Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Робастна інкрементна нейро-фаззі мережа
в задачах опрацювання потоків даних
(тема)

Виконав:
студент 2 курсу, групи СШІм-19-2
Рєпін Д.О.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту
(повна назва спеціалізації)

Керівник проф. каф. ШІ Бодянський Є.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ В.О. Філатов
(підпис) (прізвище, ініціали)

2021 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерних наук
(повна назва)

Кафедра Штучного інтелекту
(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системи штучного інтелекту (СШІ)
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«_____» _____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Рєпіну Денису Олеговичу
(прізвище, ім'я, по батькові)

1. Тема роботи Робастна інкрементна нейро-фаззі мережа в задачах опрацювання потоків даних
затверджена наказом університету від __ _____ 20 __ р. № _____
2. Термін подання студентом роботи до екзаменаційної комісії _____ _____ 20 ___ р.
3. Вихідні дані до роботи Науково-технічні публікації, дані Інтернет-джерел та відомих наукових проектів щодо розробки систем нейро-фаззі кластерування та робастних систем в задачах динамічного аналізу даних

4. Перелік питань, що потрібно опрацювати в роботі
1) Аналіз предметної галузі та формалізація задачі
2) Алгоритм кластеризації Fuzzy C-means
3) Самоорганізована інкрементна нейронна мережа
4) Робастна інкрементна нейро-фаззі мережа
5) Імітаційне моделювання і перевірка теоретичних досліджень

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) Рисунок 1 – Схема навчання з учителем, Рисунок 2 – Схема навчання без вчителя, Рисунок 3 – Приклади функцій активації, Рисунок 4 – Сигмоїдальна функція активації, Рисунок 5 – Схема SOIN, Рисунок 6 – Блок-схема алгоритму SOINN, Рисунок 7 – Схематичне зображення методу Z-Score, Рисунок 8 – Схема методу інтерквартильного діапазону, Рисунок 9 – «Боксплот», Рисунок 10 – Штучносгенерований набір даних (3 кластери), Рисунок 11 – Фінальна кластеризація даних розробленою моделлю робастної інкрементної нейро-фаззі мережі, Рисунок 12 – Вихідні дані вибірка «Іриси Фішера», Рисунок 13 – Фінальна кластеризація даних розробленою моделлю робастної інкрементної нейро-фаззі мережі вибірки «Іриси Фішера»

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата
Аналіз предметної галузі	проф. Бодянський Є.В.		26.03.2021
Формування вимог до додатку	проф. Бодянський Є.В.		13.04.2021
Розробка додатку	проф. Бодянський Є.В.		22.04.2021
Аналіз готового продукту	проф. Бодянський Є.В.		23.04.2021

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Аналіз предметної галузі	26.03.2021	виконано
2	Дослідження існуючих методів кластерування	13.04.2021	виконано
3	Розробка робастної інкрементальної нейро-фаззі мережі	14.04.2021	виконано
4	Створення імітаційної моделі	15.04.2021	виконано
5	Тестування і аналіз отриманих результатів	23.04.2021	виконано
6	Оформлення пояснювальної записки	24.04.2021	виконано
7	Попередній захист	14.05.2021	виконано
8	Захист перед ЕК	18.05.2021	

Дата видачі завдання _____ _____ 20 ___ р.

Студент _____
(підпис)

Керівник роботи _____ _____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка складається з 70 с., 13 рисунків, 3 таблиці, 66 формул, 24 джерел.

ІНКРЕМЕНТНА САМООРГАНІЗОВНА МАПА, ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ДАНИХ, КЛАСТЕРУВАННЯ, НЕЙРОННА МЕРЕЖА, НЕЧІТКА СИСТЕМА, ПОСЛІДОВНЕ НАВЧАННЯ, ШТУЧНИЙ ІНТЕЛЕКТ

Об'єктом дослідження є процес опрацювання потоків даних що надходять послідовно в режимі реального часу за допомогою робастної інкрементної нейро-фаззі мережі.

Предметом дослідження є методи інтелектуального аналізу даних при вирішенні задачі робастної кластеризації.

Метою кваліфікаційної магістерської роботі є розробка архітектури робастної інкрементної нейро-фаззі мережі для вирішення задачі кластеризації в умовах класів, що перетинаються.

Методи дослідження базуються на теорії обчислювального інтелекту, а саме на методах теорії штучних нейронних мереж для побудови архітектури робастної самоорганізованої мапи, що складаються з нечіткого шару виводу і дозволяють проводити нечітке кластерування в послідовному режимі. Імітаційне моделювання застосовується для перевірки якості кластерування з використанням синтезованої архітектури робастної інкрементної нейро-фаззі мережі.

В кваліфікаційній роботі розглядається задача робастного нечіткого послідовного кластерування потоків даних в умовах класів, що перетинаються.

РЕФЕРАТ

Пояснительная записка состоит из 70 с., 13 рисунков, 3 таблиц, 66 формул, 24 источников.

ИНКРЕМЕНТНАЯ САМООРГАНИЗУЮЩАЯСЯ КАРТА, ИНТЕЛЛЕКТУАЛЬНННЫЙ АНАЛИЗ ДАННЫХ, ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ, КЛАСТЕРИЗАЦИЯ, НЕЙРОННЫЕ СЕТИ, НЕЧЁТКАЯ СИСТЕМА, ПОСЛЕДОВАТЕЛЬНОЕ ОБУЧЕНИЕ

Объектом исследования является процесс обработки потоков данных поступающих последовательно в режиме реального времени с помощью робастной инкрементной нейро-фази сети.
Предметом исследования являются методы интеллектуального анализа данных при решении задачи робастной кластеризации.

Целью квалификационной магистерской работы является разработка архитектуры робастной инкрементной нейро-фази сети для решения задачи кластеризации в условиях классов, которые пересекаются.

Методы исследования базируются на теории вычислительного интеллекта, а именно на методах теории искусственных нейронных сетей для построения архитектуры робастных самоорганизующихся карт, состоящие из нечеткого слоя вывода, и позволяющих проводить нечеткую кластеризацию в последовательном режиме. Имитационное моделирование применяется для проверки качества кластеризации с использованием синтезированной архитектуры робастной инкрементной нейро-фази сети.

В квалификационной работе рассматривается задача робастного нечеткого последовательного кластерування потоков данных в условиях классов пересекаются.

ABSTRACT

Explanatory note: 70 pages, 13 figures, 3 tables, 66 formulas, 24 sources.

INCREMENTAL SELF-ORGANIZING MAP, INTELLIGENT DATA ANALYSIS, ARTIFICIAL INTELLIGENCE, CLUSTERING, FUZZY SYSTEM, NEURAL NETWORKS, SEQUENTIAL TRAINING

The object of the research is the process of processing data streams arriving sequentially in real time using a robust incremental neural phase of the network. The subject of the research is the methods of data mining when solving the problem of robust clustering.

The purpose of the qualifying master's work is to develop an architecture of a robust incremental neural-phase network for solving the clustering problem in the conditions of classes that overlap.

The research methods are based on the theory of computational intelligence, namely, on the methods of the theory of artificial neural networks for building the architecture of robust self-organizing maps, consisting of a fuzzy inference layer, and allowing fuzzy clustering in a sequential mode. Simulation modeling is used to check the quality of clustering using the synthesized architecture of a robust incremental neural phase network.

In the qualification work, the problem of a robust fuzzy sequential cluster is considered for data streams in terms of classes that intersect.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШНМ – штучні нейронні мережі;

FCM – Fuzzy C-means – нечіткий с-середніх;

SOINN – Self-Organizing Incremental Neural Network – самоорганізована інкрементна нейронна мережа;

SOM – Kohonen’s Self-Organizing Maps – самоорганізовані мапи Т. Кохонена.

ВСТУП

На сьогоднішній день стрімкий розвиток інформаційних систем запроваджений у багатьох сферах допомагає людям робити те, що раніше було складно уявити. В деяких областях системи роблять роботу на порядок краще людей. Системи, які базуються на штучних мережах можуть робити переклади різноманітних текстів, вирішувати задачі обробки потоків даних, проводити оптимізацію робіт. Розроблено багато систем які виконують обробку зображень та потокового відео, допомагають опрацьовувати фінансові справі такі як прогнозування курсів, пошук інформації, планування та інше.

Сьогодні існує велика кількість напрямків в науці стосовно штучного інтелекту. Сюди входить вирішення багатьох задач як класифікація, кластерування, прогнозування, аналіз сигналів різного походження і так далі. Однак найбільшого поширення набула задача кластерування, коли потрібно розбити вхідний набір даних на схожі між собою групи по деяким ознакам. Зазвичай самі ефективні алгоритми створені на базі штучні нейронних мереж.

Сьогодні нараховується багата кількість методів кластерування, обробка яких проходить в режимі самонавчання або без вчителя. Вони відрізняються вихідними результатами та математичною моделлю, яка використовується у цих алгоритмах. Самий простий, а тому і самий поширений метод кластерування вважається метод К-середніх. В цьому алгоритмі головній недолік полягає в потребі попереднього вибору кількості кластерів. Ще одна проблема полягає в тому, що більшість таких алгоритмів очікують, що кластери будуть у форми сфери та будуть лінійно роздільні. Але нажаль в реальних умовах вхідні дані можуть буті забрудненні, а елемент одного кластеру також може належати іншому.

Обробка даних може поділятися за різним ознаками. Одна з таких ознак – режим обробки даних. Вин може бути онлайновий, тобто

послідовний, або пакетний. Як можна зрозуміти, в онлайновому методі дані поступають поступово, що дозволяє отримувати нову інформацію для обробки, прі цьому тримаючи попередню отриману інформацію. В такому підході нема необхідності повторно навчати мережу, коли надходять. В той самий час також є пакетний режим. Як перевагу можна відмітити більш велику точність розбиття на кластери, але це є причиною недоліка. В пакетному режимі обробка вхідних даних потребує набагато більшого часу, тому що потрібні одразу усі дані. С цього можемо зробити висновок, що якщо нам потрібно додати нову партію даних, потрібно буде навчати мережу заново.

Наступна з ознак кластерування – чітке та нечітке. В моїй роботі буде розглянуте нечітке кластерування. Нечітке кластерування має велику перевагу тому плані, що можна працювати з різними невизначеностями, які присутні в реальних даних. Можна сказати, що нечітке кластерування є продуктом використання теорії нечітких множин на кластеризації. Це встає у нагоді, коли одним елемент може належати декільком кластерам за різними ознаками.

На останок слід відзначити, що реальні дані можуть буту забруднені, тобто мають викиди. Для цього існують робастні системи, які вміють подавляти ці викиди – демпфірувати. Робастні системи відомі, але онлайнові робастні системи – інкрементальні системи – невідомі.

В цій роботі поставлена ціль, щоб зробити систему нечіткої кластеризації, працюючий в потоці, тобто онлайн, та яка не боїться забруднених даних

Таким чином, актуальність полягає в тому, що пропонується алгоритм, який може використовуватися як робастна інкрементальна система, та в той же час відносно швидко опрацьовувати нові поступаючи дані.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ФОРМАЛІЗАЦІЯ ЗАДАЧІ

Кінець двадцятого століття характеризується інтенсивним вивченням штучних нейронних мереж (ШНМ). Поява ШНМ пов'язана з розумінням того, що мозок живого організму працює інакше, ніж комп'ютер. Людський мозок – це дуже складна нелінійна, паралельно інформаційно-керуюча система, яка здатна до мислення, накопичення і відновлення інформації, вирішення проблем. Ця система складається з великої кількості нервових клітин, або нейронів, що представляють собою прості елементи обробки сигналів, які отримують і комбінують інформацію від інших нейронів через входи – дендрити. Якщо об'єднаний з усіх входів сигнал досить сильний, нейрон переходить в збуджений стан («збуджується»), генеруючи сигнал на виході – аксоні, який пов'язаний з дендритами безлічі інших нейронів. Кожен сигнал, що надходить в нейрон, проходить через синаптичне з'єднання, де в результаті електрохімічних процесів потік електричних зарядів або прискорюється, або сповільнюється. Саме зміни провідності синаптичних зв'язків лежать в основі процесів навчання і запам'ятовування інформації. Але якщо початково дослідження базувалися на використанні моделей біологічних нейронів У. Маккалоха, У. Питтса и Ф. Розенблатта [4, 5], то сьогодні нейромережеві технології збагатилися новими моделями, заснованими на законах фізики, генетики, математики.

Поняття «навчання» є ключовим в теорії ШНМ. Тип і характер навчання визначаються на основі розв'язуваної задачі, властивостей даних, які надходять на вхід мережі із зовнішнього середовища у вигляді навчальної вибірки образів або прикладів. Відомо дві основні парадигми навчання: з учителем і без вчителя. Навчання з учителем є більш простим і очевидним. Завідомо відома інформація про зовнішнє середовище, яка задана у вигляді послідовності або пакета вхідних векторів x. Також відомий навчальний сигнал d. Реакція ненавченої мережі, відрізняється від «правильної» реакції вчителя, в результаті чого виникає помилка.

У процесі навчання необхідно так налаштувати параметри ШНМ, щоб деяка скалярна функція помилки(критерій якості) досягла свого мінімального значення. Навченою вважається мережа, яка в деякому, як правило, статистичному сенсі повторює реакцію вчителя. Оскільки інформація про зовнішнє середовище зазвичай має нестаціонарний характер, процес навчання йде безперервно, для чого використовуються ті чи інші рекурентні процедури. Відповідно оцінити обраний алгоритм і обчислити критерій якості неможливо, тому крім навчання мережі, необхідно визначати параметр по якому буде оцінено правило навчання. На рисунку 1.1 запропонована схема навчання з учителем.

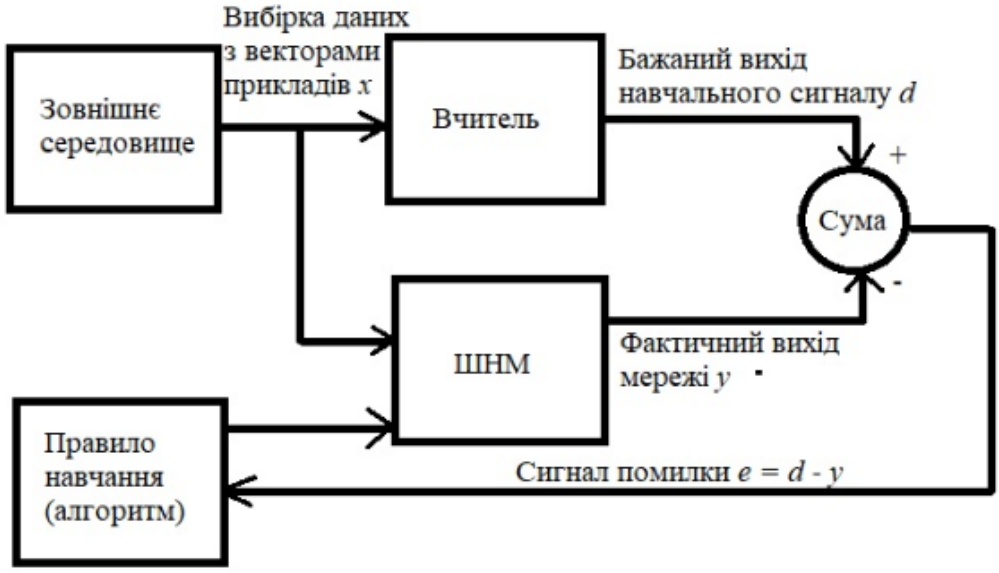


Рисунок 1.1 – Схема навчання з учителем

Парадигмою навчання без учителя, або самонавчання, називають навчання, коли правильна реакція на сигнали зовнішнього середовища невідома. Процес самонавчання схематично представлений на рисунку 1.2.

Мережі, які реалізують парадигму самонавчання, призначені, як правило, для аналізу внутрішньої латентної структури вхідної інформації і вирішують завдання автоматичної класифікації, кластеризації, факторного

аналізу, компресії даних. Ці задачі є більш складними виходячи з математичної постановки, але вирішують безліч реальних проблем.

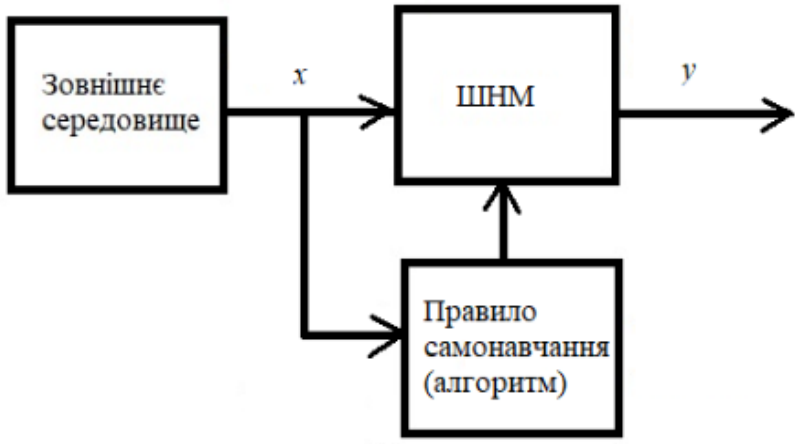


Рисунок 1.2 – Схема навчання без вчителя

Своєрідним компромісом між двома цими парадигмами є навчання з підкріпленням, при якому доступна лише непряма інформації про правильну реакції на вхідний сигнал. Нейронна мережа виробляє відображення вхідної інформації у вихідний вектор, проте, оскільки бажаний навчальний сигнал в явному вигляді не заданий, неможливо отримати помилку, на підставі якої відбувається навчання. Передбачається, що є деякі апріорні знання, що дозволяють зв'язати евристичний сигнал підкріплення з бажаним виходом за допомогою деякої функції. Зазвичай ця функція враховує зв'язок вихідних сигналів мережі з подіями у зовнішньому середовищі, для чого в схему навчання вводиться додатковий блок – «критик», що відображає поведінку мережі. Далі обчислюється евристична помилка, на основі якої і реалізується процес навчання. Досить широке поширення набула також парадигма змішаного навчання, коли частина параметрів мережі налаштовується за допомогою навчання з учителем, а інша частина або архітектура в цілому - за допомогою самонавчання. Цей

підхід набув найбільшого поширення при навчанні радіально-базисних ШНМ.

З введеними парадигмами тісно пов'язані правила навчання, що лежать в основі конкретних алгоритмів. У конкурентному навчанні, можуть бути реалізовані всі описані парадигми, при цьому його відмінною рисою є процес «змагання» нейронів вихідного шару. Існує два принципи змагання згідно з Кохоненом «переможець отримує все» і «переможець отримує більше» [3]. У першому випадку збуджується тільки один вихідний нейрон – «переможець». У другому відбувається виправлення всіх синаптичних ваг і переможець отримає більше, ніж всі інші. Найбільш яскравими прикладами мереж, що використовують це правило, є мережі адаптивного резонансу і самоорганізовані мапи (SOM).

1.1 Постановка задачі кластерування

Однією з задач навчання без вчителя є кластерування масивів даних. У дипломній роботі запропоновано робастну самонавчанну нейро-фаззі систему для вирішення цієї проблеми.

Кластерування – задача угруповання набору об'єктів таким чином, що об'єкти в одній і тій же групі (звані кластером) більш схожі (в тому чи іншому сенсі) один з одним, ніж з іншими групами (кластерами). Можна сказати, що задача полягає в пошуку взаємозв'язків в нерозміченому наборі даних.

Мета кластерування – пошук існуючих структур. Кластерування є описовою процедурою, вона не робить ніяких статистичних висновків, але дає можливість провести розвідувальний аналіз і вивчити структуру даних.

Поняття кластер визначено неоднозначно, але можна охарактеризувати як групу об’єктів, що мають спільні властивості. Характеристиками кластера є дві ознаки: внутрішня однорідність і зовнішня ізольованість. Тобто кожен кластер включає приклади схожі один до одного

за певним набором характеристик. Зовнішня ізольованість описує відміну набору характеристик від інших кластерів. Також вони можуть бути непересічними і пересічними [5].

Важливо відзначити, що кожен метод кластерування має свої переваги і недоліки, тому вибір алгоритму завжди залежить від характеристик і властивостей набору даних, які були виділені на етапі попереднього аналізу даних. Також важливо розуміти розподіл даних для того, щоб отримати на виході очікуваний результат.

Найбільш поширені властивості методів кластерування є:

– алгоритм кластерування, який базується на центроїдах. У цьому методі угруповання даних посилається на вектор значень. Кожен об'єкт є частиною кластера і відстань якого мінімальна до центроїда в порівнянні з іншими центроїдами кластерів. Кількість кластерів має бути попередньо визначено, і це найбільша проблема такого роду алгоритмів, бо не завжди є інформація про те, на скільки кластерів необхідно розбити набір даних. Також необхідно розробити метод розташування центроїдів. Ця методологія найбільш близька до предмета класифікації і широко використовується для задач оптимізації;

– на основі розподілу. Пов'язана зі заздалегідь визначеними статистичними моделями розподілення об'єктів, значення яких залежать від одного і того ж розподілу. Через свою випадкову природу генерації значень цей процес потребує добре визначеної і складної моделі для взаємодії з реальними даними. Однак ці процеси можуть забезпечити оптимальне рішення і розрахувати кореляції і залежності;

− на основі зв’язків. У цьому типі алгоритму кожен об'єкт пов'язаний з його сусідами, в залежності від ступеня цього відношення на відстані між ними. Виходячи з цього припущення, кластери створюються поруч з об'єктами і можуть бути описані як максимальна межа відстані. При такому зв'язку між членами ці кластери мають ієрархічні уявлення. Функція відстані залежить від фокуса аналізу;

− на основі щільності. Ці алгоритми створюють кластери відповідно з високою щільністю елементів набору даних в певному місці. Він об'єднує деяку відстань до стандартного рівня щільності для угруповання членів в кластерах. Такі процеси можуть мати меншу продуктивність при виявленні граничних областей групи.

Розглядуваний в дипломній роботі алгоритм кластерування відноситься до метода на основі зв’язків кожного об'єкта з його сусідом.

1.2 Постановка задачі нечіткого кластерування

Алгоритми нечіткої кластеризації дозволяють отримати нечіткий розподіл даних по кластерам. Тобто кожен з прикладів не входить однозначно в будь-який кластер, а належить всім кластерам з різними ступенями належності. Чим більше значення належності, тим с більшою вірогідністю приклад належить до кластеру.

Нечітке кластерування дає переваги в випадках, коли кластери знаходяться близько один до одного, і велике число точок знаходиться на лінії перетину кластерів. Однак ціною такої нечіткості служать великі обчислювальні витрати, в порівнянні з такими алгоритмами як с-середніх і k-середніх.

Наприклад, нечіткий с-середніх базується на мінімізації наступної цільової функції:

$$F\left(x(k), c_j, u_j(k)\right) = \sum_k^N \sum_j^m u_j^2(k) \left\| x(k) - c_j \right\|^2, \qquad (1.1)$$

де $u_j(k)$ – визначає належність елемента j вихідної множини векторів до $x(k)$ до кластеру j;

c_j - центр кластера, який розраховується як векторна норма.

При нечіткому розбитті ступінь належності об'єкта до кластеру приймає значення з інтервалу від нуля до одиниці $u_j(k) \in [0, 1]$, коли при

чіткому з двоелементної множини {0, 1}. Обмеження для матриці нечіткого розбиття записуються так:

$$\sum_{j=1}^{m} u_j(k)=1. \quad (1.2)$$

Нечітке розбиття дозволяє доволі легко вирішити проблему об'єктів, розташованих на кордоні двох кластерів і неможливо точно визначити до якого з кластерів наданий приклад відноситься. Наприклад, вибірка розбита на два класи і приклад, який розташований на їх кордоні належить з вірогідністю 0.4 до першого і 0.6 до другого кластера.

Але недолік нечіткого розбиття проявляється при роботі з об'єктами, віддаленими від центрів всіх кластерів. Віддалені об'єкти мають мало спільного з будь-яким з кластерів, тому інтуїтивно хочеться призначити для них малий ступень належності. Однак, за умовою (1.2) сума їх ступенів така ж, як і для об'єктів, близьких до центрів кластерів, тобто дорівнює одиниці.

1.3 Штучні нейронні мережі

Основною особливістю штучних нейронних мереж і, природно, що утворюють їх нейронів є здатність до навчання, в процесі якого синаптичні ваги налаштовуються за допомогою того чи іншого адаптивного алгоритму з метою найбільш ефективного вирішення поставленої проблеми.

Біологічний нейрон є особливою біологічною системою, призначеної для передачі і обробки інформації в живих організмах. Нейрон складається з тіла клітини, або соми, дендритів, аксона і синапсів.

Тіло клітини включає ядро, яке містить інформацію про спадкові властивості, і плазму, що володіє молекулярними засобами для створення необхідних нейрона матеріалів. Саме в сомі реалізуються основні функції, пов'язані з генетичними і метаболічними механізмами, необхідними для життєдіяльності, а також інформаційні функції.

Аксон – це нервове волокно, поєднане з сомой і є провідником вихідного сигналу. Аксон має розгалуження – волокна, звані терміналами аксона, за якими нервові імпульси проходять до інших нейронів. Дендрити – дуже розгалужене дерево волокон, з'єднаних з сомой. Дендрити отримують сигнал від інших аксони терміналів через спеціальні контакти - синапси.

Синапс є функціональним інтерфейсом між двома нейронами (аксонний термінал одного нейрона і дендрит іншого) і здатний підсилювати або придушувати сигнал подібно електронного підсилювача, визначаючи характер обробки інформації в сомі. Характеристики синапсів можуть перебудовуватися проходять через них сигналами так, що синапси навчаються в залежності від типів протікають в них процесів.

Інтенсивність вихідного сигналу залежить як від рівня вхідних сигналів, так і провідності відповідних синаптичних зв'язків. Інформація між нейронами передається за допомогою короткої серії імпульсів. Повідомлення передається за допомогою частотно-імпульсної модуляції, при цьому частота може змінюватися від одиниць до тисяч імпульсів в секунду. Як видно, за швидкістю обробки інформації нейрон істотно поступається сучасним електронним схемам, однак, як ми вже відзначали, висока швидкість обробки інформації в мозку забезпечується розпаралелюванням протікають в ньому процесів.

Штучна нейронна мережа – це машина, яка спроектована для моделювання функцій мозку і подібно до нього має багатошарову ієрархічну структуру і здатність до навчання [4]. Вузли штучної нейронної мережі, іменовані також штучними нейронами (нейронними клітинами, формальними нейронами) представляють собою елементарні процесори і є спрощеними моделями біологічних нейронів.

Функціонування ШНМ відображає роботу мозку в двох аспектах [4]:

– всі знання накопичуються навколишнього середовища в процесі навчання;

– навчання відбувається шляхом зміни (цілеспрямованого або випадкового) сили зв'язку між нейронами (синаптичних ваг) або топології (архітектури) мережі.

На кожен вхід нейрона подається сигнал, при цьому з кожним входом пов'язаний так звана синаптична вага. У тілі нейрона обчислюється примітивна функція суми добутку вхідних сигналів і синаптичних ваг, тобто фактично реалізується нелінійне відображення багатовимірного простору входів в скалярний вихід. Процедура, за допомогою якої відбувається навчання (настройка) окремого нейрона або нейромережі в цілому, називається алгоритмом навчання. У процесі навчання відбувається зміна (адаптація) синаптичних ваг, а можливо і топології ШНМ так, щоб вихідний сигнал був відповідним до деякого апріорі заданого критерію якості, що характеризує процес вирішення мережею конкретного завдання.

Важливою вимогою до процесу навчання є його адаптивність, яка забезпечує настройку ваг відповідно до характеристик навколишнього середовища. Якщо ці характеристики змінюються (при цьому не виключається можливість зміни і самої розв'язуваної задачі), мережа повинна в реальному часі «перенавчитися» відповідно до нових умов.
Більшість нейронних мереж утворено однотипними нейронами – це гомогенні (однорідні) мережі, хоча відомі гетерогенні мережі, сконструйовані з різних нейронів. Нейрони бувають аналоговими і бінарними, хоча цей поділ чисто умовно, оскільки один і той же формальний нейрон може функціонувати як в аналоговому, так і в цифровому режимах.

Функціональні характеристики окремих нейронів і мереж в цілому визначаються видом використовуваних активаційних функцій. Так, якщо первісна модель використовувала бінарний обмежувач (релейний функцію), то в даний час використовується безліч інших перетворень.

Найбільш розповсюджені і популярні на сьогодні функції активації наведено на рисунку 1.3 [4].

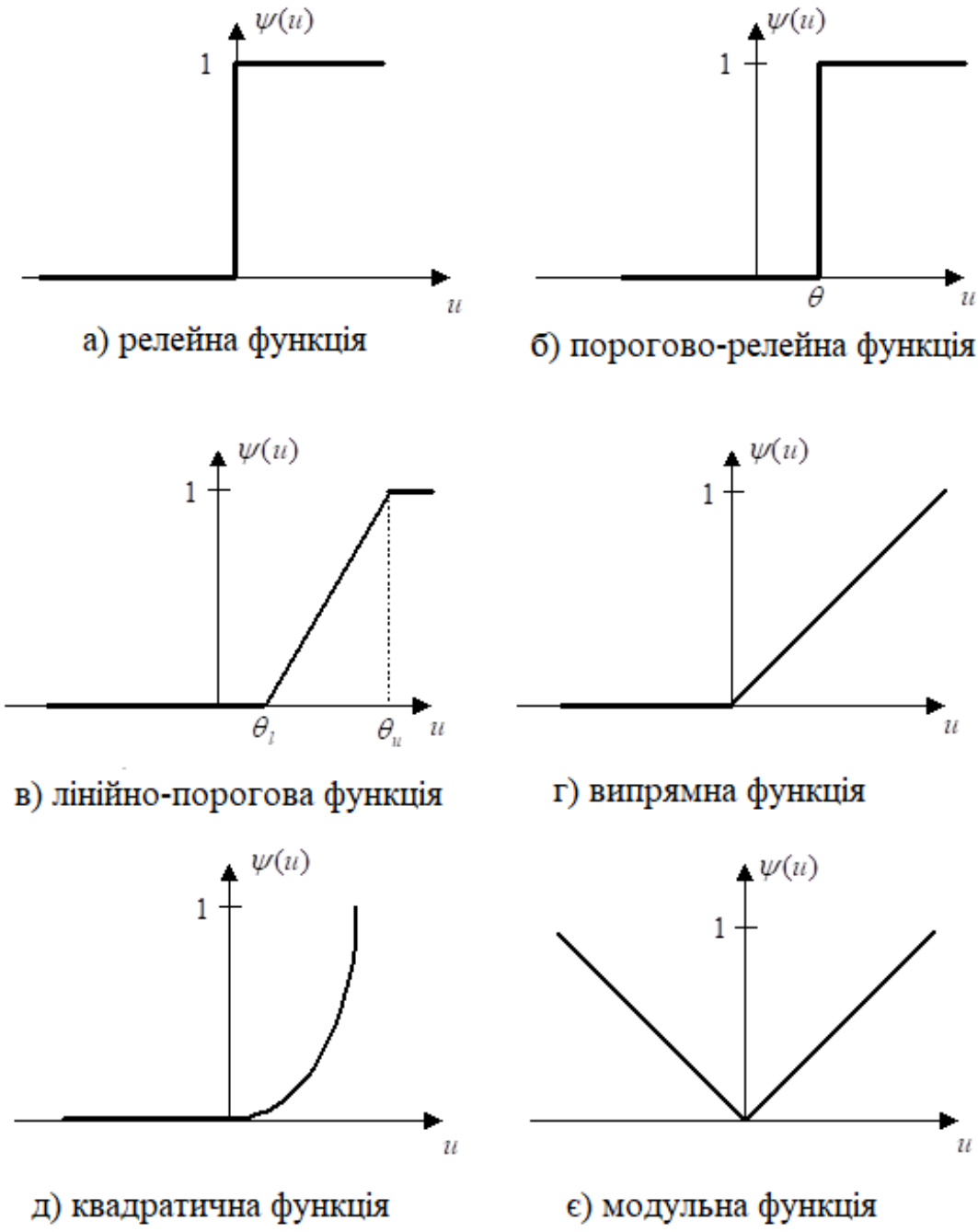


Рисунок 1.3 – Приклади функцій активації

В нейронних мережах найбільшого поширення набула сигмоїдальна функція, яка зображена на рисунку 1.4. Характеристики цієї функції в значній мірі залежать від параметра крутизни (з ростом значення параметра функція наближається до релейної, не зазнаючи при цьому розриву в нульовій точці).

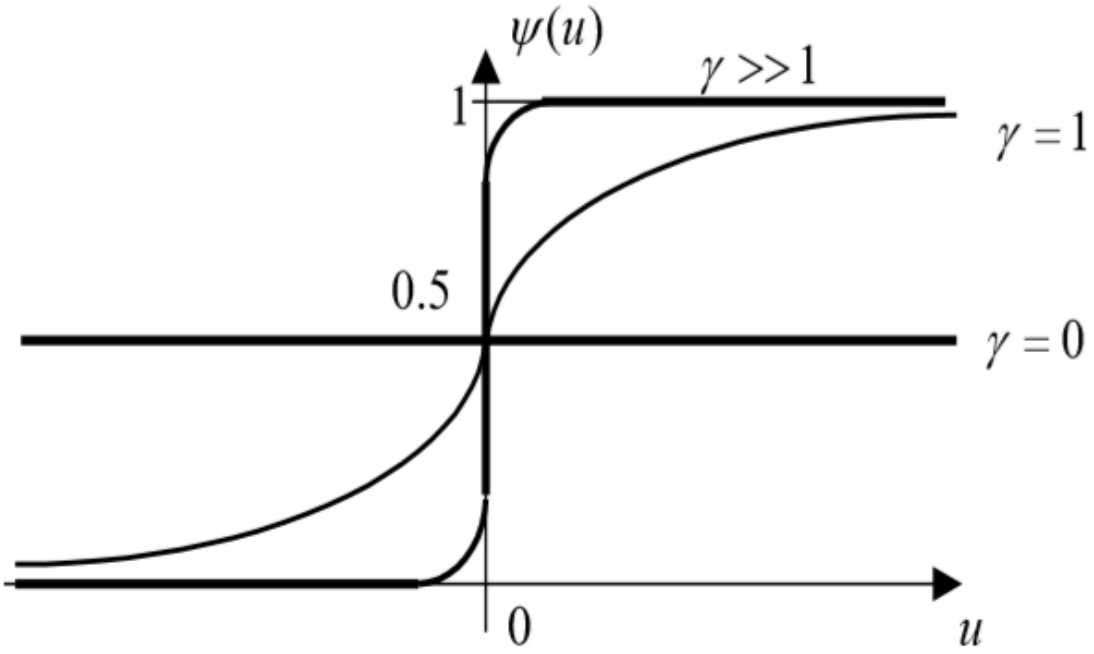


Рисунок 1.4 – Сигмоїдальна функція активації

1.4 Рішення задачі кластерування і топологічної структури на основі нейронних мереж

Класичні методи кластерування можливо реалізовувати на основі нейронних мереж. У дипломній роботі запропоновано самонавчанну нейро-фаззі систему для кластерування великих масивів даних. Також вона застосовується для відображення топологічної структури вхідних даних – перетворення вхідного багатовимірного вектора даних у двовимірний чи тривимірний для зручності подання та аналізу даних.

Існує декілька підходів для вирішення запропонованих завдань. Алгоритм на основі самоорганізованої мапи Когонена. SOM (Kohonen’s Self-Organizing Maps) є методом проектування багатовимірного простору в простір з більш низькою розмірністю з визначеною структурою. Суттєвим недоліком є те, що зменшується розмірність вихідної задачі і остаточний результат роботи нейронних мереж залежить від початкових установок

мережі. Наприклад, необхідно заздалегідь визначити кількість кластерів. У зв’язку з цим, виникають дефекти проектування, аналіз яких є досить складною задачею.

Альтернативою цьому підходу є поєднання конкурентного навчання Хебба і нейронного газу [6]. Це поєднання більш ефективно в побудові топологічної структури, але практичному застосуванню цього підходу перешкоджає ряд проблем: необхідні апріорні знання про розмір мережі і складність застосування методів адаптації швидкості навчання мережі, зайва адаптація призводить до зниження ефективності при навчанні новими даними, а надто повільна швидкість адаптації викликає високу чутливість до збурених даних. Для задач послідовного навчання, перераховані вище методи не підходять.

Таким чином фундаментальна проблема для таких завдань – це адаптація мережі до нової інформації без пошкодження або знищення вже відомої. Тобто неможливо навчати нейронну мережу в режимі послідовного подання даних.

Метою кваліфікаційної магістерської роботі є розробка архітектури робастної інкрементної нейро-фаззі мережі для вирішення задачі кластеризації в умовах класів, що перетинаються.

Об'єктом дослідження є процес опрацювання потоків даних що надходять послідовно в режимі реального часу за допомогою робастної інкрементної нейро-фаззі мережі.

Предметом дослідження є методи інтелектуального аналізу даних при вирішенні задачі робастної кластеризації.

Для побудови штучних нейронних мереж використовуються методи, які базуються на теорії обчислювального штучного інтелекту. Архітектура нечіткої самоорганізованої мапи заснована на основі SOINN і дозволяє в онлайн режимі проводити кластерування і перетворювати багатовимірні дані в двовимірні. Імітаційне моделювання застосовується для перевірки якості кластерування з використанням архітектури FSOINN. Також

необхідно порівняти якість розробленого метода кластерування з іншими алгоритмами на синтетичних та реальних наборів даних.

1.5 Постановка задачі

На основі проведеного аналізу предметної області можна сформулювати завдання, які необхідно вирішити:

− провести аналіз існуючих методів інтелектуального аналізу даних для розв'язання задачі послідовного кластерування даних, що можуть мати викиди;

− розробити метод і архітектуру робастної інкрементної самоорганізованої багатошарової нейронної мережі для потокового кластерування з елементами, що можуть перетинатися;

− на основі SOINN та методу нечітких С-середніх розробити робастну інкрементну нейро-фаззі нейронну мережу для визначення для кожного прикладу ступеню належності до кожного кластеру;

− провести імітаційне моделювання та порівняльний аналіз розробленої нейронної системи з існуючими на даний момент методами.

2 АЛГОРИТМ КЛАСТЕРИЗАЦІЇ FUZZY C-MEANS

Алгоритм нечітких С-середніх також має назву FCM-алгоритм (Fuzzy C-Means, Fuzzy Classifier Means). Головною ціллю алгоритму нечітких С-середніх є автоматизована кластеризація вхідного набору даних, який задається вектором ознак у просторі ознак. Ця система дозволяє встановити кластери і відповідно віднести усі елементи до цих кластерів з вхідного набору даних. Також при використанні цього алгоритму межі між встановленими кластерами залишаються нечіткими. Це робиться тому, що алгоритм нечітких С-середніх допускає, що елемент може належати одразу усім кластерам, але з певною долею (функцією належності). Ця доля розраховується за допомогою відстані між елементом та центроїдами. Даний алгоритм проводить обчислення в декілька епох та на кожному кроці розраховує центроїди і нові долі для кожного з елементів вхідних даних. Основна задача кластеризації за допомогою алгоритму нечітких С-середніх робиться завдяки процесу мінімізації цільовою функції яку ми можемо бачити у формулі 2.1:

$$Q = \sum_{i=1}^{c} \sum_{j=1}^{N} w_{i,j}^{\beta} ||x_j - v_i||^2, \tag{2.1}$$

при обмеженнях наведених у формулі 2.2:

$$\begin{gathered} w_{i,j} \geq 0, \forall i = 1, \dots, c; \ \forall j = 1, \dots, N, \\ \sum_{i=1}^{c} w_{i,j} = 1, \ \forall j = 1, \dots, N, \\ \sum_{j=1}^{N} w_{i,j} > 0, \forall i = 1, \dots, c, \end{gathered} \tag{2.2}$$

де $w_{i,j}$ – рівень належності j-го спостереження до і-го центроїда, а β – потрібний параметр фаззіфікації.

При цьому рівень належності і прототипи кластерів розраховуються

за допомогою формул 2.3 та 2.4:

$$w_{t,j} = \frac{1}{\sum_{i-1}^{c} \left(\frac{||x_j - v_t||}{||x_j - v_i||}\right)^{\frac{2}{\beta-1}}}, \quad \forall t = 1, \dots, c; \ \forall j = 1, \dots, N; \tag{2.3}$$

$$v_t = \frac{\sum_{j=1}^{N} w_{t,j}^{\beta} x_j}{\sum_{j=1}^{N} w_{t,j}^{\beta}}, \quad \forall t = 1, \dots, c. \tag{2.4}$$

З формул 2.3 та 2.4 можна побачити, що при розрахуванні рівня належності конкретного спостереження до центроїду $w_{i,j}$ ми використовуємо відстань між елементом и відповідними центроїдом певного кластеру v_i. Після цього заново розраховується v_i за допомогою рівнів належності до центроїдів $w_{i,j}$. Розрахування йде кожну епоху, доки не буде виконана умова зупинки алгоритму.

Прі знайомстві з цим алгоритмом нечітких С-середніх можно зробити висновок, що він застряє в локальних екстремумах, також це алгоритм пакетної обробки.

3 САМООРГАНІЗОВНА ІНКРЕМЕНТНА НЕЙРОННА МЕРЕЖА

3.1 Самоорганізовна нейрона мережа

Одним із завдань навчання без вчителя є завдання знаходження топологічної структури, яка найбільш точно відображає топологію розподілу вхідних даних. Існує кілька підходів вирішення цього завдання. Наприклад, алгоритм самоорганізованих карт Кохонена є методом проектування багатовимірного простору в простір з більш низькою розмірністю (як правило, двовимірне) з визначеною структурою. У зв'язку зі зниженням розмірності вихідної задачі, і зумовленою структурою мережі, виникають дефекти проектування, аналіз яких є складним завданням. В якості однієї з альтернатив до даного підходу, поєднання конкурентного навчання Хебба і нейронного газу є більш ефективним в побудові топологічної структури. Але практичного застосування даного підходу перешкоджає ряд проблем: необхідні апріорні знання про необхідному розмірі мережі і складність застосування методів адаптації швидкості навчання до даної мережі, зайва адаптація призводить до зниження ефективності при навчанні новим даними, а надто повільна швидкість адаптації викликає високу чутливість до забрудненими даними.

Для задач онлайн навчання або тривалого навчання, перераховані вище методи не підходять. Фундаментальна проблема для таких завдань - як система може пристосуватися до нової інформації без пошкодження або знищення вже відомою.

В даному розділі розглядається алгоритм SOINN, який частково вирішує озвучені вище проблеми.

У задачі навчання без вчителя, нам необхідно розділити дані, що надходять на класи, без попередніх знань про їх кількість. Також необхідно визначити топологію вихідних даних.

Отже, алгоритм повинен відповідати таким вимогам:

− навчання в режимі онлайн, або lifetime;

− поділ на класи без попередніх знань про вхідні даних;

− поділ даних з нечітким кордоном і виявлення основної структури кластерів.

SOINN являє собою нейронну мережу з двома шарами. Перший шар використовується для визначення топологічної структури кластерів, а другий для визначення числа кластерів і виявлення вузлів-прототипів для них. Спочатку навчається перший шар мережі, а потім, використовуючи дані отримані під час навчання першого шару, навчається другий шар мережі.

Для завдання класифікації без вчителя, необхідно визначити, чи належить вхідний образ одного з раніше отриманих кластерів або представляє новий. Припустимо, що два образи належать одному кластеру, якщо відстань (по заздалегідь заданій метриці) між ними менше порога відстані T. Якщо T буде занадто великим, то все вхідні образи будуть віднесені до одного кластеру. Якщо T занадто маленький, то кожен образ буде ізольований як окремий кластер. Для отримання розумного розбиття на кластери, T має бути більше, ніж внутрікласова відстань і менше міжкласової.

Для кожного з шарів мережі SOINN ми повинні обчислити поріг T. Перший шар не має апріорних знань про структуру вихідних даних, тому T підбирається адаптивно, ґрунтуючись на знаннях структури вже побудованої мережі і поточного вхідного образу. При навчанні другого шару, ми можемо порахувати внутрікласову і межкласову відстань і підібрати константне значення порога T.

Для подання топологічної структури, в онлайн або lifetime задачах навчання, зростання мережі є важливим елементом для зниження помилки і адаптації до мінливих умов, зберігаючи старі дані. Але в той же час, неконтрольоване збільшення числа вузлів призводить до перевантаження

мережі і перенавчання її в цілому. Тому необхідно вирішувати, коли і як додавати нові вузли в мережу, а коли запобігти додавання нових вузлів.

Для додавання вузлів, SOINN використовує схему, при якій вставка відбувається в регіоні з максимальною помилкою. А для оцінки корисності вставки, використовується «радіус помилки» є оцінкою корисності вставки. Ця оцінка гарантує, що вставка призведе до зменшення помилки і контролює приріст вузлів, і в кінці кінців стабілізує їх кількість.

Для складання зв'язків між нейронами використовується конкурентне правило Хебба, запропоноване Мартінесом в мережах визначення топології (TRN). Конкурентне правило Хебба може бути описано так: для кожного вхідного сигналу, об'єднуються два найближчих вузла. Доведено, що кожен граф оптимально представляє топологію вхідних даних. В задачах навчання онлайн або lifetime, вузли змінюють своє місце розташування повільно, але постійно. Таким чином, вузли, які є сусідніми на ранній стадії, можливо, не є близькими в більш пізній стадії. Тим самим з'являється необхідність видалення сполук, які останнім часом не оновлюються.

У загальному випадку відбувається також перекриття існуючих кластерів. Щоб визначити кількість кластерів точніше, передбачається, що вхідні дані розділяються: щільності ймовірності в центральній частині кожного кластера вище, ніж щільність в частині між кластерами, а також перекриття кластерів має низьку щільність ймовірності. Поділ кластерів відбувається шляхом видалення вузлів з регіонів з низькою щільністю ймовірності.

Алгоритм SOINN пропонує наступну стратегію для видалення вузлів з регіонів з низькою щільністю ймовірності: якщо число сигналів, що генеруються досі є цілим кратним заданому параметру, то видалити ті вузли, які мають тільки одного топологічного сусіда, або не мають сусідів зовсім і якщо локальний накопичений рівень сигналу є маленьким, то вважаємо такі вузли лежать в області низької щільності ймовірності і видаляємо їх.

3.2 Опис алгоритму SOINN

Тепер, на основі вищезазначеного, ми можемо дати формальний опис алгоритму навчання для мережі SOINN. Цей алгоритм використовується для навчання і першого і другого шарів мережі. Різниця полягає лише в тому, що вхідні дані для навчання другого шару породжуються першим шаром і при навчанні другого шару, ми маємо знання про топологічної структурі першого шару, для обчислення постійного порога подібності. На прикінці навчання другого шару SOINN коректує множину кластерів та повертає прототипи вузлів. Такий саме алгоритм застосовується також при навчанні першого і другого шарів.

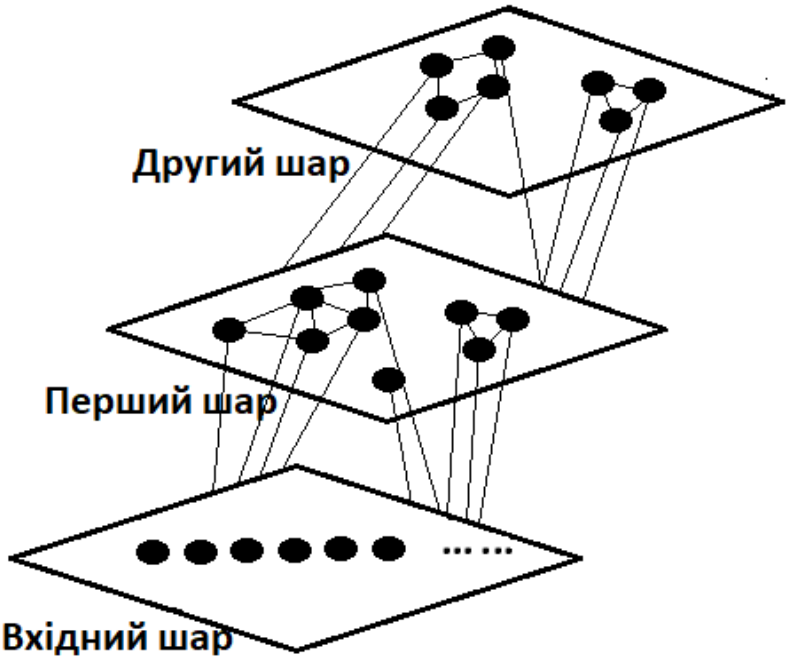


Рисунок 3.1 – Схема SOINN

На рисунку 3.2 представлена блок-схема алгоритму SOINN.

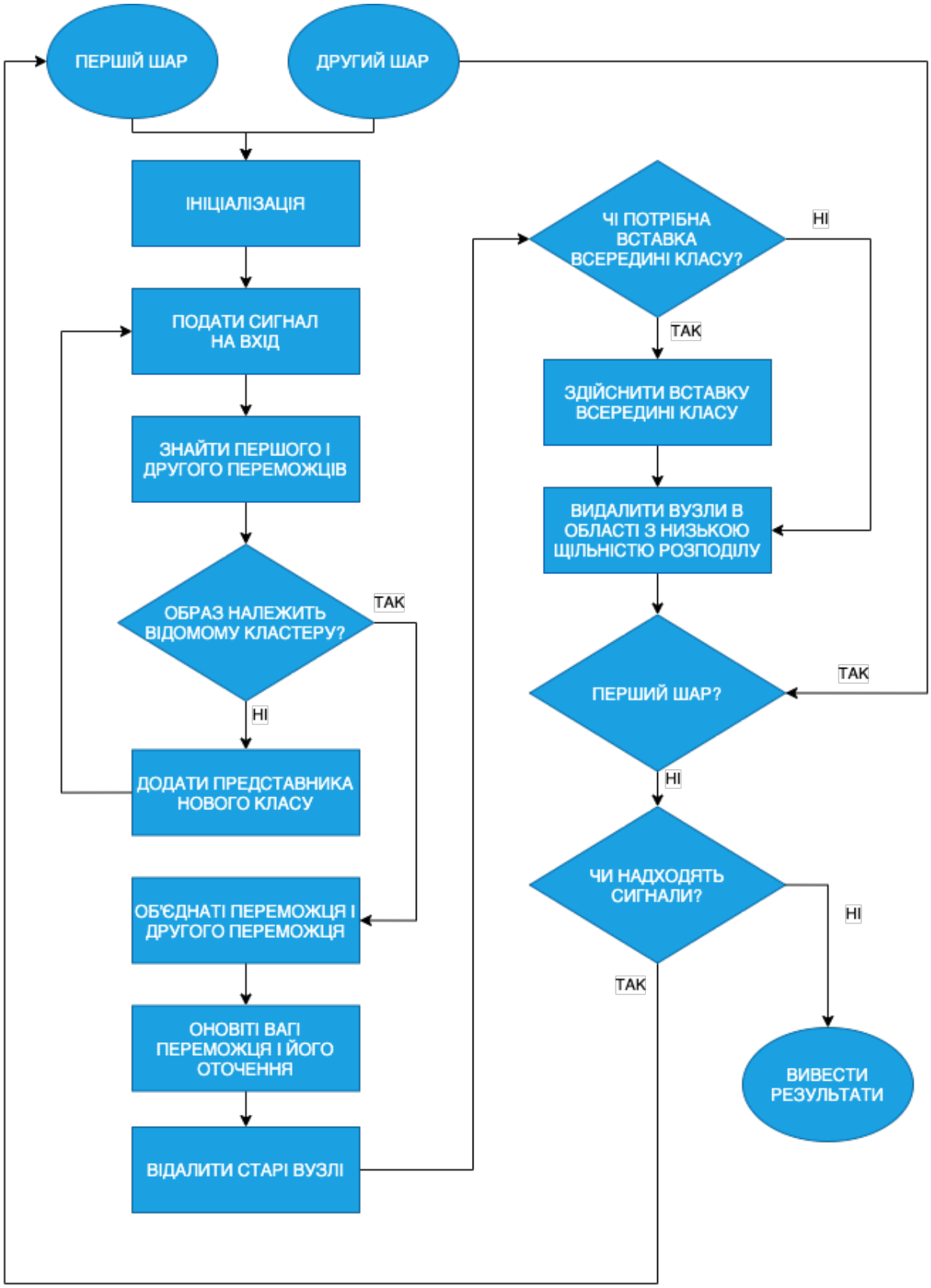


Рисунок 3.2 – Блок-схема алгоритму SOINN

Щоб використати вставку в рамках кластера застосовується наступна логіка: додаємо новий вузол з найбільшою накопиченою помилкою та вузлом, який присутній між сусідніми вузлами з найбільшою накопиченою помилкою. На етапі, коли додається новий вузол, ми оцінюємо цю вставку за коефіцієнтом корисності – якщо радіус накопиченої помилки став іншим, ми вирішуємо чи потрібна ця вставка. Такий підхід може нам гарантувати, що нова вставка буде направляти нас до зменшення помилки та стежить за кількістю вузлів.

Як ми знаємо, кластери можуть перетинатися між собою. Якщо нам потрібно визначити кількість кластерів на високому рівні, ми використовуємо наступне правило – щільність ймовірності в центральній частині кожного кластеру вище, ніж щільність в між кластерами; і перекриття між кластерами має низьку ймовірність.

Щоб оцінити щільність ймовірності і визначити регіон з низькою ймовірністю щільності, якщо щільність нижче порога η, вузол видаляється. Тут пропонується стратегія: якщо кількість вхідних сигналів, є цілим числом, то будуть видалятися вузли тільки з одним або без топологічних сусідів. Звідси виходить, що якщо у вузла є лише один сусід чи нема взагалі, то протягом періоду навчання накопичена похибка цього вузла має дуже низьку ймовірність, а вставка нових вузлів поблизу цього вузла є деструктивним: щільність ймовірності регіону, що містить вузол дуже низький.

Зважаючи на це, можемо описати алгоритм який використовується в SOINN. Цей алгоритм використовується для навчання першого, а також другого шарів мережі. Різниця полягає в тому, що вхідні дані для навчання другого шару генеруються першим шаром, при навчанні другого шару, вже є інформація про топологічну структуру першого шару, для розрахування постійного порога подібності T.

Першим етапом необхідно задати множину вузлів A, яка буде брати участь для збереження всіх вузлів мережі, двома вузлами c_1 і c_2, де A={c_1, c_2} та множину ребр C порожньою множиною. Перші два вузла обираються випадково з поданих на вхід прикладів. Подати на вхід новий сигнал x ϵ R^n. R служить пам'яттю для радіуса помилки вузла і в момент вставки. Далі необхідно знайти в множині A вузли переможця s_1 і другого переможця s_2, як вузли з найближчим і наступним за ним векторами ваг W_c, за формулою

$$s_{1,2}= \arg \min_{c \epsilon A} \|x- W_c\|. \quad (3.1)$$

У випадку коли відстань між x, s_1 та s_2 більше порогів подібності T_{s1} або T_{s2}, додаємо новий вузол, додати до множини A і подати на вхід новий сигнал. В іншому випадку створити ребро між s_1 та s_2, якщо воно не запроваджено. Додати нове ребро до множини ребр C та встановити вік ребра між ними рівним 0. Збільшити вік всіх дуг, що виходять з s_1 на 1. Після чого, додати відстань між вхідним сигналом і переможцем до локальної сумарної помилки E_{s1}. Збільшити локальну кількість сигналів вузла s_1:

$$M_{s1}=M_{s1}+1. \quad (3.2)$$

Адаптувати вектор ваг переможця

$$\Delta W_{s1}=\eta_1(t)(x-W_{s1}) \quad (3.3)$$

та його прямих топологічних сусідів

$$\Delta W_i=\eta_2(t)(x-W_i), \quad (3.4)$$

де $\eta_1(t)$ і $\eta_2(t)$ – крок навчання переможця і його сусідів.

Видалити ребра з віком, вище заданого порогового значення. Якщо генеруються досі вхідні сигнали і їх числа кратні параметру λ, вставити новий вузол і видалити вузли в областях з низькою щільністю. По перше, обирається вузол q з максимальною локальною помилкою $q= \arg\max E_c$. По друге, серед сусідів q знайти вузол f з максимальною локальною помилкою. Додати до мережі новий вузол r та перетворити його у вектор ваг із q і f:

$$W_r=\frac{W_q+ W_f}{2.0}. \tag{3.5}$$

Також це необхідно зробити з локальною кількістю сигналів, радіусом помилки та локальною помилкою вузла.

Якщо вставка не дозволяє зменшити середню помилку цієї локальної області, вставлення не вдалося. Новий вузол r видаляється з набору A і всі параметри відновлюються на встановленні.

Після λ ітерацій (λ виступає таймером) SOINN [2] вставляє нові вузли в положення, де накопичена помилка має велике значення. Відміняється вставка, якщо вона не може зменшити помилку. Вставка називається внутрікласовою вставкою, якщо новий вставлений вузол знаходиться в межах існуючого класу. Крім того, під час вставки новий клас не буде створено.

Потім SOINN знаходить вузли, кількість сусідів яких менша або дорівнює одиниці і видаляє ці вузли. Після того, як поріг навчання [3] аналізує ітерації першого шару, результати навчання використовуються як вхід для другого шару. Другий шар SOINN використовує аналогічний алгоритм навчання, що і перший.

Для другого шару поріг подібності є завжди постійним. Він розраховується з використанням відстані всередині кластера і на основі відстані між кластерами. При великому постійному порозі подібності помилка накопичення для вузлів другого шару буде дуже високою, а всередині класова вставка грає важливу роль в процесі навчання. При

великому постійному порозі подібності другий шар також може видалити деякі «вузли шуму», які залишаються під час навчання на першому рівні.

Щоб видалити вузли, які були викликані шумом, SOINN видаляє вузли в областях з дуже низькою щільністю ймовірності. Якщо кількість вхідних сигналів, що генеруються досі, є цілим числом, яке кратне параметру λ, будуть видаленні ці вузли з одним або зовсім без сусіда. Для одновимірних вхідних даних і наборів даних з невеликим шумом SOINN використовує локально накопичену кількість сигналів вузлів-кандидатів для управління поведінкою видалення.

Також, двошарова мережева структура надає можливість SOINN видаляти вузли викликані шумом, тому що перевірка виконується декілька разів з різним порогом подібності.

Швидкість навчання η(t) визначає ступінь, з якою переможець і сусіди переможця адаптуються до вхідного сигналу і, де t – це загальна кількість кроків адаптації.

Для завдань в режимі онлайн навчання загальна кількість кроків адаптації t недоступна, бо данні подаються на вхід по мірі появи. Тому SOINN використовує схему, схожу з методом під назвою k-середніх, щоб з часом змінювати та адаптувати швидкість з якою навчається алгоритм:

$$\eta_1 = \frac{1}{t},\ \eta_2 = \frac{1}{100t}. \tag{3.6}$$

Де t представляє кількість сигналів що входять, для яких даний конкретний вузол був переможцем, тобто $t = M_i$. Цей сценарій використовується, тому що вузол стає більш стабільним шляхом зниження швидкості навчання, якщо вузол стає переможцем практично на кожному етапі подачі сигналу в рамках одного кластера.

Коли вхідний вектор подається на вхід SOINN [1], він знаходить найближчий вузол (переможець) і другий найближчий вузол (другий переможець) вхідного вектору. Потім використовуючи порогові критерії

подібності мережа визначає, до якого з переможців буде належати вхідний вектор.

Перший рівень SOINN адаптивно оновлює свій поріг подібності кожного вузла, оскільки розподіл вхідних даних нам невідомий. Якщо і-тий вузол має сусідні вузли, поріг подібності T_i буде обчислюватися з використанням максимальної відстані між і-тим вузлом і його сусідніми вузлами

$$T_i = \max_{j \epsilon N_i} \left\| w_i - w_j \right\|. \quad (3.7)$$

N_i – множина сусідніх вузлів, а w_i – векторні ваги вузла і.

Поріг схожості T_i визначається як мінімальна відстань між вузлом та іншими вузлами в мережі, якщо вузол і не має сусідніх вузлів

$$T_i = \min_{j \epsilon N\{i\}} \left\| w_i - w_j \right\|, \quad (3.8)$$

де N – множина всіх вузлів.

Вхідний вектор – новий вузол для представлення першого вузла нового класу, у випадку якщо відстань між вхідним вектором і переможцем або другим переможцем більше, ніж поріг схожості переможця або другого переможця. Якщо вхідний вектор буде визначений як належний до одного кластеру як у переможця або другого переможця і якщо бракує ребра, що з'єднує переможця і другого переможця. Тоді ми з'єднаємо переможця і другого переможця за допомогою ребра та встановимо вік ребра рівним нулю. Надалі будемо збільшувати вік усіх ребр, які будуть пов'язані з переможцем, на одиницю. Після цього оновлюється вектор ваги переможця та його сусідні вузли [4]. Використаємо і для позначення вузла переможця і M_i, щоб показати час, коли він став переможцем. Зміна ваги переможця Δw_i та зміна ваги Δw_j сусіднього вузла $j \epsilon N_i$ визначаються за наступною формулами:

$$\Delta w_i = \frac{1}{M_i}(w_s - w_i), \tag{3.9}$$

та

$$\Delta w_j = \frac{1}{M_i}(w_s - w_j), \tag{3.10}$$

де w_s – вага вхідного вектору.

Однак при використанні нечіткого кластерування слід взяти до уваги, що одиін елмент може мати ознаки декількох кластерів з певним ступенем належності. В таких ситуаціях вибір функції належності грає важливу роль, щоб алгоритм працював коректно. Аналіз збіжності процесів конкурентного самонавчання, проведений М. Котреля і Дж. Фортом, дав зрозуміти, що під час налаштування синаптичних ваг, ми повинні зменшуватися не тільки крок пошуку, а також і параметр ширини функції сусідства. Чим більша відстань переможця від об’єкта, тим менше змінюється його синаптична вага.

4 РОБАСТНА ІНКРЕМЕНТНА НЕЙРО-ФАЗЗІ МЕРЕЖА

4.1 Адаптивні онлайн-методи нечіткої робастної кластеризації

Більшість практичних проблем, пов'язаних з обробкою потокових даних, характеризуються наявністю викидів в даних, що сильно впливає на результати кластеризації з використанням класичних методів, що призводить до виявлення несуттєвих кластерів, зміщення прототипів і радіусів кластерів.

У зв'язку з цим все більше уваги приділяється проблемі кластерного аналізу даних, що генеруються розподілами з повільно падаючими (або важкими) «хвостами». Різні робастні модифікації класичних процедур кластеризації для обробки даних, що містять викиди.

У той же час більшість запропонованих методів стійкою нечіткої кластеризації можна використовувати для послідовної роботи або роботи в реальному часі. Щоб подолати цей недолік, доцільно синтезувати рекурентні процедури для стійкої нечіткої кластеризації часових рядів, які мають адаптивними властивостями, можуть застосовуватися до послідовної обробки вхідних даних, і коли властивості системи, що генерує ці дані, змінюються з часом.

Оцінки, пов'язані з квадратичною цільовою функцією, є оптимальними, коли дані належать класу розподілу обмеженою дисперсії. Найбільш важливим представником цього класу є розподіл Гауса. Змінюючи значення параметра p, можна підвищити надійність процедури кластеризації. Однак слід пам'ятати, що оцінка якості визначається розподілом даних. Наприклад, оцінки, відповідні $p = 1$, є оптимальними для розподілу Лапласа, але для їх отримання потрібно велику кількість обчислень.

Найбільш важливою функцією для наближення щільності ймовірності, близькою до нормального розподілу, є функція

$$p(x_i,\ c_i) = Se(c_i,\ s_i) = \frac{1}{2\,s_i}\ \sinh^{2}\left(\frac{x_i - c_i}{s_i}\right), \qquad (4.1)$$

де c_i, s_i – параметри, що визначають центр і дисперсію розподілу відповідно.

Ця функція близька до гаусової в околиці центру, але істотно відрізняється від неї наявністю важких «хвостів». Розподіл (4.1) пов'язано з цільовою функцією

$$f_i(x_i, c_i) = \beta_i \ln\left[\cosh\left(\frac{x_i - c_i}{\beta_i}\right)\right], \qquad (4.2)$$

де ßi - параметр, що визначає, наскільки швидко ця функція змінюється.

Слід зазначити, що функція (4.2) близька до квадратичної в околі центру c_i і прагне до лінійної з віддаленням від центру. Похідна цієї функції має вигляд

$$f_{i'}(x_i) = \phi(x_i) = \tanh\left(\frac{x_i}{\beta_i}\right), \qquad (4.3)$$

і збігається зі стандартною функцією активації штучного нейрона [44].

Розглянемо функцію

$$D^R\left(x(k), c_j\right) = \sum_{i=1}^{n} f_i\left(x_i(k), c_{ji}\right) = \sum_{i=1}^{n} \beta_i \ln\left[\cosh\left(\frac{x_i(k) - c_{ji}}{\beta_i}\right)\right]. \qquad (4.4)$$

Далі в якості відстані до цільової функції нечіткої кластеризації скористаємося формою

$$D\left(x(k), c_j\right) = \left(D^R\left(x(k), c_j\right)\right)^{\frac{1}{2}}. \qquad (4.5)$$

4.1.1 Імовірнісний підхід

Розглянемо цільову функцію для робастної ймовірнісної кластеризації:

$$E^R\left(w_j(k), c_j\right) = \sum_{k=1}^{N}\sum_{j=1}^{m} w_j^{\beta} D^R\left(x(k), c_j\right) = \\ = \sum_{k=1}^{N}\sum_{j=1}^{m} w_j^{\beta} \sum_{i=1}^{n} \beta_i \ln\left[\cosh\left(\frac{x_i(k) - c_{ji}}{\beta_i}\right)\right]. \tag{4.6}$$

Відповідна йому функція Лагранжа має вигляд:

$$L^R\left(w_j(k), c_j, \lambda(k)\right) \\ = \sum_{k=1}^{N}\sum_{j=1}^{m} w_j^{\beta}(k) \sum_{i=1}^{n} \beta_i \ln\left[\cosh\left(\frac{x_i(k) - c_{ji}}{\beta_i}\right)\right] \\ + \sum_{k=1}^{N} \lambda(k)\left(\sum_{j=1}^{m} w_j(k) - 1\right). \tag{4.7}$$

Сідлову точку функції Лагранжа можна знайти, вирішивши систему рівнянь Куна-Таккера. Градієнт по всім параметрам, що треба налаштувати має вигляд:

$$\nabla_{c_j} L^R\left(w_j(k), c_j, \lambda(k)\right) = \sum_{k=1}^{N} w_j^{\beta} \nabla_{c_j} D^R\left(x(k), c_j\right) = 0, \tag{4.8}$$

очевидно, не має аналітичного рішення.

Рішення (4.8) може бути знайдено чисельно на основі локальної модифікації функції Лагранжа з використанням процедури рекуррентной нечіткої кластеризації. Одночасне перебування сідлової точки локальної функції Лагранжа для метрики (4.4) на основі процедури Ерроу-Гурвіца-Удзави отримуємо наступну процедуру нечіткої робастної кластеризації:

$$\begin{cases} w_j^{pr}(k) = \dfrac{\left(D^R\left(x(k),c_j\right)\right)^{\frac{1}{1-\beta}}}{\sum_{l=1}^{m}\left(D^R(x(k),c_l)\right)^{\frac{1}{1-\beta}}} \\ c_{ji}(k+1) = c_{ji}(k) - \eta(k)\dfrac{\partial}{\partial c_{ji}} L_k^R\left(w_j(k), c_j, \lambda(k)\right) \end{cases}. \quad 4.9$$

4.1.2 Можливісний підхід

В рамках можливісного підходу критерій кластеризації з урахуванням робастної метрики (4.4) має вигляд:

$$E^R\left(w_j(k), c_j\right) = \sum_{k=1}^{N}\sum_{j=1}^{m} w_j^{\beta}(k) D^R\left(x(k), c_j\right) + \sum_{j=1}^{m} \mu_j \sum_{k=1}^{N}\left(1 - w_j(k)\right)^{\beta}. \quad (4.10)$$

Вирішуючи систему рівнянь Куна-Таккера, аналогічну, з використанням метрики (4.4) для перших двох рівнянь, ми можемо отримати рішення у вигляді (5.20 5.21). Однак третє рівняння системи

$$\nabla_{c_j} E^R\left(w_j(k), c_j\right) = \sum_{k=1}^{N} w_j^{\beta} \nabla_{c_j} D^R\left(x(k), c_j\right) = 0 \quad (4.11)$$

повністю збігається з (5.47). Це призводить до неможливості її вирішення в аналітичному вигляді.

Тепер розглянемо локальну модифікацію критерію (4.10). Використовуючи процедуру Ерроу-Гурвіца-Удзави, ми отримуємо рекурентну процедуру нечіткої можливісної кластеризації виду:

$$\begin{cases} w_j^{pos}(k) = \left(1 + \left(\dfrac{D^R\left(x(k),c_j(k)\right)}{\mu_j}\right)^{\frac{1}{\beta-1}}\right)^{-1} \\ c_{ji}(k+1) = c_{ji}(k) - \eta(k)\dfrac{\partial E_k\left(w_j(k),c_j,\mu_j(k)\right)}{\partial c_{ji}} \end{cases}. \quad (4.12)$$

Тут параметр відстані $\mu_j(k)$ обчислюється за формулою для k<N спостережень:

$$\mu_j(k+1) = \frac{\sum_{p=1}^{k} w_j^{\beta}(p) D^R\left(x(p), c_j(k+1)\right)}{\sum_{p=1}^{k} w_j^{\beta}(p)} \quad (4.13)$$

Слід зазначити, що рівняння для $c_{ij}(k)$ в системах (4.9), (4.12) повністю збігаються і визначаються обраної метрикою, а інші рівняння не залежить від метрики, тобто від вибору довільної метрики для процедури кластеризації буде впливати тільки на процедури налаштування прототипу кластера, а рівняння для ваг $w_j^{pr}(k)$ і $w_j^{pos}(k)$ залишиться незмінним.

4.1.3 Робастний метод послідовної нечіткої кластеризації

Як аналог метрики для методу робастної рекуррентной нечіткої кластеризації можна використовувати функцію:

$$D^R\left(x(k), c_j\right) = \sum_{i=1}^{n}\left(1 - \sinh^2\left(x_i(k) - c_{ji}\right)\right)\left(x_i(k) - c_{ji}\right)^{\frac{2}{3}} \quad (4.14)$$

що подавлює аномальні викиди в спостереженнях.

Ця функція задовольняє аксіомам метрики в околиці центру c_j, однак при $|x(k) - c_j| > 0{,}8762$, він не задовольняє нерівностям трикутника.

Використовуючи (4.14), запишемо цільову функцію для робастної кластеризації у вигляді

$$E^R\left(w_j(k), c_j\right) = \sum_{k=1}^{T}\sum_{j=1}^{m} w_j^{\beta} D^R\left(x(k), c_j\right) = \sum_{k=1}^{T}\sum_{j=1}^{m} w_j^{\beta} \sum_{i=1}^{n}\left(1 - \sinh^2\left(x_i(k) - c_{ji}\right)\right)\left(x_i(k) - c_{ji}\right)^{\frac{2}{5}} \quad (4.15)$$

і відповідна функція Лагранжа

$$L^R\left(w_j(k), c_j, \lambda(k)\right) = \sum_{k=1}^{T}\sum_{j=1}^{m} w_j^{\beta}(k) \sum_{i=1}^{n}\left(1 - \sinh^2\left(x_i(k) - c_{ji}\right)\right)\left(x_i(k) - c_{ji}\right)^{\frac{2}{5}} + \sum_{k=1}^{T} \lambda(k)\left(\sum_{j=1}^{m} w_j(k) - 1\right). \quad (4.16)$$

Аналогічно висновку процедури (4.9), використовуючи метод Ерроу-Гурвіца-Удзави для знаходження сідлової точки функції Лагранжа (4.16), ми отримуємо наступну робастну рекурентну процедуру нечіткої кластеризації:

$$\begin{cases} w_j(k) = \dfrac{\left(D^R\left(x(k), c_j\right)\right)^{\frac{1}{1-\beta}}}{\sum_{l=1}^{m}\left(D^R\left(x(k), c_l\right)\right)^{\frac{1}{1-\beta}}} \\ c_{ji}(k+1) = c_{ji}(k) + \eta(k) w_j^{\beta}(k)\left(2\sinh^2\left(x_i(k) - c_{ji}(k)\right)\right) \end{cases}. \quad (4.17)$$

Пропонований метод стійкої рекуррентной нечіткої кластеризації може використовуватися як в пакетному режимі, так і в однопрохідному варіанті. Обчислювальна складність запропонованого методу така ж, як і для інших відомих процедур рекуррентной кластеризації, і лінійно залежить від кількості спостережень в вибірці даних.

4.2 Нечітка кластеризація зі змінним фаззіфікатором

Ще раз введемо в розгляд функцію Лагранжа

$$L\left(u_j(k), c_j, \lambda(k)\right) = \sum_{k=1}^{N}\sum_{j=1}^{m} u_j^{\beta}(k)|x(k) - c_j|^2 + \sum_{k=1}^{N}\lambda(k)\left(\sum_{j=1}^{m} u_j(k) - 1\right) \tag{4.18}$$

де λ(k) позначає невизначений множник Лагранжа.

Вирішивши систему рівнянь Каруша-Куна-Таккера, отримаємо рішення у вигляді

$$\begin{cases} u_j(k) = \dfrac{\left(|x(k)-c_j|^2\right)^{\frac{1}{1-\beta}}}{\sum_{l=1}^{m}\left(|x(k)-c_1|^2\right)^{\frac{1}{1-\beta}}} \\ c_j = \dfrac{\sum_{k=1}^{N} u_j^{\beta}(k)x(k)}{\sum_{k=1}^{N} u_j^{\beta}(k)} \\ \lambda(k) = -\left(\left(\sum_{l=1}^{m}\beta|x(k) - c_l|^2\right)^{\frac{1}{1-\beta}}\right)^{1-\beta} \end{cases} \tag{4.19}$$

що при β = 2 збігається з алгоритмом Fuzzy C-Means (FCM) Дж. Бездека. При β -> 1 результат цього рішення близький до результатів, отриманих за допомогою традиційного алгоритму чіткої кластеризації K-середніх.

В якості альтернативи процедурі, в якій використовується фаззіфікатор $1<\beta<\infty$, цільова функція для нечіткої ймовірнісної кластеризації була запропонована в

$$E\left(u_j, c_j\right) = \sum_{k=1}^{N}\sum_{j=1}^{m}\left(\alpha u_j^2(k) + (1-\alpha)u_j(k)\right)|x(k) - c_j|^2, \tag{4.20}$$

де 0 <α <1 – параметр, що визначає характер отриманого рішення.

Введення в функцію Лагранжа

$$L\left(u_j(k), c_j, \lambda(k)\right) = \sum_{k=1}^{N}\sum_{j=1}^{m}\left(\alpha u_j^2(k) + (1-\alpha)u_j(k)\right)|x(k) - c_j|^2 + \sum_{k=1}^{N}\lambda(k)\left(\sum_{j=1}^{m}u_j(k) - 1\right) \tag{4.21}$$

і рішення системи рівнянь Каруша-Куна-Таккера

$$\begin{cases} \dfrac{\partial L\left(u_j(k), c_j, \lambda(k)\right)}{\partial u_j(k)} = \left(2\alpha u_j(k) + 1 - \alpha\right)|x(k) - c_j|^2 + \lambda(k) = 0 \\ \nabla_{c_j} L\left(u_j(k), c_j, \lambda(k)\right) = -\sum_{k=1}^{N} 2\left(\alpha u_j^2(k) + (1-\alpha)u_j(k)\right)\left(x(k) - c_j\right) = \vec{0} \\ \dfrac{\partial L\left(u_j(k), c_j, \lambda(k)\right)}{\partial \lambda(k)} = \sum_{j=1}^{m} u_j(k) - 1 = 0 \end{cases} \tag{4.22}$$

отримуємо рішення у вигляді

$$\begin{cases} u_j(k) = -\frac{1-\alpha}{2\alpha} + \frac{1 + m^{\frac{1-\alpha}{2\alpha}}}{\sum_{l=1}^{m}\left(\frac{|x(k)-c_j|^2}{|x(k)-c_l|^2}\right)} \\ c_j = \frac{\sum_{k=1}^{N}\left(\alpha u_j^2(k) + (1-\alpha)u_j(k)\right)x(k)}{\sum_{k=1}^{N}\left(\alpha u_j^2(k) + (1-\alpha)u_j(k)\right)} \end{cases} \tag{4.23}$$

При α = 1 ця процедура збігається з FCM. Важливо відзначити, що метод (4.23) не можна використовувати для вирішення завдань Data Stream Mining, оскільки він не може обробляти інформацію в онлайн-режимі. Отже, адаптивна модифікація, введена в

$$\begin{cases} u_j(k+1) = -\dfrac{1-\alpha}{2\alpha} + \dfrac{1+m\dfrac{1-\alpha}{2\alpha}}{\sum_{l=1}^{m}\dfrac{\left(|x(k+1)-c_j(x)|^2\right)}{\left(|x(k+1)-c_j(k)|^2\right)}} \\ c_j(k+1) = c_j(k) + \eta(k)\left(\alpha u_j^2(k+1) + (1-\alpha)u_j(k+1)\right)\left(x(k+1) - c_j(k)\right) \end{cases} \quad (4.24)$$

де η(k) – параметр швидкості навчання.

Легко побачити, що друге рекурентне вираз в (4.24) є правилом самонавчання Кохонена відповідно до принципу «Переможець отримує більше» з функцією сусідства u_j^2 (k + 1) + (1 - α) u_j(k + 1).

4.3 Розвиток адаптивної нейро-фази мережі Кохонена та її онлайн-алгоритму самонавчання в задачах нечіткої кластеризації

Раніше було запропоновано оригінальний метод динамічної нечіткої кластеризації (EFCM). Він заснований на імовірнісному підході до вирішення завдання, при цьому основним параметром, який в кінцевому підсумку визначає кінцевий результат, є радіус формованих кластерів, вибирається з емпіричних міркувань і в кінцевому підсумку визначає кількість можливих класів. Незважаючи на ефективність алгоритмів ймовірнісної нечіткої кластеризації, їх «слабким» місцем є «жорстке» обмеження.

$$\sum_{j=1}^{m} u_j(k) = 1 \forall k = 1,2,\dots,N \quad (4.25)$$

що призводить до того, що спостереження належить в рівній мірі до всіх класів, тобто рівновіддаленим від усіх центроїдів, може мати однакові рівні членства як спостереження, яке також рівновіддаленим від центроїда, але не належить жодному з кластерів.

Можливісний підхід (PCM) до нечіткої кластеризації позбавлений цього недоліку. Такий підхід пов'язаний з оптимізацією цільової функції

$$E(u_j, c_j) = \sum_{k=1}^{N}\sum_{j=1}^{m} u_j^{\beta}(k)|x(k) - c_j|^2 + \sum_{j=1}^{m} \mu_j \sum_{k=1}^{N} \left(1 - u_j(k)\right)^{\beta} \tag{4.26}$$

де $c_j = (c_{j1}, c_{j2}, ..., c_{jn})^T$ – центр j-го кластера, який вираховується в процесі обробки даних, ß> 1 - параметр фаззіфікатора, що визначає «розмитість» меж між кластерами і зазвичай має значення ß = 2, μj> 0 - це скалярний параметр, що визначає відстань, на якому рівень приналежності дорівнює 0,5, т. е. якщо

$$|x(k) - c_j|^2 = \mu_j, \tag{4.27}$$

тоді $u_j(k) = 0{,}5$.

Використання цього підходу веде до модифікованого методу кластеризації:

Крок 1: коли приходить спостереження x(1), формується перший кластер з центроїдом c1.

Крок 2: коли приходить спостереження x(2), умова

$$|x(2) - c_1| \le \Delta \tag{4.28}$$

перевіряється (тут Δ - деякий поріг, встановлений апріорі). Якщо ця умова виконується, то спостереження x(1) не утворює новий центр ваги, тобто воно належить першому кластеру з рівнем приналежності

$$u_1(2) = \left(1 + \left(\frac{|x(2) - c_1|^2}{\mu_1}\right)^{\frac{1}{\beta-1}}\right)^{-1}. \tag{4.29}$$

Якщо умова

$$\Delta < |x(2) - c_1| \leq 2 \tag{4.30}$$

виконується, то корекція центроїда виконується відповідно до правила самонавчання WTA Кохонена [47]:

$$c_1(2) = c_1(1) + \eta(2)\big(x(2) - c_1(1)\big), \tag{4.31}$$

де η(2) ¬ параметр швидкості навчання. При цьому центр ваги c1 «підтягується» до вектору спостережень x(1); якщо для x(1) виконується нерівність

$$2\Delta < |x(2) - c_1|, \tag{4.32}$$

то другий кластер формується з центроїдом

$$c_2 = x(2). \tag{4.33}$$

В цьому випадку рівні членства $u_2(1)$ і $u_1(2)$ слід розраховувати за формулами, наведеними нижче.

Крок N: Тепер нехай є N спостережень і m кластерів з центроїдами cj, обчислення всіх рівнів належності і скоригованих координат центроїдів оцінюються згідно співвідношенням:

$$\begin{cases} u_j(k) = \left(1 + \left(\frac{|x(k)-c_j|^2}{\mu_j}\right)^{\frac{1}{\beta-1}}\right)^{-1} \\ c_j = \frac{\sum_{k=1}^{N} u_j^{\beta}(k)x(k)}{\sum_{k=1}^{N} u_j^{\beta}(k)} \\ \mu_j(k) = \frac{\sum_{k=1}^{N} u_j^{\beta}(k)|x(k)-c_j|^2}{\sum_{k=1}^{N} u_j^{\beta}(k)} \end{cases} \tag{4.34}$$

отримується мінімізацією (4.26) за всіма оціночними параметрами. Система напр. (4.34) – це, по суті, алгоритм пакетної обробки інформації, так що, коли приходить спостереження x(N + 1), всі обчислення повинні бути виконані знову. Зрозуміло, що при досить високій частоті потоку даних розглянутий підхід може виявитися неефективним.

У зв'язку з цим вважаємо за доцільне розробити адаптивну нейро-фаззі систему, що дозволяє обробляти інформацію, що надходить з допомогою повторюваних процедур, що не вимагають зберігання раніше оброблених даних. Можна організувати дійсно оперативну обробку інформації по мірі надходження нових даних, використовуючи замість традиційних пакетних FCM і PCM їх повторювані частини, які по суті є процедурами градієнтної оптимізації для прийнятого критерію самонавчання (кластеризації). Було показано, що ці процедури, по суті, є WTM-правилами для самонавчання Кохонена з функцією звуження околиці і що вони можуть бути успішно використані для навчання самоорганізованих карт з фіксованою архітектурою.

Можна реалізувати систему нейро-фаззі кластеризації на основі дворівневої адаптивної нейро-фаззі мережі.

Перший прихований шар мережі утворено звичайними нейронами Кохонена N_j^K, з'єднаними бічними зв'язками, через які реалізується процес конкуренції. Вихідний шар мережі, утворений вузлами N_j^u, призначений для обчислення рівнів належності кожного спостереження x(k) до кожного j-му кластеру, j = 1, 2, ..., m.

Для настройки Центроїд кластерів використовується самонавчальна рекурентна процедура, яка має вигляд:

$$\begin{cases} c_j(k+1) = c_j(k) + \dfrac{u_j^{\beta}(k)}{k+1}\Big(x(k+1) - c_j(k)\Big) \\ u_j(k+1) = \dfrac{1}{1+\left(\dfrac{|x(k+1)-c_j(k+1)|^2}{\mu_j(k)}\right)^{\frac{1}{1-\beta}}} \\ \mu_j(k+1) = \dfrac{\sum_{p=1}^{k+1} u_j^{\beta}(p)|x(p)-c_j(k+1)|^2}{\sum_{p=1}^{k+1} u_j^{\beta}(p)} \end{cases}. \quad (4.35)$$

Легко побачити, що перше співвідношення в (4.35) є правилом самонавчання WTM з функцією сусідства $(k + 1)^{-1}u_j^{\beta}(k)$.

Процес еволюції системи, як і попередній, починається з одного нейрона Кохонена, який коригує координати першого центроїда c_1. Наступний нейрон додається в мережу при виконанні умови (4.35), яке в цьому випадку приймає вид

$$2\Delta < |x(k) - c_1(k-1)| \tag{4.36}$$

У цей момент формується нейрон з центроїдом $c_2(k) = x(k)$. Тут слід зазначити, що, оскільки в нейронних мережах Кохонена дані попередньо нормовані на гіперсферу, так що

$$|x(k)|^2 = |c_j(k)|^2 = 1 \tag{4.37}$$

нерівність (4.35), що визначає необхідність введення нових нейронів в мережу, приймає вид

$$-1 \le 1 - 2\Delta^2 < c_j^{\mathrm{T}}(k-1)x(k) \le 1, \forall j = 1,2,\dots,m \tag{4.38}$$

або ж

$$-1 \le 1 - 2\Delta^2 < \cos\Big(c_j(k-1), x(k)\Big) \le 1 \tag{4.39}$$

Таким чином, нарощування архітектури відбувається в результаті постійного контролю нерівностей (4.38) або (4.39) і відбувається в тому випадку, якщо ці нерівності порушуються. Слід також зазначити, що тут поріг А має те ж значення, що і в (4.32).

У зв'язку з використанням можливісного підходу доцільно реалізувати іншу «гілку еволюції», а саме, якщо в якийсь момент часу стане

ясно, що рівні приналежності спостереження x(k) не перевищуйте якесь додаткове порогове значення

$$u_j(k) < \varepsilon \forall j = 1,2,\dots,m, \tag{4.40}$$

тобто спостереження x(k) знаходиться досить далеко від всіх вже сформованих центроїдів, це також може служити сигналом для створення нового кластера

$$c_{m+1}(k) = x(k). \tag{4.41}$$

Для оцінки якості нечіткої кластеризації можна використовувати популярний індекс Ксі-Бені в його розширеній формі. Для фіксованого набору даних, що містить N спостережень, цей індекс має вигляд

$$EXB(N) = \frac{\left(\sum_{k=1}^{N}\sum_{j=1}^{m} u_j^{\beta}(k)|x(k)-c_j(N)|^2\right)/N}{\min\limits_{j\neq l}|c_j(N)-c_l(N)|^2} = \frac{NEXB(N)}{DEXB(N)}. \tag{4.42}$$

Включення вираження (4.42) в процедуру навчання дозволяє організувати додатковий контроль над кількістю кластерів, які формуються системою. Так. вводячи третій поріг δ і перевіряючи умова

$$EXB(k+1) > \delta \tag{4.43}$$

на кожному кроці можна зупиняти процес нарощування нейронів в разі порушення нерівності (4.43).

5 ІМІТАЦІЙНЕ МОДЕЛЮВАННЯ І ПЕРЕВІРКА ТЕОРЕТИЧНИХ ДОСЛІДЖЕНЬ

Викиди або «Outlier» – це спостереження, які чисельно віддалені від решти даних, або можна сказати, що це значення, які виходить за межі діапазону. В таблиці 5.1 наведено приклад даних без викидів та з викидами.

Таблиця 5.1 – Порівняння даних з викидами та без

	Дані без викидів	Дані з викидами
Спостереження	1, 2, 3, 3, 5, 4, 5	1, 2, 3, 3, 5, 4, 5, 500
Середнє значення	3.124	59.714
Медіана	3	3
Стандартне відхилення	1.345	150.057

Як можна побачити з таблиці 5.1, набір даних із викидами має суттєво різне середнє та стандартне відхилення. У першому випадку можна зробити припущення, що середнє значення становить 3,124. Але в порівнянні з середніми показниками даних з викидами середнє в цьому випадку сягає 59,714. Це повністю змінює наступні розрахунки та може привести до невірно побудованої моделі.

Випадки виникнення викидів:

– помилки введення даних: – Помилки людини, такі як помилки, спричинені під час збору даних, запису або введення, можуть спричинити відхилення в даних;

– помилка вимірювання: – Це найпоширеніше джерело викидів. Це відбувається, коли використовуваний вимірювальний прилад виявляється несправним;

– «природні викиди»: – коли викид не є штучним (через помилку), це природний викид. Більшість реальних даних належать до цієї категорії.

Викиди можуть бути двох типів: однофакторні та багатофакторні. В таблиці 5.1 було наведено приклад однозначного (однофакторного) відхилення. Ці відхилення можна знайти, коли розглядається розподіл однієї змінної. Багатофакторне відхилення - це відхилення в n-мірному просторі.

5.1 Основні методи знаходження викидів в даних

Z-Score метод, цей метод показує скільки спостережень по середньому значенню відрізняються від стандартного відхилення:

$$Zscore = \frac{X - mean}{StandardDeviation} \tag{5.1}$$

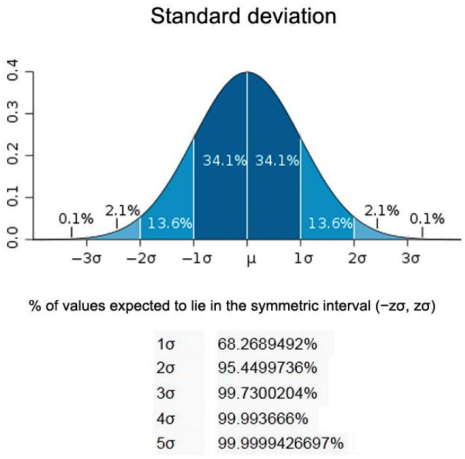


Рисунок 5.1 – Схематичне зображення методу Z-Score

Робастний Z-score метод, цей метод також має назву метод абсолютного медіанного відхилення. В цьому методі замість середнього арифметичного використовується медіана. З точки зору робастних оцінок, тобто оцінок які не пригнічуються викидами, краще використовувати медіану в якості розрахунку:

$$R.Zscore = \frac{Q_3(X_i - Median)}{MAD} \quad (5.2)$$

де

$$MAD = median(|x - median|) \quad (5.3)$$

Q_3 – значення нижнього квартилю.

Наступний метод – метод інтерквартильного діапазону. Цей метод використовує значення інтерквартильного діапазону та основан на квартильних значеннях даних. На рисунку 5.2 схематично зображено розрахунок інтерквартирних діапазонів на основі квартильних значень.

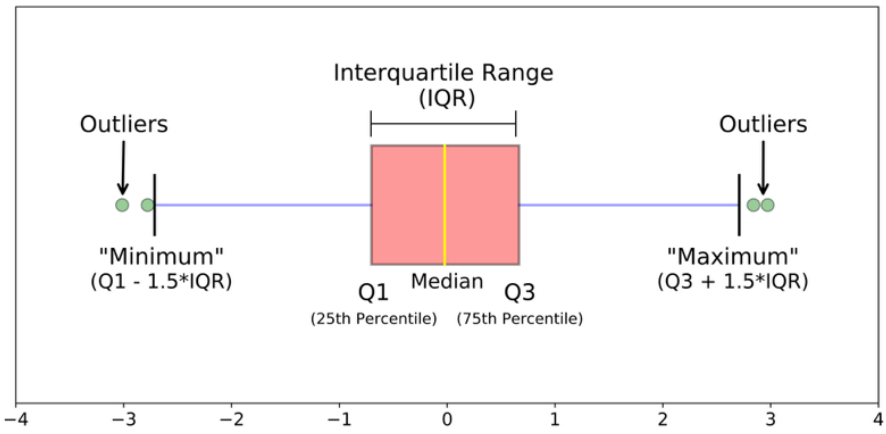


Рисунок 5.2 – Схема методу інтерквартильного діапазону

Метод вінзорізації даних, цей метод подібний до методу інтерквартильного діапазону. Якщо значення перевищує значення 99-го процентилю і нижче 1-го процентилю, в такому випадку ці спостереження можна вважати викидами.

Метод візуального знаходження викидів в даних, цей метод використовує візуальне представлення змінних у вигляді «боксплоті», які по факту також будуються на розрахунку квартильних значень, або будь-якої іншої візуалізації даних. На рисунку 5.3 наведено класичне представлення

змінної через «боксплот».

До візуальних технік знаходження викидів доцільно віднести гістограми щільності розподілу змінних та графіки частот. Завдяки цим методам можна знайти важкі хвати розподілів та по ним зробити припущення про викиди на хвостах розподілів.

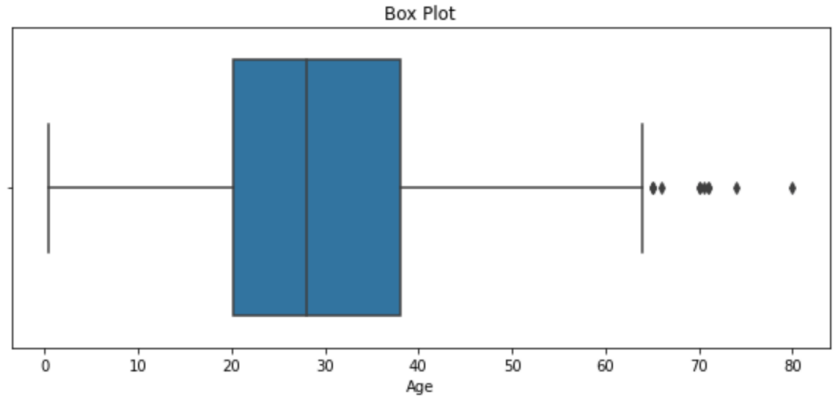


Рисунок 5.3 – «Боксплот»

Але, все ж таки, використовування додаткових етапів попередньої обробки приводить до збільшення часу яке витрачається на будову моделі архітектури системи для вирішення задачі динамічного аналізу даних. Тому, доцільним є використання безпосередньо моделей нейронних мереж або нейро-фаззі систем, які вже є робастними, тобто нечутливими до викидів.

5.2 Розробка моделі робастної інкрементної нейро-фаззі мережі для вирішення задачі кластеризації

Для підтвердження пропонованої архітектури робастної інкрементної нейро-фаззі мережі було обрано декілька наборів даних. Перший набір представляє собою штучносгенеровані три лінійно роздільні кластери, кожен з яких містить по 75 спостережень в діапазоні від -1 до 3, з них 25% є викидами. На рисунку 5.4 наведено початкова ініціалізація цього набору.

Після початкової ініціалізації набору даних, було побудовано модель робастної інкрементної нейро-фаззі мережі та також моделі стандартних методів кластеризації нечітких С-середніх та стандартної інкрементної мапи. Аналіз результатів кожної моделі було виконано на основі декількох метрик оцінки якості кластеризації.

Як відомо, на відміну від класифікації, при вирішенні задачі кластеризації складно оцінити якість результатів. Тут метрика не може залежати від міток, а лише від якості розбиття. Частіше за все, при вирішенні задачі класифікації не має справжніх міток спостережень.

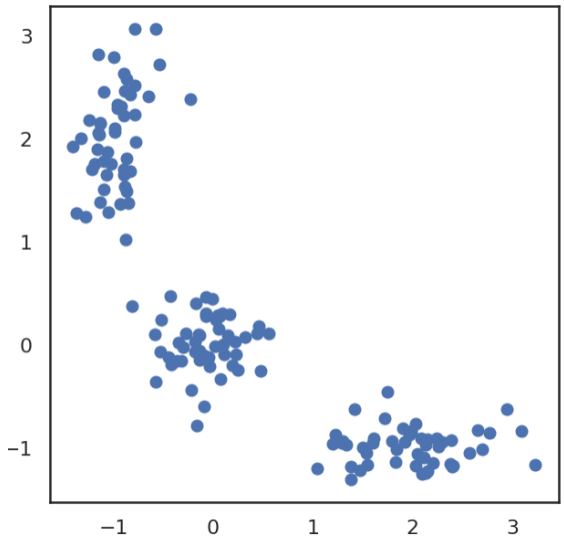


Рисунок 5.4 – Штучносгенерований набір даних (3 кластери)

Існують внутрішні та зовнішні показники якості розбиття даних. Зовнішні метрики використовують інформацію про відомий справжній розподіл, тоді як внутрішні метрики не використовують жодної зовнішньої інформації і оцінюють корисні якості кластерів лише на основі вихідних даних. Оптимальна кількість кластерів зазвичай визначається щодо деяких внутрішніх показників.

– Adjusted Rand Index (ARI) – скорегований індекс довільного розбиття:

$$ARI = \frac{RI - E[RI]}{Max(RI) - E[RI]}, \quad (5.4)$$

де RI – індекс довільного розбиття, який розраховується по співіідношенню вірних міток класів та результуючих міток.

– Однорідність, повнота, V-міра – Формально ці метрики визначаються на основі функції ентропії та умовної функції ентропії, інтерпретуючи вибіркові розбиття як дискретні розподіли:

$$h = 1 - \frac{H(C|K)}{H(C)}, \quad (5.5)$$

$$c = 1 - \frac{H(K|C)}{H(K)}, \quad (5.6)$$

$$v = 2\frac{hc}{h+c}, \quad (5.7)$$

де K – результат кластеризації; C – початкове розподілення по кластерам.

Результати порівняльної якості кластеризації штучнозгенерованого набору даних наведено в таблиці 5.2., а на рисунку 5.5 представлена фінальна візуалізація даних.

Таблиця 5.2 – Порівняння якості кластеризації

	ARI	Однорідність	Повнота	V-міра
FCM	0.66	0.61	0.73	0.71
SOIN	0.75	0.78	0.76	0.69
Robust SOIN	0.89	0.95	0.91	0.93

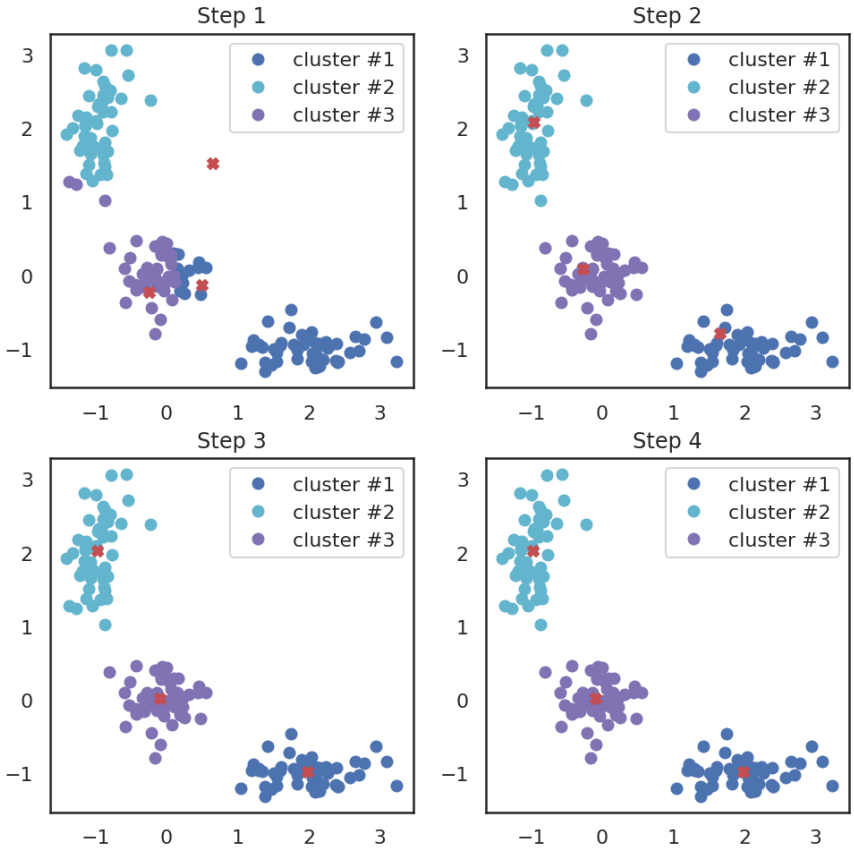
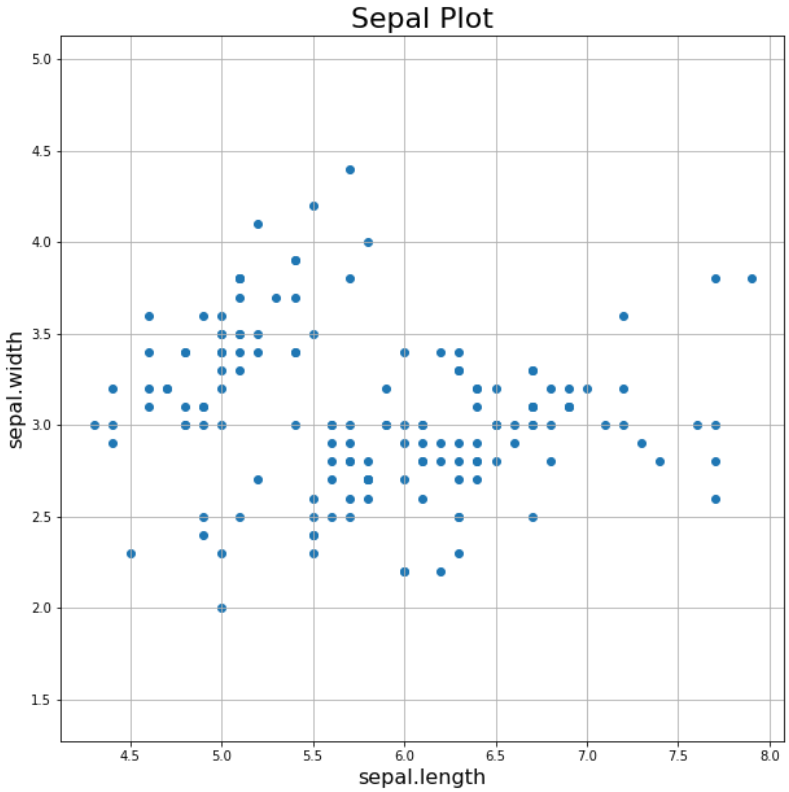
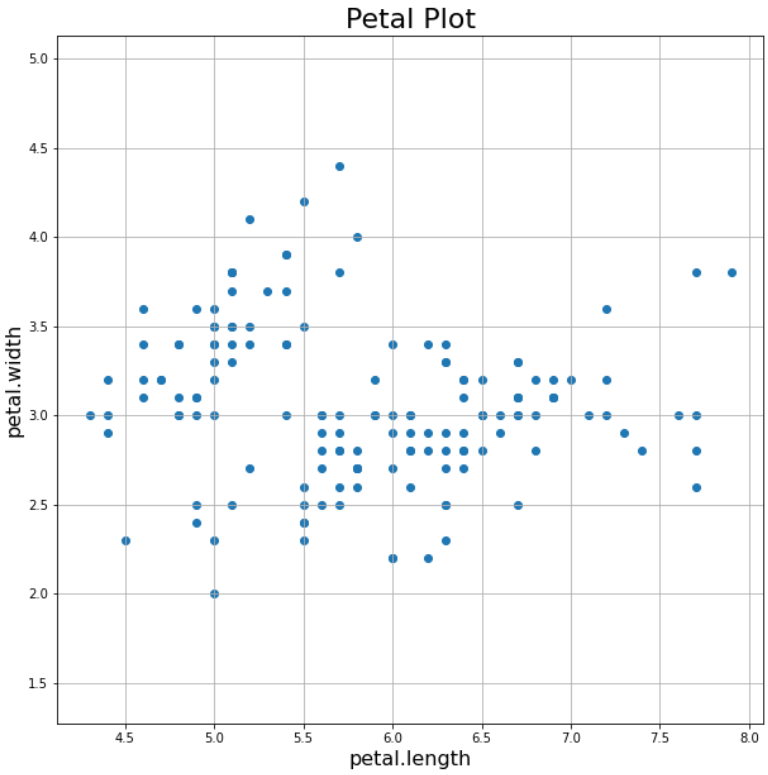


Рисунок 5.5 – Фінальна кластеризація даних розробленою моделлю робастної інкрементної нейро-фаззі мережі

Наступну частину імітаційного моделювання було виконано на вибірці з UCI репозіторію. Вихідні данні мають наступний вигляд (рисунок 5.6). Ця вибірка є дуже відомим тестом для моделювання працездатності систем для вирішення по-перше задачі класифікації, тому що містить атрибут з вірними мітками класифікації, але її дуже часто використовують для тестування розробок пов’язаних в вирішенням задачі кластеризації.



а)



б)

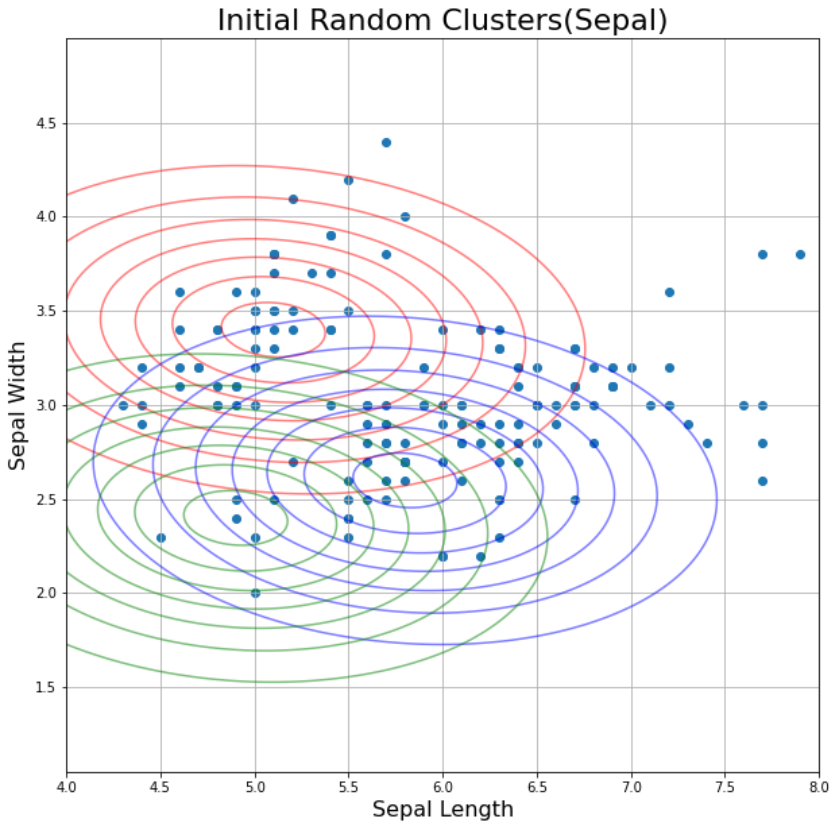
Рисунок 5.6 – Вихідні дані вибірки «Іриси Фішера»: а) – відображення sepal; б) – відображення petal

Чисельні результати якості кластеризації вибірки «Ірриси Фішера» представлені в таблиці 5.3

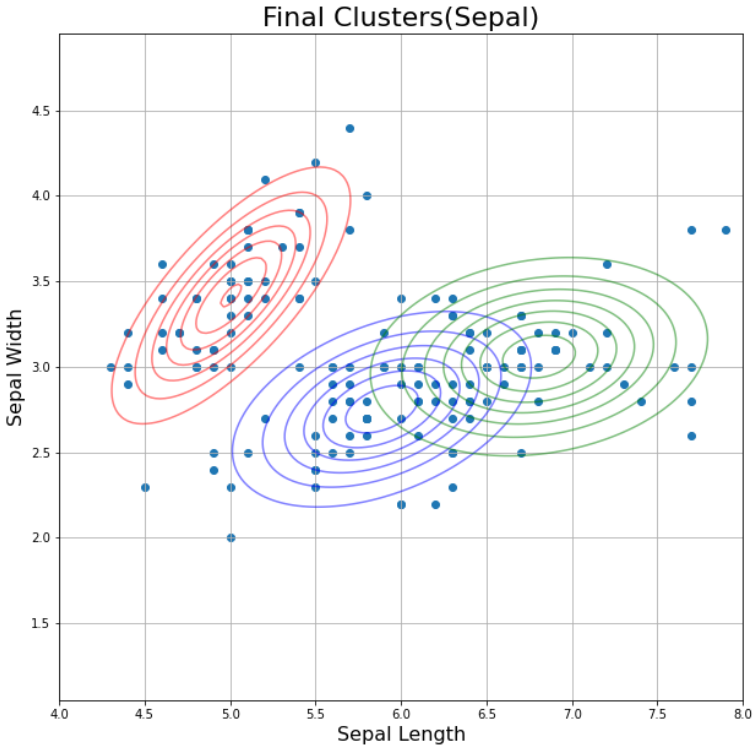
Таблиця 5.3 – Порівняльний аналіз якості кластеризації вибірки «Ірриси Фішера»

	ARI	Однорідність	Повнота	V-міра
FCM	0.71	0.66	0.61	0.72
SOIN	0.74	0.81	0.77	0.79
Robust SOIN	0.91	0.96	0.93	0.96

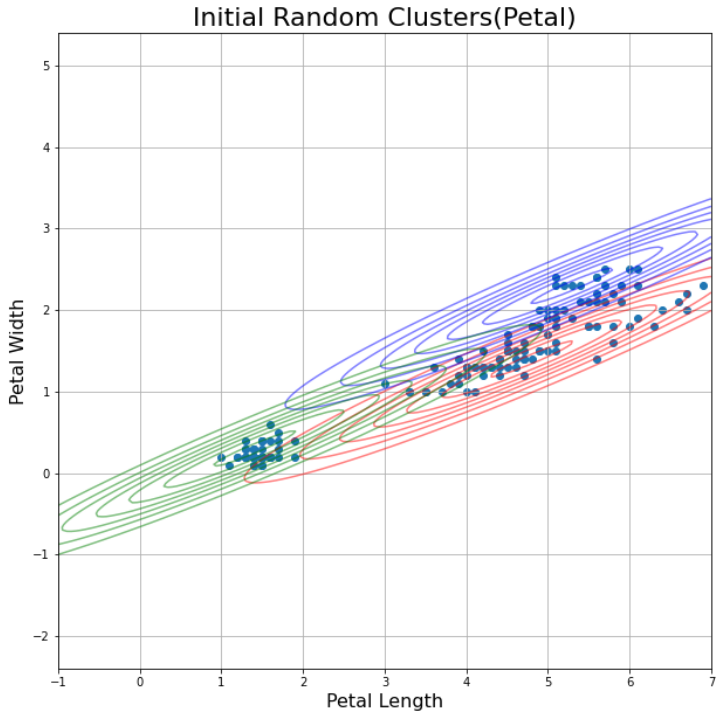
Фінальна візуалізація отриманих результатів кластеризації вибірки «Ірриси Фішера» розробленою моделлю робастної інкрементної нейронної мережі представлена на рисунку 5.7



а)



б)



в)

Рисунок 5.7 – Фінальна кластеризація даних розробленою моделлю робастної інкрементної нейро-фаззі мережі вибірки «Іриси Фішера» : а) – початкове відображення; б) – фінальне відображення; в) – початкове відображення вид збоку

ВИСНОВКИ

В кваліфікаційній магістерській роботі представлено результати, що є відповідними до поставленої мети рішенням актуальної задачі розробки архітектури робастної інкерементної нейро-фаззі мережі для вирішення задачі кластеризації в умовах класів, що перетинаються.

За допомогою створеного алгоритму який поєднує в собі переваги робастних систем, нечітких С-середніх та самоорганізуючих мап можна працювати з масивами даних які послідовно надходять, мають різноманітні ознаки, та можуть мати викиди, тобто можуть буті забруднені на початковому етапі.

Результати, представлені в рамках цієї роботи демонструють переваги алгоритму перед вже існуючими. Після виконаних досліджень і проаналізованих існуючих методів та підходів для потокової кластеризації можна виділити наступні пункти в якості висновків:

− в рамках проведеного аналізу існуючих алгоритмів можна побачити, що реальні дані зазвичай забруднені викидами. З такими вхідними даними стандартні системи кластеризації втрачають свою ефективність. Існуючи класи робастних систем дозволяють демпфірувати таки викиди, але вони працюють лише в пакетному режимі;

− в рамках роботи проаналізовані можливі проблеми кластерування в умовах забруднених даних, що надходять на опрацювання послідовно;

− в рамках роботи поставлене завдання дослідження;

− розроблена архітектура робастної інкрементної нейро-фаззі мережі;

− проведен синтез моделі запропонованої нейро-фаззі архітектури для вирішення завдання кластеризації даних різної фізичної природи;

− отриман повний аналіз розробленої моделі запропонованої нейро-фаззі мережі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Shen F., Osamu H. An Algorithm for Incremental Unsupervised Learning and Topology Representation. *Computer Vision and Pattern Recognition, 2005*: IEEE Computer Society Conference, Tokyo, Japan, 20 July, 2005. IEEE Xplore Digital Library, 2005. Vol. 1. P. 651-656.

2. Shen F. An enhanced self-organizing incremental neural network for online unsupervised learning. *Neural Networks*. 2007. №20. P. 893–903.

3. Shen F., Osamu H. A fast nearest neighbor classifier based on self-organizing incremental neural network. *Neural Networks*. 2008. Vol. 21, No 10. P. 1537–1547.

4. Shen F. An incremental network for on-line unsupervised classification and topology learning. *Neural Networks*. 2006. Vol. 19. No 21. P. 90–106.

5. Borgelt C. Prototype-based Classification and Clustering. Magdeburg, 2005. 350 p.

6. Kohonen T. Self-Organizing Maps. Berlin: SpringerVerlag, 1995. 362 p.

7. Bodyanskiy Ye., Deineko A., KutsenkoY., Zayika O. Data streams fast EM-fuzzy clustering based on Kohonen`s self-learning. *The 1th IEEE International Conference on Data Stream Mining & Processing (DSMP 2016):* proc. of int. conf. Lviv, August 23-27, 2016 p. Lviv, 2016. P. 309–313.

8. Hoeppner F., Klawonn F., Kruse R. Fuzzy-Clusteranalysen. Braunschweig:Vieweg, 1997. 280 p.

9. Бодянский Е.В., Руденко О.Г. Искусственные нейронные сети: архитектуры, обучение, применение. Харьков: ТЕЛЕТЕХ, 2004. 372 с.

10. Бодянский Е. В., Самитова В.А. Нечёткая кластеризация данных в порядковой шкале на основе совместного использования функций принадлежности и правдоподобия. *Збірник наукових праць Харківського університету Повітряних сил.* 2010. No 3. С. 91-95.

11. Томашевский Ю. Б. Нечёткая кластеризация. Саратов: РИЦ СГТУ, 2009.

12. Tsoukalas L. H., Uhrig R. E. Fuzzy and Neural Approaches in Engineering.

N.Y.:John Wiley camp; Sons, Inc., 1997. 587 p.

13. Villmann, T., Schleif, F.-M., & Hammer, B. Supervised neural gas and relevance learning in learning vector quantization. *Proceedings of the Workshop on Self-Organizing Maps (WSOM).* Japan, 2003.

14. King, B. Step-wise clustering procedures. *Journal of American Statistical Association*. 1967. No. 69, P. 86–101.

15. Kohonen, T. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*. 1982. No. 43, P. 59–69.

16. Likas, A., Vlassis, N., & Verbeek, J. J. (2003). The global k-means clustering algorithm. *Pattern Recognition*. 2003. No. 36, P. 451–461.

17. Lim, C. P., Harrison, R. F. A incremental adaptive network for on-line supervised learning and probability estimation. *Neural Net-works*. 1997, No. 10, P. 925–939.

18. Carpenter, G. A., Grossberg, S. The art of adaptive pattern recognition by a self-organizing neural network. *IEEE Computer*. 1998, No 21, P. 77–88.

19. Fritzke, B. Growing cell structures – a self-organizing network for unsupervised and supervised learning. *Neural Networks*. 1994, No. 7, P. 1441–1460.

20. Круглов В. В., Борисов В. В. Искусственные нейронные сети. Теория и практика. М.: Горячая линия, Телеком, 2001. 382 с.

21. Горбань А. Н., Россиев Д. А. Нейронные сети на персональном компьютере. Новосибирск: Наука, 1996. 276 с.

22. Handbook of Intelligent Control: Neural, Fuzzy and Adaptive Approaches. Ed. by D. A. White, D. A. Sofge. N.Y.: Van Nostrand Reinhold, 1992. 568 p.

23. Розенблатт Ф. Обобщение восприятий по группам преобразований. В кн.: «Самоорганизующиеся системы». М.: Мир, 1964. 65-112 с.

24. Розенблатт Ф. Модель памяти на нейронных сетях. *Автоматика*. 1965, No 5, P. 4.