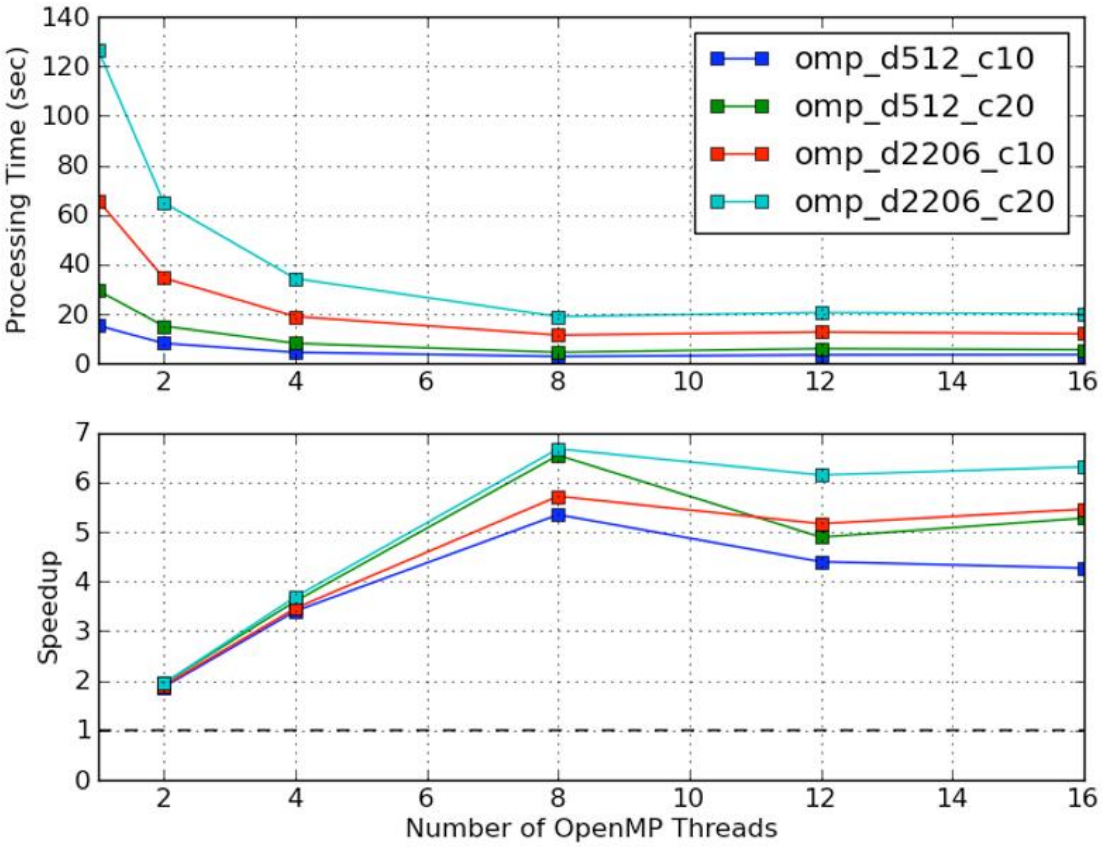

Рисунок 3.6 – Тестування алгоритму

Верхній графік демонструє залежність часу виконання кількості потоків; нижній–прискорення зі зростанням кількості потоків (експеримент 1).

Варто звернути увагу, що точки даних для кількості потоків, що дорівнює одиниці, відносяться до часу виконання для послідовної програми. Прискорення збільшується із збільшенням кількості потоків до 8 потоків, після чого прискорення зменшується.

Для найбільш інтенсивного обчислювального випадку (зображення 2206x614 з 20 кластерами) прискорення масштабується майже лінійно до 16 вузлів, що забезпечує прискорення значення на 14 вузлах. Для інших випадків прискорення відхиляється від лінійності при менших числах процесорів і зображення 512x614 з 10 вузлами прискорення фактично починає знижуватися з більш ніж 6 MPI–процесами. Це пов’язане з тим, що процеси витрачають більшу частину часу для виконання міжпроцесного зв'язку, ніж у інших випадках (рисунок 3.7).

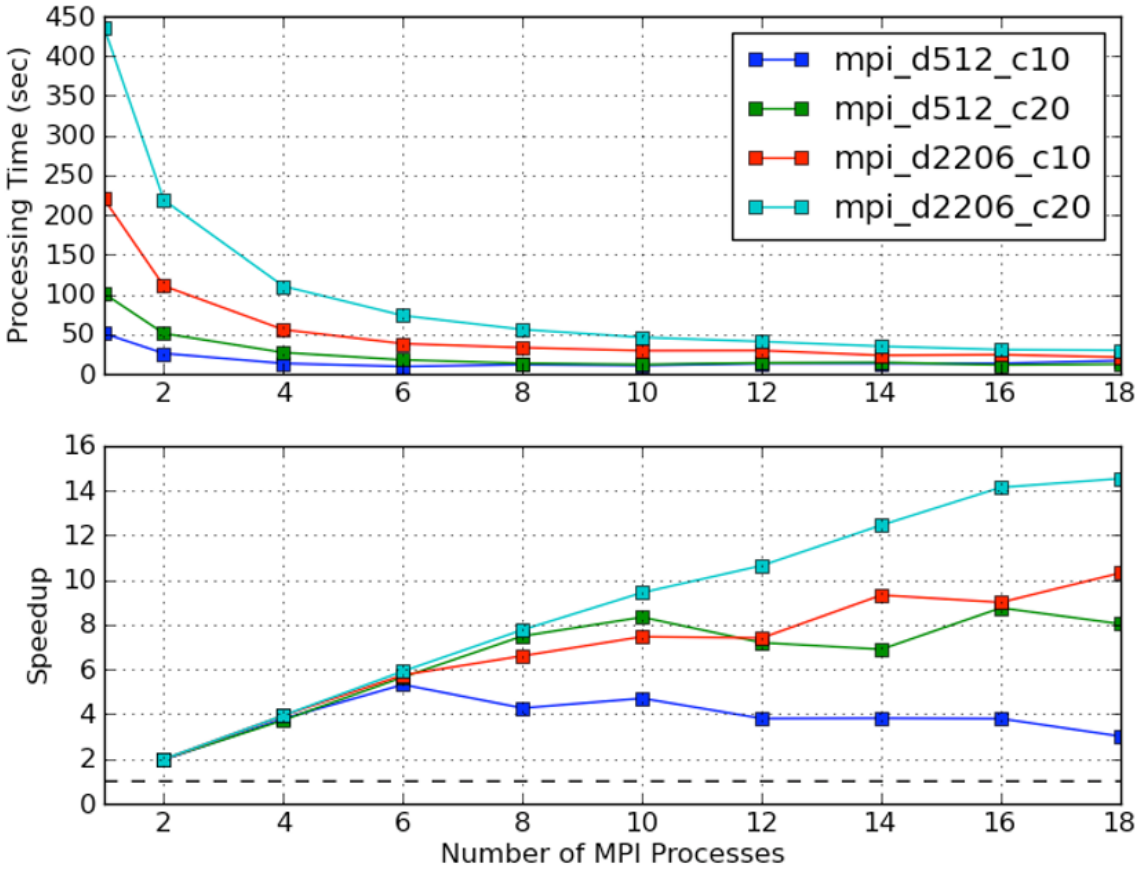

Рисунок 3.7 – Тестування алгоритму (експеримент 2):

На рисунку 3.8 представлено результати третього експерименту: червоний графік демонструє залежність часу виконання кількості потоків; синій–прискорення зі зростанням кількості потоків.

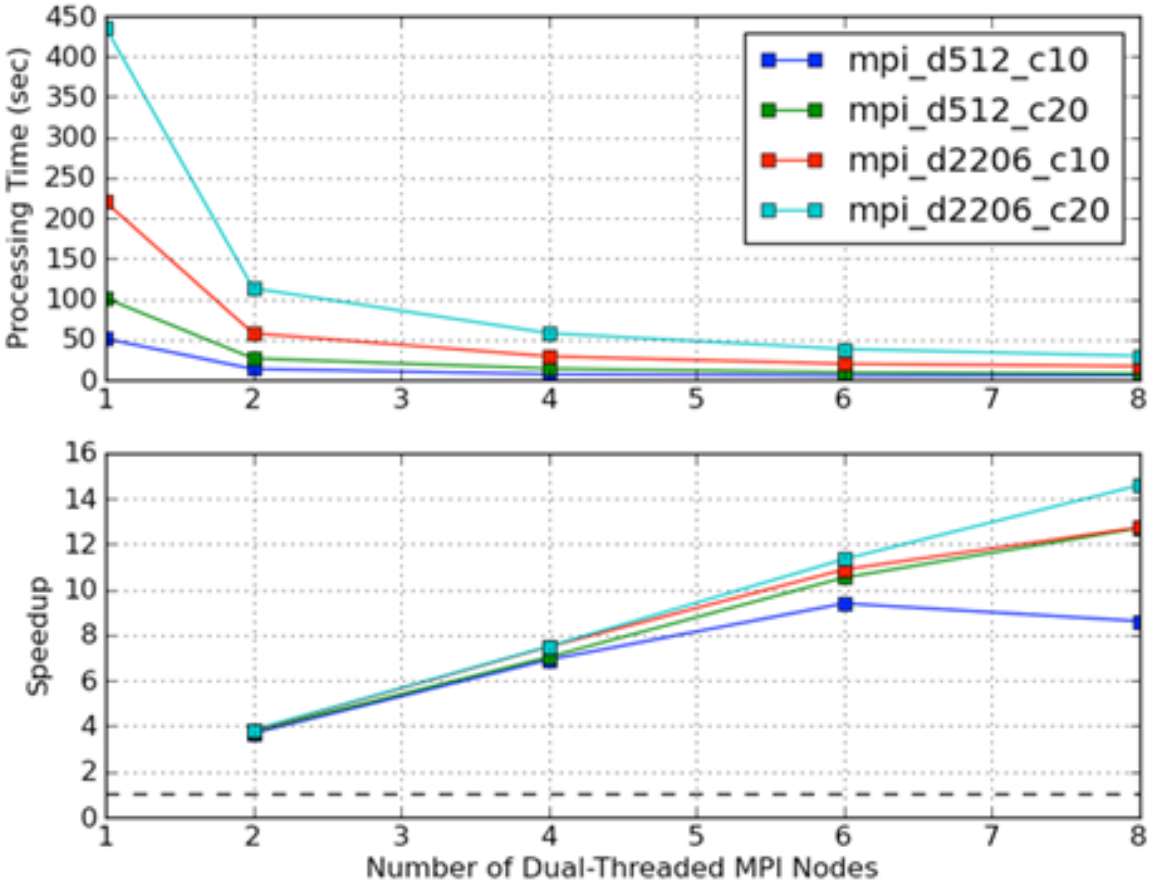

Рисунок 3.8 – Результати третього експерименту

Розроблений алгоритм нейромережевої кластеризації має наступну структуру (рис. 3.9).

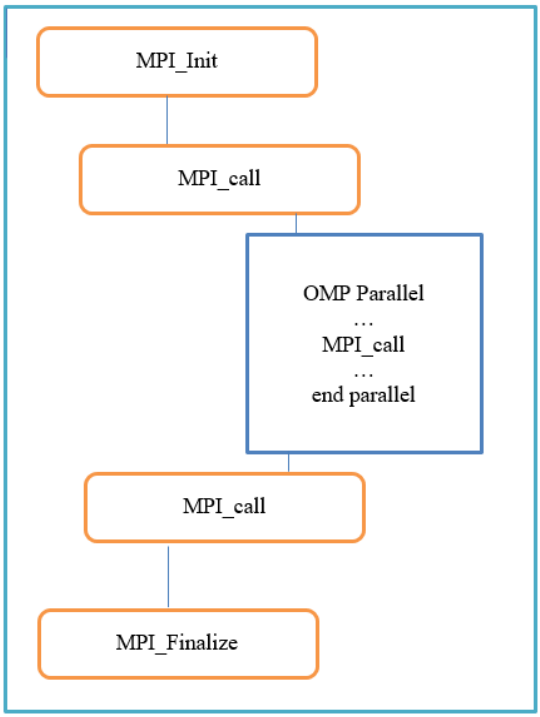

Рисунок 3.9 – Структура алгоритму нейромережевої кластеризації

На рисунках 3.10 та 3.11 порівнюються характеристики трьох експериментів для найбільших та найменш обчислювальних інтенсивних випадків, відповідно.

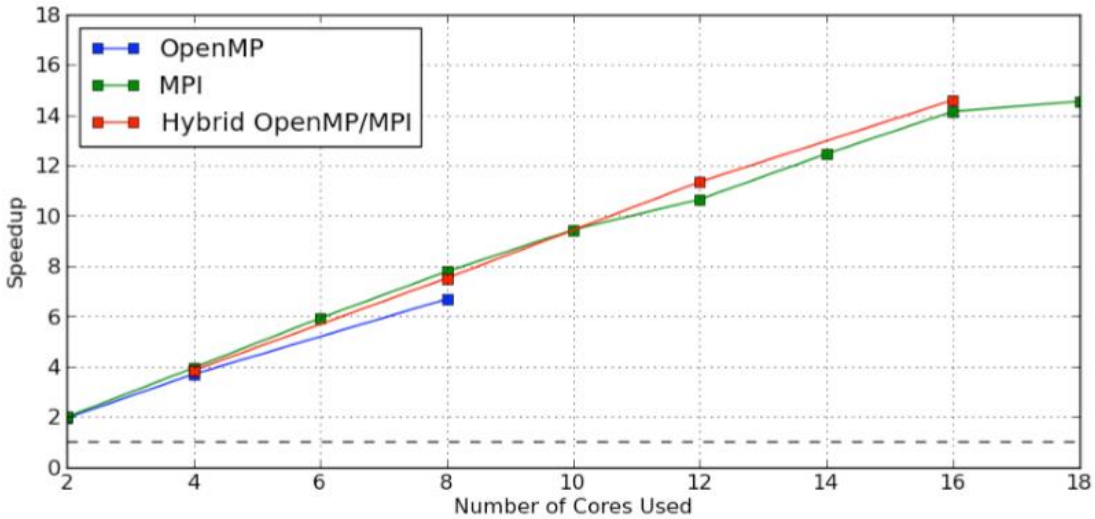

Рисунок 3.10 – Прискорення паралельних алгоритмів зображення 2206x614 з 20 кластерами

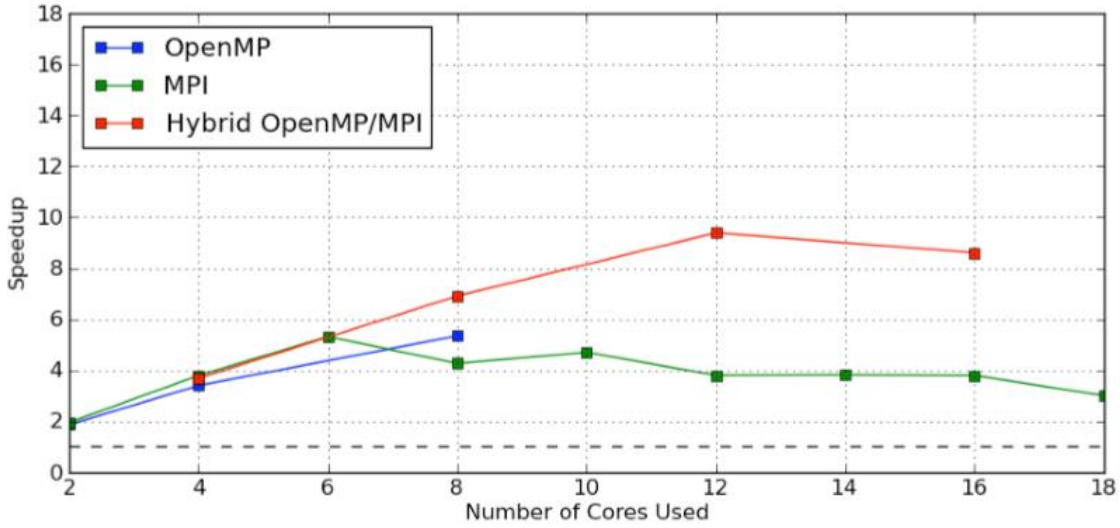

Рисунок 3.11 – Прискорення паралельних алгоритмів зображення 512x614 з 10 кластерами

Таким чином, значення 12 для MPI відноситься до 12 процесів MPI, що виконуються на 9 вузлах CDS, тоді як значення 12 для гібридного OpenMP / MPI відноситься до 6 двопотокових MPI–процесів, що виконуються на 6 вузлах обчислювального кластера.

Варто зауважити, що на рисунку 3.11 прискорення для MPI починає зменшуватися з більш ніж 6 ядрами, а прискорення для гібридного OpenMP/MPI починає зменшуватися з більш ніж 12 ядрами. Обидва ці випадки відповідають виконанню більше 6 процесів MPI, що узгоджується з ситуацією.

Таким чином, розроблений метод нейромережевої кластеризації працює коректно, що говорить про можливість застосування нейронної мережі прямого поширення та середовища Scilab для вирішення завдань кластеризації.

ВИСНОВКИ

В результаті написання магістерської дисертації було проведене дослідження методів нейромережевої кластеризації даних на основі нейронних мереж прямого поширення.

На основі огляду існуючих алгоритмів і існуючих рішень з застосування нейронних мереж для кластеризації даних було встановлено, що актуально застосовувати алгоритм кластеризації, який поєднує в собі метод *k*–середніх і багатошаровий персептрон.

В експериментальній частині було реалізовано паралельну версію алгоритму *k*–середніх, проведено дослідження показників ефективності паралельного алгоритму на кластері *Jet*.

Проведені тести наочно демонструють, що значних прискорень можна досягти за рахунок використання паралельної реалізації алгоритму *k*–середніх для гіперспектральних зображень. Для великого файлу (600 МБ) найбільше прискорення досягнуто під час запуску паралельної програми на 14 обчислювальних вузлах кластера *Jet*. Для ближчого до типового розміру (140 МБ) найбільше прискорення отримано під час запуску паралельної програми на 9 обчислювальних вузлах кластера *Jet*.

Експериментально встановлено, що застосовування даного підходу для нейромережевої кластеризації є ефективним.

В результаті написання магістерської дисертації було вирішено наступні завдання:

1. Здійснено огляд літератури з теми дослідження. Розглянуто задачі нейромережевої кластеризації даних та існуючі підходи.

2. Досліджено особливості застосування нейронних мереж для кластеризації даних.

3. Наведено приклад нейромережевої кластеризації даних на основі нейронних мереж прямого поширення.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ситник В. Ф., Краснюк М. Т. Інтелектуальний аналіз даних: Навч. посібник. Київ, КНЕУ, 2007. 376 с.

2. Данильченко О.М., Данильченко А.О. Аналіз даних: Навч. посібник. Житомир, ЖДТУ, 2009. 405 с.

3. Черняк О.І. Інтелектуальний аналіз даних: підручник. Київ, Знання, 2014. 599с.

4. Стандартні методи кластеризації даних. URL: http://csc.knu.ua/media/study/asp/mod_probl_inf_tech_sys_analysis_ivohin/lecture/lec2.pdf (дата звернення: 10.11.2022).

5. Md Rezaul Karim, Oya Beyan, Achille Zappa, Ivan G Costa, Dietrich Rebholz–Schuhmann, Michael Cochez, Stefan Decker, Deep learning–based clustering approaches for bioinformatics, Briefings in Bioinformatics, Volume 22, Issue 1, January 2021. pp. 393–415.

6. Oyelade J, Isewon I, Funke, et al. Clustering algorithms: their application to gene expression data. Bioinform Biol Insights, 2016. 25 p.

7. Min E, Guo X, Qiang, et al. A survey of clustering with deep learning: from the perspective of network architecture. IEEE Access, 2018. 45 p.

8. Zhao L, Zaki MJ. Tricluster: an effective algorithm for mining coherent clusters in 3D microarray data. In: Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data. Baltimore, MA: ACM, 2005. pp. 694–705.

9. Jaskowiak PA, Costa IG, Campello RJGB. Clustering of rna–seq samples: comparison study on cancer data. Methods 2018. 46 p.

10. Karmakar B, Das S, Sohom, et al. Tight clustering for large datasets with an application to gene expression data. Sci Rep, 2019. 36 p.

11. Yang J, Parikh D, Batra D. Joint unsupervised learning of deep representations and image clusters. In: Proceedings of the IEEE Conference on

Computer Vision and Pattern Recognition, 2016, 5147–56. Caesars Palace Las Vegas Hotel & Casino, Las Vegas, Nevada, USA, Jun 26–Jul 1, 2016. 38 p.

12. Ніколенко С. Глибоке навчання, Київ, КНЕУ, 2019. 412 с.

13. Хайкін С. Нейронні мережі: повний курс. Запоріжжя, ЗНУ, 2019. 868 с.

14. Малов Р. Нейронні мережі: Короткий довідник. Київ, КНЕУ, 2017. 288 с.

15. Субботін С.О. Нейронні мережі: теорія та практика. Житомир: О.О. Євенок, 2020. 184 с.

16. C. Chandhok. A Novel Approach до Image Segmentation using Artificial Neural Networks and K–Means Clustering. International Journal of Engineering Research and Applications (IJERA), 2 (3), 2012. pp. 274-279.

17. Z. Qing, Y. Guanhui, G. Tingling, Z. Hong, L. Junxiao. Fabric Defect Segmentation Based on Region Growing PCNN Model. Computer application and software, 28(11), 2011. 46 p.

18. O. Senyukova, A. Lukin, D. Vetrov. Automated Atlas–Based Segmentation of NISSL–Stained Mouse Brain Sections Using Supervised Learning. Programming and Computer Software, 2011. pp. 245-251.

19. V.B. Nemirovsky, A.K. Stoyanov. Multi–step segmentation of images by means of a recurrent neural network. Proc. of the 7th Intern. forum on strategic technology (IFOST–2012), Tomsk, 1, 2012. pp. 557–560.

20. Функції в Scilab. URL: https://ppt–online.org/41640 (дата звернення: 18.11.2022).

21. Руденко О. Г. Искусственные нейронные сети: архитектуры, обучение, применения / О. Г. Руденко, Е. В. Бодянский. – Харьков: ТЕЛЕТЕХ, 2004. – 370 с.