УДК 519.7:004.8
УКПП
№ДР 0116U002539
Інв. №

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки
(ХНУРЕ)
61166, м. Харків, просп. Науки, 14
тел. (057) 7021-016; факс (057) 7021-013

ЗАТВЕРДЖУЮ
Проректор з наукової роботи ХНУРЕ
канд. фіз.-мат. наук

М. В. Неофітний

ЗВІТ
ПРО НАУКОВО-ДОСЛІДНУ РОБОТУ
ДИНАМІЧНИЙ ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ ПОСЛІДОВНОСТЕЙ НЕЧІТКОЇ ІНФОРМАЦІЇ ЗА УМОВ СУТТЄВОЇ НЕВИЗНАЧЕНОСТІ НА ОСНОВІ ГІБРИДНИХ СИСТЕМ ОБЧИСЛЮВАЛЬНОГО ІНТЕЛЕКТУ
№ 307
(остаточний)

Науковий керівник НДР
д-р техн. наук, професор Є. В. Бодянський

2018

Рукопис закінчено 15 грудня 2018 р.

Результати роботи розглянуто Науково-технічною радою ХНУРЕ
протокол №7 від 17 грудня 2018 року

СПИСОК АВТОРІВ

Керівник НДР, г.н.с., д.т.н., проф.	Бодянський Є.В. (реферат, вступ, розд. 1-3, висновки)
Відповідальний виконавець, п.н.с., к.т.н., с.н.с.	Плісс І.П. (розд. 1-3)
г.н.с., д.т.н, проф.	Винокурова О.А. (розд. 1-3)
г.н.с., д.т.н., проф.	Машталір В.П. (розд. 4-6)
г.н.с., д.т.н., проф.	Путятін Є.П. (розд. 4)
г.н.с., д.т.н., с.н.с.	Попов С.В. (підрозд. 1.1.1.)
п.н.с., д.т.н., доц.	Машталір С.В. (розд. 4-6)
с.н.с., к.т.н., доц.	Кобилін О.А. (розд. 5)
с.н.с., к.т.н., доц.	Перова І.Г. (підрозд. 2.3, 3.2)
с.н.с., к.т.н., доц.	Кулішова Н.Є. (підрозд. 2.2)
с.н.с., к.т.н., доц.	Ткачова Т.С. (підрозд. 3.7)
с.н.с., к.т.н.	Тищенко О.К. (підрозд. 1.3, 2.1-2.3)
с.н.с., к.т.н.	Долотов А.І. (підрозд. 3.1, 3.2)

с.н.с., к.т.н.	Дейнеко А.О. (підрозд. 2.1, 2.2)
с.н.с., к.т.н.	Шафроненко А.Ю. (підрозд. 2.3)
с.н.с., к.т.н.	Бойко О.О. (розд. 1)
с.н.с., к.т.н.	Богучарський С.І. (розд. 5)
м.н.с., к.т.н.	Копаліані Д.С. (підрозд. 1.3, 1.4)
м.н.с., к.т.н.	Хаустова (Куценко) Я.В. (підрозд. 2.1, 2.2)
м.н.с., к.т.н.	Самітова В.О. (підрозд. 2.1)
м.н.с. (асп.)	Кобилін І.О. (підрозд. 2.2)
м.н.с. (асп.)	Столбовий М.І. (розд. 5, 6)
м.н.с.	Жернова П.С. (підрозд. 3.6, 3.7)
аспірант	Антоненко Т.Є. (підрозд. 1.6)
студент	Езе Ф.М. (підрозд. 2.1, 2.2)
студент	Заїка О.А. (підрозд. 2.2)
студент	Агафонов В.В. (підрозд. 2.1.1)
студентка	Комір А.О. (підрозд. 2.3.2)

студентка	Чала О.С. (підрозд. 1.4)
студент	Білоног Б.С. (розд. 4, 6)

РЕФЕРАТ

Звіт про НДР: 350 с., 70 рис., 5 табл., 190 посилань.

ДИНАМІЧНИЙ ІНТЕЛЕКТУАЛЬНИЙ АНАЛІЗ, ЕВОЛЮЦІЙНІ НЕЙРО-НЕО-ФАЗЗІ СИСТЕМИ, КАСКАДНІ СИСТЕМИ, НАВЧАННЯ-САМОНАВЧАННЯ, НЕВИЗНАЧЕНІСТЬ, НЕСТАЦІОНАРНІСТЬ, НЕЧІТКА КЛАСИФІКАЦІЯ І КЛАСТЕРИЗАЦІЯ, НЕЧІТКЕ ОПРАЦЮВАННЯ ПОТОКІВ ДАНИХ, ОНЛАЙН РЕЖИМ, ТЕМПОРАЛЬНИЙ АНАЛІЗ ВІДЕОІНФОРМАЦІЇ, ШВИДКОДІЮЧІ АДАПТИВНІ ГІБРИДНІ СИСТЕМИ ОБЧИСЛЮВАЛЬНОГО ІНТЕЛЕКТУ

Об’єкт дослідження – гібридні системи обчислювального інтелекту для вирішення задач Dynamic Data Mining та Data Stream Mining.

Мета роботи – проведення комплексу фундаментальних досліджень, спрямованих на створення теоретичних положень, моделей, методів, архітектур, алгоритмів навчання нових гібридних систем обчислювального інтелекту реального часу, призначених для розв’язання широкого класу задач online інтелектуального аналізу даних високої розмірності та об’ємів з використанням найсучасніших досягнень у цій галузі (Data Science, Data Stream, Evolving Systems, Advanced Fuzzy Clustering, High-Dimensional Clustering).

Методи дослідження – математичний апарат обчислювального інтелекту, а саме: теорія оптимізації, теорія апроксимації, теорія розпізнавання образів, теорія випадкових процесів, хаосдинаміка, теорія ідентифікації та статистичного оцінювання, теорія міри, теорія машинного навчання, теорія нечіткого висновування, теорія матриць, теорія адаптивних систем, Data Science.

Розроблено нові швидкодіючі методи та моделі нечіткого опрацювання потоків даних високої розмірності, створено на їх основі адаптивні гібридні системи обчислювального інтелекту: нейрон-фаззі-, нео-фаззі, вейвлет-нейро-нео-фаззі системи зі спеціалізованою архітектурою, що здатні навчатися, для розв’язання в online режимі задач класифікації, кластеризації, емуляції, апроксимації, прогнозування за умов апріорної та поточної невизначеності-нечіткості, створено нові моделі, методи та інформаційні технології темпорального аналізу відеоінформації за умов невизначеності.

Результати дослідження мають фундаментальний теоретичний характер та можуть бути використані для розв’язання низки практичних завдань і, перш за все, темпоральної обробки потоків відео, Web Mining, Medical Data Mining, Content Based Information (Image, Video) Retrieval, прогнозування, класифікації та кластеризації, визначення розладнань у технічних, виробничих системах, у системах спеціального призначення та Green IT, а також для медичної діагностики, та дозволяють підвищити ефективність розробки систем спостереження за об’єктами, представленими потоками відео.

Отримані наукові результати відповідають світовому рівню, є актуальними та є підґрунтям для створення нових підходів і методів для опрацювання великих та надвеликих обсягів різної фізичної природи на основі глибинного навчання.

ЗМІСТ

ВСТУП

Сьогодні у всьому IT-світі проводяться інтенсивні дослідження, пов’язані з опрацюванням інформації в online режимі, причому слід відзначити, що online системи обчислювального інтелекту – це зазвичай фіксовані архітектури, а тому вирішення задач динамічного та темпорального інтелектуального аналізу даних на їх основі потребує значних обсягів апріорних знань. Сучасні еволюційні системи в основному призначені для роботи з даними типу «об’єкт-властивість», методи нечіткої кластеризації хоча і існують у online варіантах, але для відомої кількості опуклих класів, що ж до високорозмірної кластеризації, яка зараз інтенсивно розвивається у ФРН (F.Klawonn та його школа) та США (J.Bezdek та його школа), то вона взагалі існує лише у пакетних варіантах. Перехід на online вирішення цих задач ми бачимо у відмові від традиційних архітектур нейро-фаззі-систем та традиційних метрик, що полягають в основі сучасних критеріїв навчання-самонавчання.

У той же час, коли на перший план виходять задачі, пов’язані з концепцією Big Data (у сенсі як обсягів даних, так і їх розмірності), відомі доробки і, перш за все ті, що ґрунтуються на парадигмі самонавчання, не пристосовані для вирішення таких задач, а тому потребують суттєвого удосконалення. Найбільш відомими й популярними є багатошарові штучні нейронні мережі (ШНМ) типу MLP, CPN, SVM, RNN та ін., що навчаються, зазвичай, на основі зворотного поширення похибок по багатьом епохам, що суттєво знижує швидкодію.

Реалізувати online режим навчання й опрацювання інформації можна в нейронних мережах, чий вихідний сигнал лінійно залежить від синаптичних ваг, наприклад RBFN, NRBFN, PNN, однак їх використання ускладнюється так званим «прокльоном розмірності», який в принципі можна обійти, але лише в стаціонарному offline режимі.

Нейро-фаззі системи, що об’єднують в собі здатність до навчання нейромереж та прозорість й інтерпретовність результатів м'яких обчислень (Soft Computing), мають низку переваг перед нейромережами. Тут, перш за все, слід відзначити TSK-систему та ANFIS, чиї вихідні сигнали також лінійно залежать від синаптичних ваг, кількість яких, однак, менша, ніж у штучних нейронних мереж. Відомі і більш складні гібридні системи обчислювального інтелекту такі, як, наприклад, гібридні вейвлет-нейро-фаззі-системи, складність навчання яких обмежує їх використання у online режимі.

На сьогоднішній день, коли на основі класичного Data Mining виникли такі нові напрямки, як Dynamic Data Mining, Data Stream Mining, Temporal Data Mining, де інформація надходить у реальному часі у формі багатовимірних часових рядів, потоків відео, тощо, класичні нейронні мережі, системи нечіткого висновування (fuzzy systems), еволюційні алгоритми виявилися неефективними.

За умов опрацювання інформації високого ступеню невизначеності, виникає додаткова проблема вибору архітектури системи. Вирішення цієї проблеми в принципі можна автоматизувати за допомогою апарату еволюційних систем. Тут слід відзначити, що найбільш популярні еволюційні системи DENFIS, EFuNN, eTS, FLEXFIS, SAFIS, SOFNN, SONFIN, PANFIS переналаштовують архітектуру занадто повільно для задач реального часу.

Складними задачами у межах інтелектуального аналізу даних є ті, що базуються на парадигмі самонавчання і, перш за все, задачі кластеризації та сегментації. Найбільш складні ситуації виникають у випадку високого рівня перекриття класів (fuzzy clustering) дуже великих обсягів даних (Big Data) і особливо при високих розмірностях векторів ознак, коли виникає ефект «концентрації норм», що повністю виключає використання традиційних критеріїв.

Крім того, відомі підходи до вирішення цих задач базуються на пакетних процедурах обробки даних.

Тому відомі напрацювання потребують суттєвого вдосконалення та розвитку і, перш за все, створення методів обробки, заснованих на нетрадиційних метриках за умов апріорі невідомої кількості класів, їх форми та рівня перекриття, дефіциту інформації про властивості інформації, що надходить на обробку у режимі реального часу.

На цей час існує гостра проблема розробки online методів опрацювання інформації за умов суттєвої невизначеності щодо властивостей потоків даних і, перш за все, нестаціонарності, нелінійності, хаотичності, нечіткості, кількості та форми кластерів, що утворюються цими даними.

Ефективним апаратом для вирішення таких задач можуть бути гібридні системи обчислювального інтелекту, але відомі такі системи орієнтовані на пакетну обробку стаціонарних даних при відомій кількості класів. Тому є доцільною розробка апарату адаптивних систем обчислювального інтелекту, що мають властивості як параметричної, так і структурної адаптації.

Крім того, наявність великих обсягів неструктурованих відеоданих, що викликана появою нових засобів збору та формування таких даних, призводить до необхідності розробки нових методів аналізу відео, присвячених, в першу чергу, скороченню часу, необхідного для подальшого прийняття рішень, наприклад, в задачах контекстного пошуку. Таким чином, з вище згаданого випливає, що розробка нових методів сегментації/кластеризації відео є вкрай актуальною.

Фундаментальною проблемою, на вирішення якої спрямовано дослідження, є створення нових підходів та методів обчислювального інтелекту на основі гібридизації відомих напрямів та надання їм адаптивних властивостей, що дало можливість опрацьовувати послідовності даних у формі багатовимірних часових рядів великої розмірності або потоків відео,

що надходять на опрацювання або з зовнішнього середовища, або з надвеликих баз даних (VLDB) (концепція Big Data) в режимі реального часу.

У межах загальної проблеми необхідно розробити архітектури нейро-фаззі-, нео-фаззі-, вейвлет-нейро-фаззі-систем типу узагальнених адитивних моделей, що здатні навчатися і опрацьовувати інформацію в режимі реального часу, розв'язуючи задачі класифікації, кластеризації, прогнозування, емуляції за умов невизначеності, нестаціонарності, нелінійності, хаотичності, нечіткості, кількості та форми кластерів-сегментів, що утворюються цими даними.

Метою НДР є проведення комплексу фундаментальних досліджень, спрямованих на створення теоретичних положень, моделей, методів, архітектур, алгоритмів навчання нових гібридних систем обчислювального інтелекту реального часу, призначених для розв'язання широкого класу задач online інтелектуального аналізу даних високої розмірності та об'ємів з використанням найсучасніших досягнень у цій галузі (Data Science, Data Streams, Evolving Systems, Advanced Fuzzy Clustering, High-Dimensional Clustering).

Основними завданнями у ході виконання НДР є розробка нових швидкодіючих методів та адаптивних моделей нечіткого опрацювання інформації обчислювального інтелекту на основі нео-фаззі та вейвлет-нейро-нео-фаззі технологій для вирішення в online режимі задач емуляції, апроксимації, кластеризації, класифікації, прогнозування за умов апріорної та поточної невизначеності-нечіткості і, в першу чергу:

– методи створення архітектур адаптивних нео-фаззі моделей та вейвлет-нейро-нео-фаззі систем, що забезпечують підвищену швидкодію при опрацюванні потоків даних;

– методи нечіткого опрацювання інформації, що не потерпають від ефектів «прокльону розмірності» та «концентрації норм»;

– методи нечіткого опрацювання даних, що здатні оцінювати апріорі невідому кількість класів та кластерів, що можуть змінюватись у процесі аналізу інформації;

– методи нечіткого опрацювання інформації, що є ефективними за умов кластерів та класів довільної форми та високого рівня їх перетинання;

– класифікаційна система ознак з визначенням методів співставлення результатів обробки відеоінформації;

– система просторово-часової сегментації відеорядів та метричного темпорального аналізу часткової семантики динамічної візуальної інформації.

Дані наукові дослідження пов’язані з новим науковим напрямом «Computational Intelligence in Big Data», що був сформований та анонсований IEEE у грудні 2014 року на First IEEE Symposium on Computational Intelligence in Big Data (Флорида, США).

Запропоновані методи, підходи, архітектури, алгоритми навчання «прив’язані» саме до цього наукового напряму, що тільки почав розвиватися.

Об’єднання можливостей цього підходу з апаратом обчислювального інтелекту відкриває нові можливості для виділення широкого класу практичних задач, що традиційно розглядаються в межах класичного Data Mining, але на цей час не мають ефективного вирішення.

На сьогодні, коли обсяги інформації, що підлягає опрацюванню у реальному часі, постійно зростають, а традиційні методи інтелектуального аналізу даних просто не встигають за цим зростанням, актуальним є створення нових підходів до розробки адаптивних систем обчислювального інтелекту для online опрацювання нестаціонарних потоків даних можуть бути використані для вирішення низки практичних завдань і, перш за все, потоків відео, Web Mining, Medical Data Mining, Content Based Information (Image, Video) Retrieval, прогнозування та визначення розладнань у

технічних, виробничих системах, системах спеціального призначення, а також для медичної діагностики.

1 РОЗРОБКА ІНТЕЛЕКТУАЛЬНИХ СИСТЕМ ОБРОБКИ ПОТОКІВ ДАНИХ ВИСОКОЇ РОЗМІРНОСТІ НА ОСНОВІ НЕО-ФАЗЗІ- ТА ВЕЙВЛЕТ-НЕЙРО-НЕО-ФАЗЗІ ПІДХОДУ

1.1 Робастні методи прогнозування та сегментації нестаціонарних часових послідовностей

Останнім часом в задачах аналізу та обробки нестаціонарних сигналів довільної природи за умов невизначеності все ширше застосовуються методи обчислювального інтелекту, серед яких можна виділити гібридні нейронні мережі. Однією з важливих задач, пов’язаних з обробкою сигналів, є прогнозування та сегментація нестаціонарних часових рядів за умов невизначеності. Такі задачі досить часто виникають при опрацюванні потоків відео.

Під задачею прогнозування нестаціонарних часових рядів будемо розуміти вивчення поведінки або стану систем, що аналізуються, у майбутні моменту часу на підставі передісторії станів цієї системи. На цей час досліджено широкий спектр методів прогнозування, починаючи від підходів, що засновані на адаптивних прогнозуючих авторегресійних моделях для лінійного випадку [1] і закінчуючи гібридними нейро-фаззі-мережевими технологіями для нелінійного випадку [2]–[4].

Під задачею сегментації нестаціонарних часових рядів будемо розуміти розбиття часового ряду на однорідні гомоморфні сегменти на основі аналізу змін внутрішніх властивостей часового ряду. На цей час відома низка методів сегментації, а саме за допомогою вейвлет-аналізу [5], фрактально-вейвлетних технологій [6], кусочно-лінійного представлення часових рядів [7], нейро-фаззі технологій [8]–[10] тощо. Залежно від специфіки задачі, що вирішується, можуть бути застосовані два основні типи методів прогнозування і сегментації: реального часу та пакетні.

Для вирішення такого роду задач використовується велика кількість архітектур нейронних мереж, у тому числі і гібридні структури, однак ці системи або громіздкі за своєю архітектурою, або недостатньо пристосовані для навчання в реальному часі. У більшості випадків активаційними функціями таких мереж є сигмоїдальні функції, сплайни, поліноми та радіально-базисні функції.

З іншого боку на сьогодні широке поширення одержала також теорія вейвлет-аналізу [11]–[13], яка дозволяє з високою точністю виявляти локальні особливості нестаціонарних сигналів. На стику цих двох підходів і виникли, так звані, гібридні вейвлет-нейронні мережі [14]–[26] завдяки своїм високим апроксимуючим властивостям і чутливістю до змін характеристик аналізованих процесів.

1.1.1 Робастні критерії для задач прогнозування та сегментації

Для вирішення задач прогнозування та сегментації дуже важливим моментом є вибір критерію оптимізації для синтезу алгоритмів навчання гібридних нейро-мережевих систем.

Розглянемо критерій оптимізації в загальному вигляді

$$E(k) = f(e(k)), \tag{1.1}$$

де $e(k) = d(k) - y(k)$ – похибка навчання;

$d(k)$ – зовнішній навчальний сигнал;

$y(k)$ – отриманий реальний сигнал системи;

$f(\bullet)$ – цільова функція [27], [28].

У більшості випадків як критерії оптимізації використовується критерій найменших квадратів (L_2 –норма)

$$f(e(k)) = \frac{1}{2}e^2(k), \tag{1.2}$$

або критерій найменших значень по абсолютній величині (L_1 –норма)

$$f(e(k)) = |e(k)|. \tag{1.3}$$

Досвід показує, що методи ідентифікації, що засновані на критерії найменших квадратів, виявляються надзвичайно чутливими до відхилень фактичного закону розподілу даних від нормального. За умов різного типу викидів, грубих похибок, неґаусівських збурень із «важкими хвостами» методи, що пов'язані із критерієм найменших квадратів, втрачають свою ефективність. У цій ситуації на перший план виходять методи робастного оцінювання [29], які до теперішнього часу одержали поширення і для навчання штучних нейронних мереж [30]–[33].

Для того щоб зменшити вплив завад з невідомим законом розподілу при вирішенні задач прогнозування і сегментації пропонується використовувати відомі критерії робастної статистики [27, 32]:

– логістична функція:

$$f_L(e(k)) = \beta^2 \ln\left[\cosh\left(\frac{e(k)}{\beta}\right)\right]; \tag{1.4}$$

– функція Х'юбера:

$$f_H(e(k)) = \begin{cases} \dfrac{e^2(k)}{2}, & для\ |e(k)| \le \beta, \\ \beta|e(k)| - \dfrac{\beta^2}{2}, & для\ |e(k)| > \beta; \end{cases} \tag{1.5}$$

– функція Х'юбера з насиченням (функція Талвара):

$$f_T(e(k)) = \begin{cases} \dfrac{e^2(k)}{2} \text{ для } |e(k)| \le \beta, \\ \dfrac{\beta^2}{2} \text{ для } |e(k)| > \beta; \end{cases} \tag{1.6}$$

– функція Хампеля:

$$f_{Ha}(e(k)) = \begin{cases} \dfrac{\beta^2}{\pi}\left[1 - \cos\left(\dfrac{\pi e(k)}{\beta}\right)\right] \text{для } |e(k)| \le \beta, \\ 2\dfrac{\beta^2}{\pi} \text{для } |e(k)| > \beta. \end{cases} \tag{1.7}$$

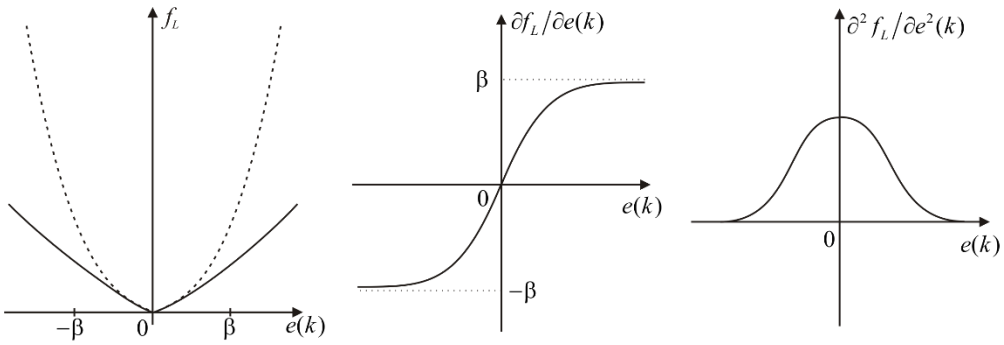
Графіки робастних критеріїв і їхні похідні представлені на рис. 1.1. Перша похідна таких функцій ще називається функцією впливу [28]. Помітимо, що всі чотири критерії двічі неперервно диференційовані на всьому проміжку існування.

Для того, щоб збільшити швидкість збіжності алгоритмів навчання і/або поліпшити апроксимуючі властивості, можливо також використовувати комбіновані критерії оптимізації [34], які в загальному вигляді представляються виразом

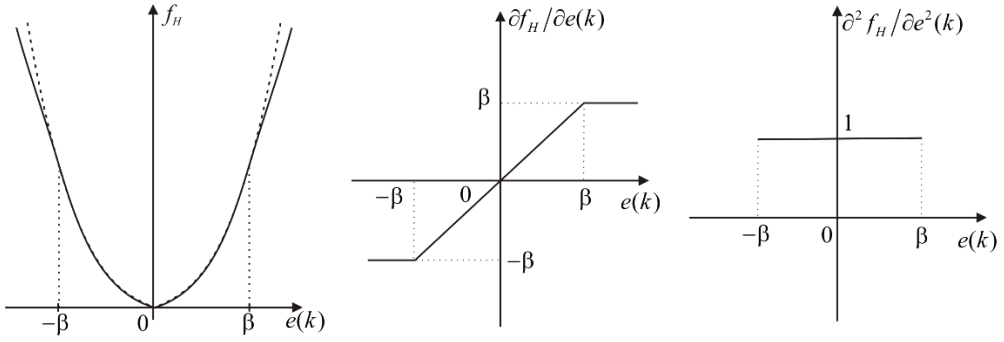
$$E(k) = (1-\lambda) f_1(e(k)) + \lambda f_2(e(k)), \tag{1.8}$$

де $f_1(e(k))$ і $f_2(e(k))$ – відповідні критерії якості, які є опуклими і диференційованими функціями;

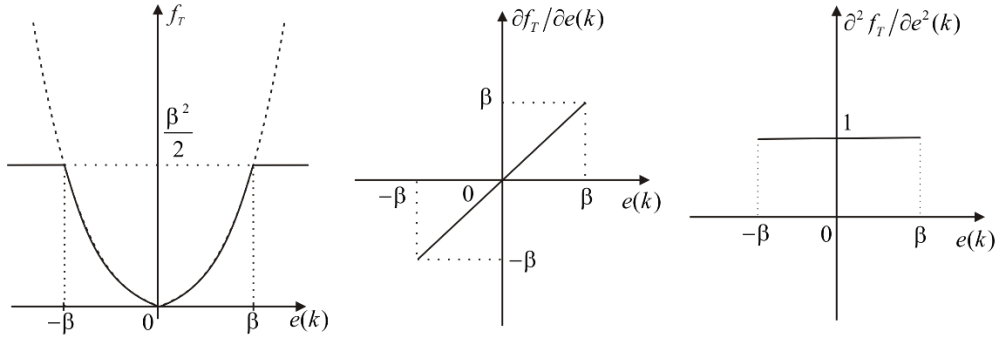
$\lambda \in [0,1]$ – параметр, що поступово зменшується від 1 до 0 у процесі навчання.



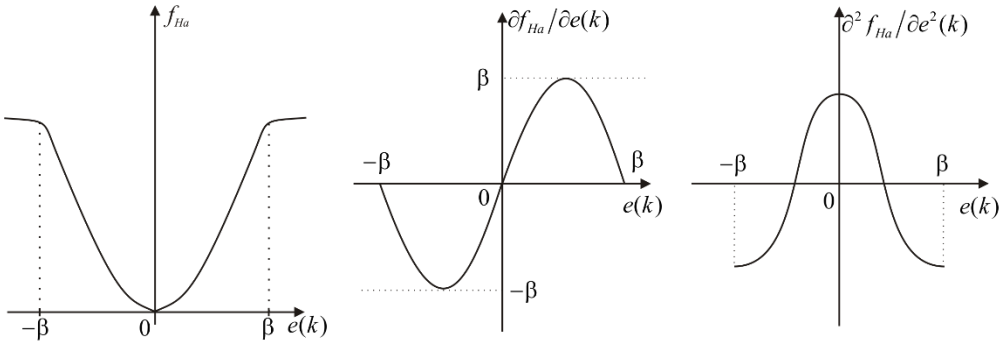
а)



б)



в)



г)

Рисунок 1.1 – Робастні критерії оптимізації та їх перша і друга похідні

У роботі [34] було запропоноване використання параметра λ, що обчислюється відповідно до правила

$$\lambda = \lambda(E) = e^{-\frac{c}{E^2}}, \tag{1.9}$$

де $c > 0$ – додатний параметр;

E – узагальнений критерій якості.

Такий гібридний критерій оптимізації може сполучити в собі як і звичайні, так і робастні локальні критерії оптимізації.

Розглянемо більш докладно критерій оптимізації, що засновано на логістичній функції

$$E(k) = f(e(k)) = \beta^2 \ln\left[\cosh\left(\frac{e(k)}{\beta}\right)\right], \tag{1.10}$$

де β – додатний параметр, що обирається з емпіричних міркувань і визначає розмір зони нечутливості до викидів.

Порівняння робастного критерію оптимізації (1.10) і критерію найменших квадратів (1.2) наведено на рис. 1.2.

Необхідно відзначити, що робастний критерій (1.10) задовольняє аксіомам метричного простору [35]:

1. Невиродженість: $f(x-y) = 0,$ у тому і тільки в тому випадку, коли x і y той самий елемент.

Дійсно, $y = x \Rightarrow f(x-x) = f(0) = \beta^2 \ln\left(\cosh\left(0/\beta\right)\right) = \beta^2 \ln(1) = 0$.

2. Симетричність: $f(x-y) = f(y-x)$. Дійсно,

$$f(x-y) = \beta^2 \ln\left(\cosh\left(\left(x-y\right)/\beta\right)\right) \text{ і}$$

$$f(y-x) = \beta^2 \ln\left(\cosh\left(\left(y-x\right)/\beta\right)\right) = \beta^2 \ln\left(\cosh\left(-\left(x-y\right)/\beta\right)\right) =$$

$$= \beta^2 \ln\left(\cosh\left(\left(x-y\right)/\beta\right)\right) \Rightarrow f(x-y) = f(y-x).$$

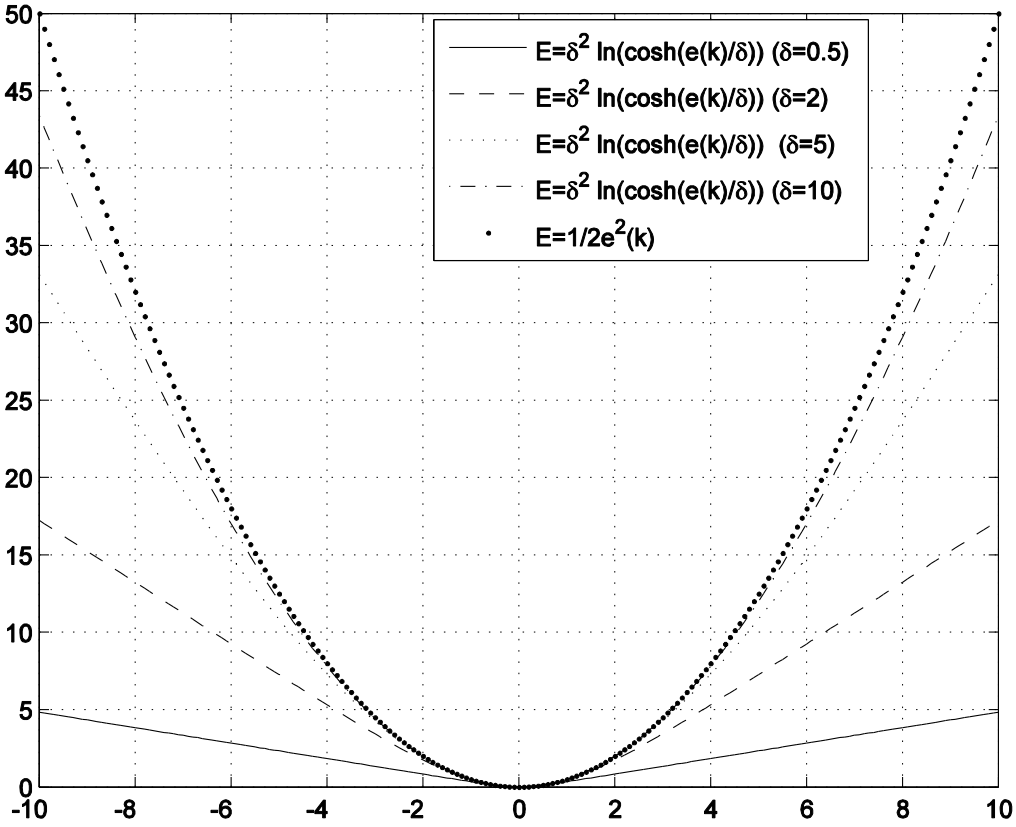


Рисунок 1.2 – Порівняння критеріїв оптимізації

3. Невід'ємність: $f(x-y) \geq 0$.

Дійсно, $f(x-y) = \beta^2 \ln\left(\cosh\left((x-y)/\beta\right)\right) \geq 0$, оскільки функція cosh(•) – додатна.

4. Нерівність трикутника: $f(x-y) \leq f(x-z) + f(z-y)$.

Дійсно,

$$\beta^2 \ln\left(\cosh\left((x-y)/\beta\right)\right) \leq \beta^2 \ln\left(\cosh\left((x-z)/\beta\right)\right) + \beta^2 \ln\left(\cosh\left((z-y)/\beta\right)\right);$$

$$\ln\left(\cosh\left((x-y)/\beta\right)\right) \leq \ln\left(\cosh\left((x-z)/\beta\right)\cosh\left((z-y)/\beta\right)\right);$$

$$0 < \cosh\left((x-y)/\beta\right) \leq \cosh\left((x-z)/\beta\right)\cosh\left((z-y)/\beta\right);$$

$$\frac{e^{\frac{x-y}{\beta}} + e^{-\frac{x-y}{\beta}}}{2} \leq \frac{e^{\frac{x-z}{\beta}} + e^{-\frac{x-z}{\beta}}}{2} \cdot \frac{e^{\frac{z-y}{\beta}} + e^{-\frac{z-y}{\beta}}}{2}.$$

Провівши ряд нескладних перетворень, одержуємо, що

$$2\cosh\left(\frac{x-2z+y}{\beta}\right)\geq 0\,.$$

Ця нерівність виконується при будь-яких значеннях x, y, z, оскільки функція $\cosh(\bullet)$ – додатна.

На основі цих критеріїв можна провести синтез робастних алгоритмів навчання гібридних вейвлет-нейро- та вейвлет-нейро-фаззі- систем, що дозволять обробляти нестаціонарні часові ряди, що забруднені завадами, у реальному часі за умов невизначеності.

1.1.2 Робастний алгоритм навчання гібридного подвійного вейвлет-нейрона

Досить привабливим з погляду технічної реалізації, забезпечення точності та простоти навчання є, так званий, вейвлет-нейрон [36]–[38]. При цьому вейвлет-функції реалізовані або на рівні синаптичних ваг, або на виході нейрона, а для навчання використовується градієнтний алгоритм навчання з постійним кроком. Для поліпшення апроксимуючих властивостей і прискорення процесу навчання введена конструкція, що названа подвійним вейвлет-нейроном, і алгоритм його навчання, що має як згладжуючі, так і слідкуючі властивості [39].

Як активаційні функції подвійного вейвлет-нейрона можна використовувати різні види аналітичних вейвлетів. Одними з найцікавіших за своїми властивостями є два сімейства: POLYWOG-вейвлети та RASP-вейвлети.

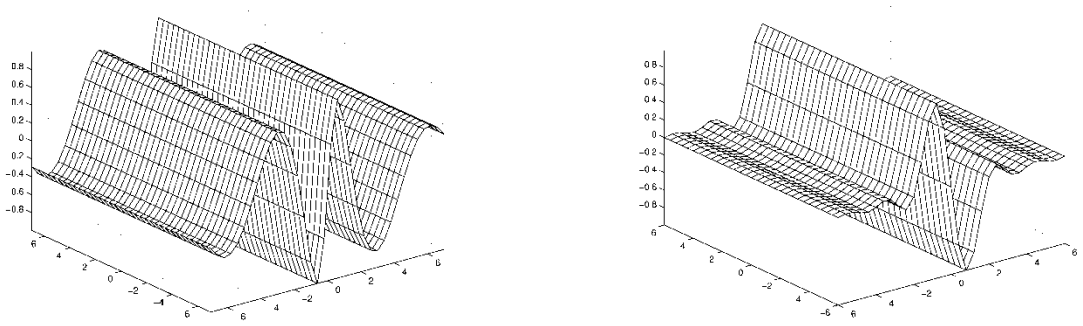
Сімейство вейвлетів RASP – вейвлети на основі раціональних функцій (RAtional functions with Second-order Poles – RASP), пов’язані з теоремою про лишки комплексних змінних [14].

На рис. 1.3 представлені два типових представники материнських вейвлетів RASP, що описуються виразами

$$\varphi_{ji}^{1}(x_i(k)) = \frac{\beta^1 \cos(x_i(k))}{x_i^2(k)+1}, \quad \beta^1 = 2.7435, \tag{1.11}$$

$$\varphi_{ji}^{2}(x_i(k)) = \frac{\beta^2 \sin(\pi x_i(k))}{x_i^2(k)-1}, \quad \beta^2 = 0.6111. \tag{1.12}$$

Ці вейвлети є дійсними непарними функціями з нульовим середнім.



а) б)

Рисунок 1.3 – Представники сімейства вейвлетів RASP

Ще одне досить широке сімейство вейвлетів можна одержати з поліноміальних віконних гаусових функцій (POLYnomials Windowed with Gaussians type of function – POLYWOG) [14]. Цікаво відмітити, що похідні від цих функцій також є вейвлетами POLYWOG і можуть використовуватися як материнські вейвлети.

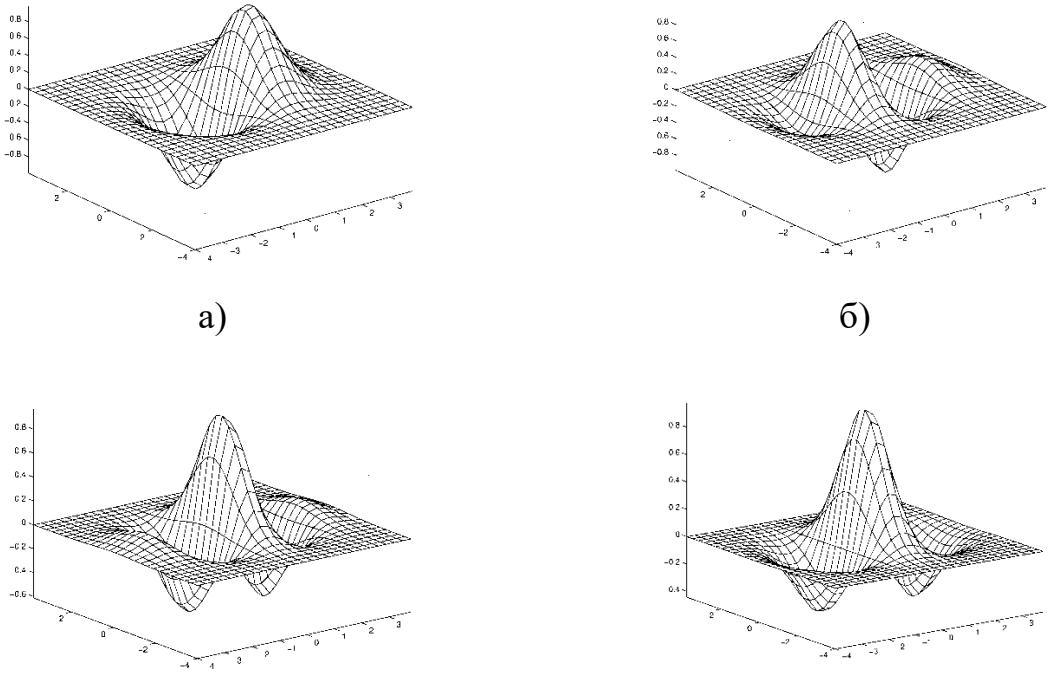
На рис. 1.4 представлено декілька типових вейвлетів із сімейства POLYWOG, що описуються виразами

$$\varphi_{ji}^{1}(x_i(k)) = \mu^1 x_i(k) \exp\left(\frac{-x_i^2(k)}{2}\right), \quad \mu^1 = \exp\left(-1/2\right), \tag{1.13}$$

$$\varphi_{ji}^{2}(x_i(k)) = \mu^2\left(x_i^3(k) - 3x_i(k)\right)\exp\left(\frac{-x_i^2(k)}{2}\right), \quad \mu^2 = 0.7246, \tag{1.14}$$

$$\varphi_{ji}^{3}(x_i(k)) = \mu^3\left(x_i^4(k) - 6x_i^2(k) + 3\right)\exp\left(\frac{-x_i^2(k)}{2}\right), \quad \mu^3 = 1/3, \tag{1.15}$$

$$\varphi_{ji}^{4}(x_i(k)) = \left(1 - x_i^2(k)\right)\exp\left(\frac{-x_i^2(k)}{2}\right). \tag{1.16}$$



а) б)

в) г)

Рисунок 1.4 – Представники сімейства вейвлетів POLYWOG

Деякі вейвлети сімейства POLYWOG можуть бути отримані за допомогою простих генераторів. Так, зокрема, вейвлети цього сімейства можуть бути сгенеровані з урахуванням властивостей ермитовості похідної полінома і функції Ґауса.

Введемо до розгляду структуру подвійного вейвлет-нейрону, що наведено на рис. 1.5 [40]. Як видно, подвійний вейвлет-нейрон досить близький по конструкції до n-входового вейвлет нейрона, проте містить нелінійні вейвлет-функції на рівні синаптичних ваг, а також і на виході структури.

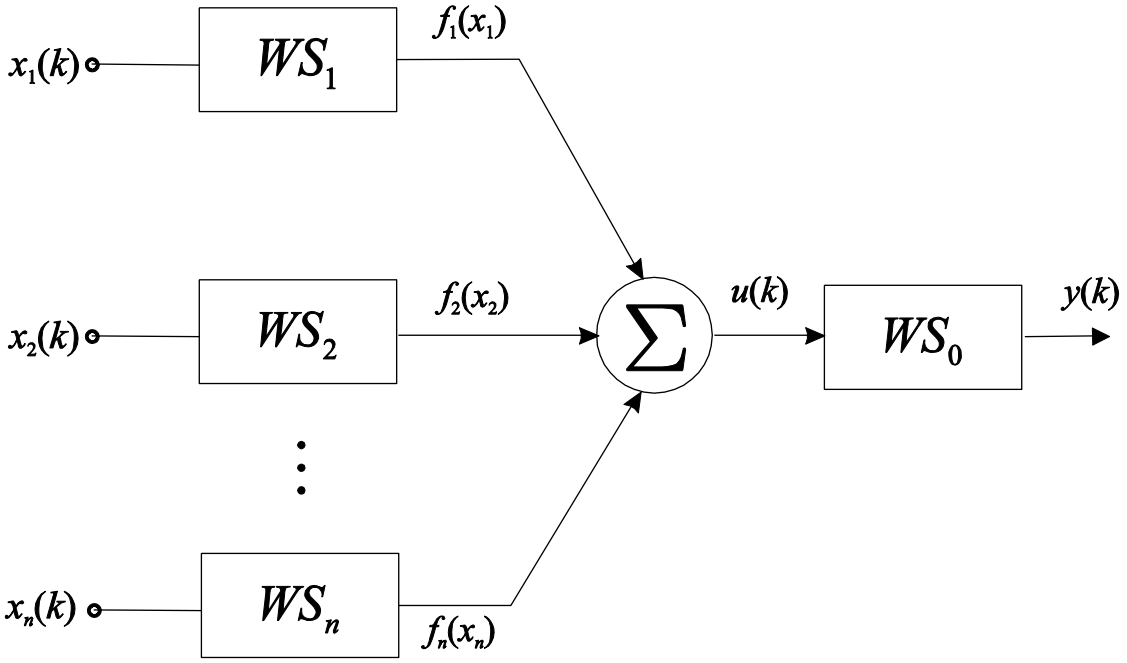


Рисунок 1.5 – Узагальнена структура подвійного вейвлет-нейрона

При подачі на вхід подвійного вейвлет-нейрона, що наведено на рис. 1.6, векторного сигналу $x(k)=(x_1(k),\ x_2(k),\ \ldots,\ x_n(k))^T$ на його виході з'являється скалярний сигнал вигляду

$$\begin{aligned} y(k) &= f_0\left(\sum_{i=1}^{n} f_i(x_i(k))\right) = f_0(u(k)) = \\ &= \sum_{l=0}^{h_2}\varphi_{l0}\left(\sum_{i=1}^{n}\sum_{j=0}^{h_1}\varphi_{ji}(x_i(k))w_{ji}(k)\right)w_{j0} = \sum_{l=0}^{h_2}\varphi_{l0}(u(k))w_{l0}(k), \end{aligned} \tag{1.17}$$

який визначається як синаптичними вагами $w_{ji}(k),\ w_{l0}(k)$, що настроюються, так і значеннями вейвлет-функцій $\varphi_{ji}(x_i(k)),\ \varphi_{l0}(u(k))$, при цьому обумовлюється, що $\varphi_{00}(\bullet)=\varphi_{0i}(\bullet)\equiv 1$.

Подвійний вейвлет-нейрон складається із двох шарів: прихованого шару, в якому n вейвлет-синапсів по h_1 вейвлет-функції у кожному та вихідного шару, що складається з одного вейвлет-синапсу з h_2 вейвлет-функціями.

У кожному вейвлет-синапсі реалізовані вейвлети, що відрізняються між собою параметрами розтягання (ширини) та зсуву (центру).

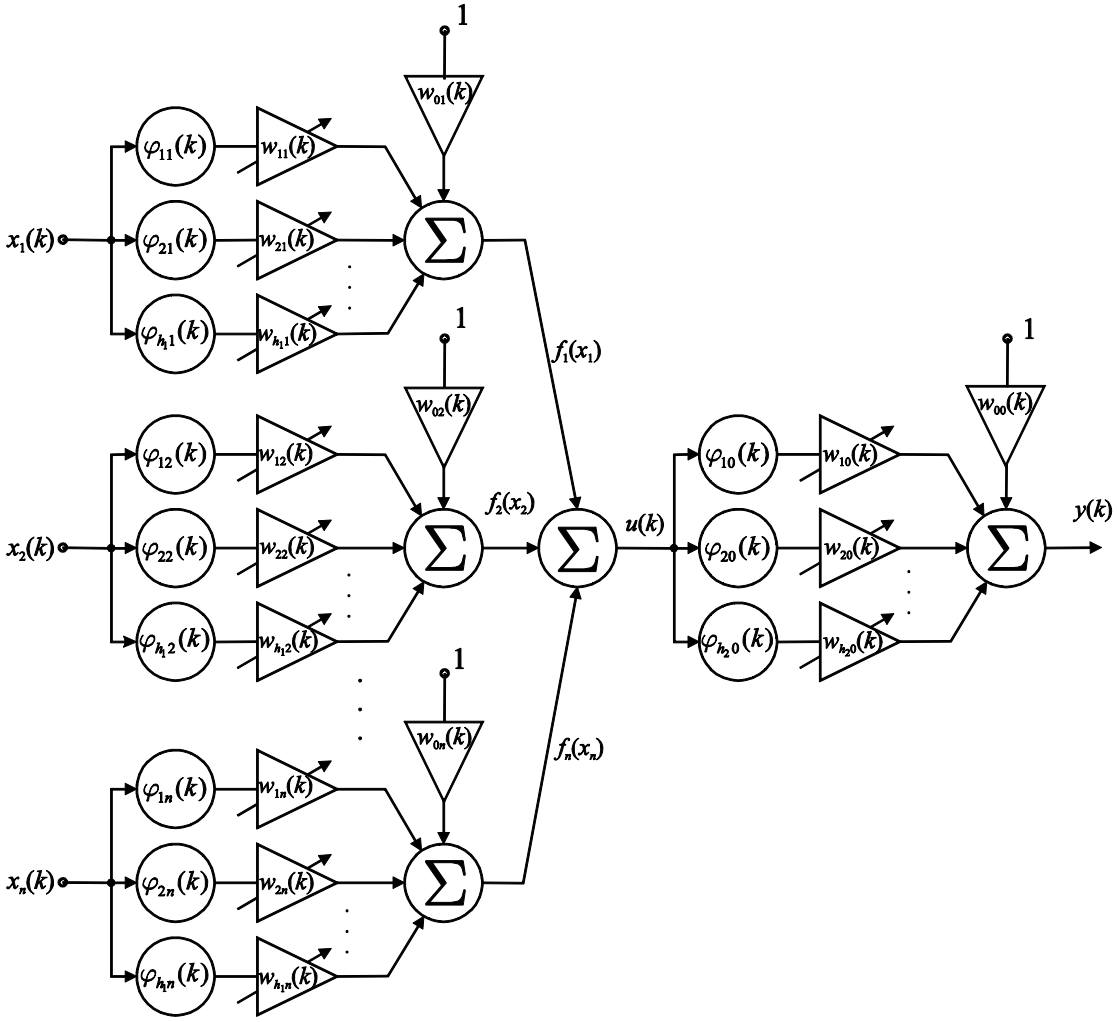


Рисунок 1.6 – Архітектура подвійного вейвлет-нейрона з нелінійними вейвлет-синапсами

Розглянемо робастний алгоритм навчання такої архітектури [41]. Для навчання вихідного шару гібридного робастного подвійного вейвлет-нейрона введемо до розгляду похибку навчання

$$e(k) = d(k) - y(k),$$

на основі якої введемо робастний критерій вигляду

$$E(k) = \beta^2 \ln\left[\cosh\left(\frac{e(k)}{\beta}\right)\right], \tag{1.18}$$

де β – додатний параметр, що обирається з емпіричних міркувань і визначає розмір зони нечутливості до викидів.

Робастний алгоритм навчання вихідного шару подвійного вейвлет-нейрона на основі градієнтного підходу має вигляд

$$w_{j0}(k+1) = w_{j0}(k) + \eta_0(k)\beta \tanh\left(\frac{e(k)}{\beta}\right)\varphi_{j0}(u(k)), \tag{1.19}$$

або у векторній формі

$$w_0(k+1) = w_0(k) + \eta_0(k)\beta \tanh\left(\frac{e(k)}{\beta}\right)\varphi_0(u(k)), \tag{1.20}$$

де $w_0(k) = (w_{10}(k), w_{20}(k), \ldots, w_{h_2 0}(k))^T$ – вектор синаптичних ваг;

$\varphi_0(k) = (\varphi_{10}(k), \varphi_{20}(k), \ldots, \varphi_{h_2 0}(k))^T$ – вектор вейвлет-активаційних функцій;

$\eta_0(k)$ – крок навчання, що підлягає визначенню.

Для збільшення швидкості збіжності процесу навчання варто перейти від градієнтних процедур до квазін’ютонівських алгоритмів, найбільше поширення серед яких одержав алгоритм Левенберга-Марквардта.

Після нескладних перетворень [16] одержуємо алгоритм навчання у вигляді

$$\begin{cases} w_0(k+1) = w_0(k) + \dfrac{\beta \tanh\left(\dfrac{e(k)}{\beta}\right)\varphi_0(u(k))}{\gamma_i^{w_0}(k)}, \\ \gamma_i^{w_0}(k+1) = \alpha\gamma_i^{w_0}(k) + \left\|\varphi_0(u(k+1))\right\|^2, \end{cases} \tag{1.21}$$

де α – параметр забування застарілої інформації ($0<\alpha<1$).

Навчання прихованого шару проводиться аналогічним чином на основі зворотного поширення похибок з використанням того ж критерію, але записаного у формі

$$E(k)=\beta^2\ln\left[\cosh\left(\frac{d(k)-f_0(u(k))}{\beta}\right)\right]= \\ =\beta^2\ln\left[\cosh\left(\frac{\left(d(k)-f_0\left(\sum_{i=1}^{n}\sum_{j=0}^{h_1}\varphi_{ji}(x_i(k))w_{ji}(k)\right)\right)}{\beta}\right)\right]. \quad (1.22)$$

Робастний алгоритм навчання прихованого шару подвійного вейвлет-нейрона на основі градієнтної оптимізації має вигляд

$$w_{ji}(k+1)=w_{ji}(k)+\eta(k)\beta\tanh\left(\frac{e(k)}{\beta}\right)f_0'(u(k))\varphi_{ji}(x_i(k)), \quad (1.23)$$

або у векторній формі

$$w_i(k+1)=w_i(k)+\eta(k)\beta\tanh\left(\frac{e(k)}{\beta}\right)f_0'(u(k))\varphi_i(x_i(k)), \quad (1.24)$$

де $w_i(k)=(w_{1i}(k),w_{2i}(k),\ldots,w_{h_1 i}(k))^T$ – вектор синаптичних ваг;

$\varphi_i(k)=(\varphi_{1i}(k),\varphi_{2i}(k),\ldots,\varphi_{h_1 i}(k))^T$ - вектор вейвлет-активаційних функцій.

За аналогією з (1.21) можна ввести процедуру

$$\begin{cases} w_i(k+1) = w_i(k) + \dfrac{\beta \tanh\left(\dfrac{e(k)}{\beta}\right) f_0'(u(k))\varphi_i(x_i(k))}{\gamma_i^{w_1}(k)}, \\ \gamma_i^{w_1}(k+1) = \alpha\gamma_i^{w_1}(k) + \left\|\varphi_i(x_i(k+1))\right\|^2. \end{cases} \quad (1.25)$$

На рис. 1.7. представлено архітектуру адаптивного подвійного вейвлет-нейрона з робастним блоком.

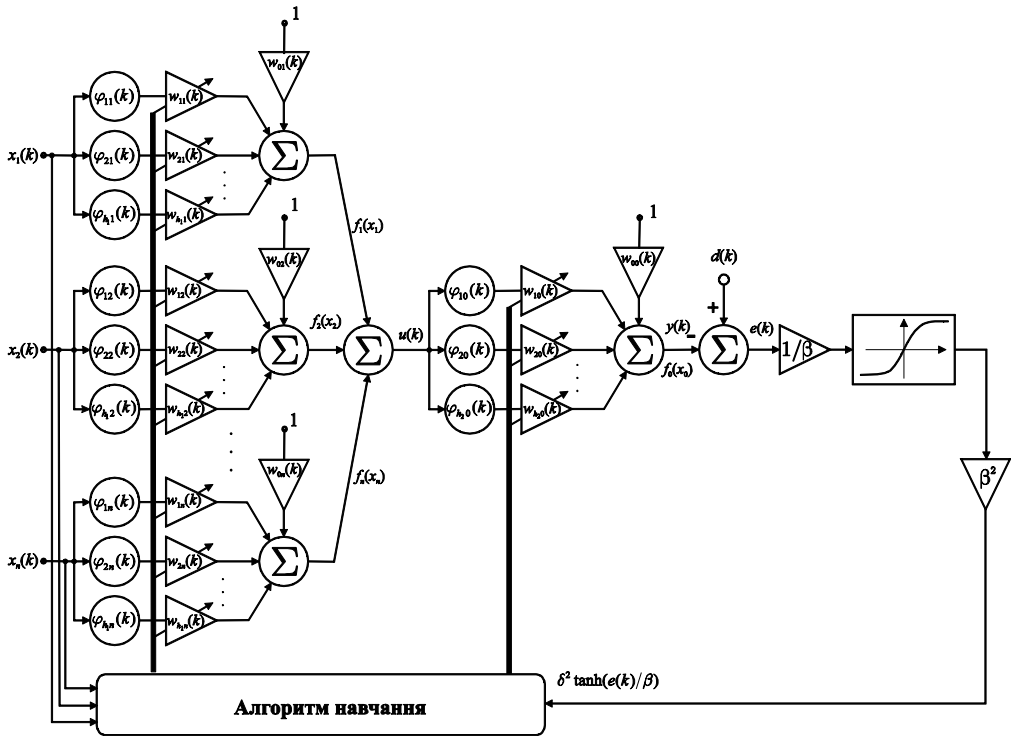


Рисунок 1.7 – Архітектура подвійного вейвлет-нейрона з робастним блоком

Запропонований підхід дозволяє обробляти сигнали за умов різного роду викидів, грубих похибок, неґаусівських збурювань із «важкими хвостами».

1.1.3 Робастний алгоритм навчання складеного адаптивного вейвлона

Найбільш відомими і популярними штучними нейронними мережами є багатошарові мережі із прямою передачею інформації типу тришарового

персептрона, елементарними вузлами яких є, так звані, P-нейрони з монотонними функціями активації.

Ефективність багатошарових мереж пояснюється їх універсальними апроксимуючими властивостями в поєднанні з відносно компактним поданням нелінійної системи, що моделюється. Це означає, що вони з успіхом можуть бути використані в задачах моделювання нелінійних систем, що описуються рівняннями типу

$$y(k) = F(x(k)) + \xi(k). \tag{1.26}$$

Звичайно передбачається також, що функція $F(\bullet)$ задана або на одиничному гіперкубі, або на ортотопі

$$x_i(k) \in [x_i^{\min}, x_i^{\max}],\ i = 1, 2, \ldots, n,$$

де $x_i^{\min}$, $x_i^{\max}$ – відомі нижня та верхня границі варіювання i-го вхідного сигналу.

Основним недоліком багатошарових мереж є низька швидкість їхнього навчання, що засновано на зворотному поширенні похибок, що унеможливлює їхнє використання в задачах реального часу.

Альтернативою багатошаровим штучним нейронним мережам є радіально-базисні нейронні мережі (РБНМ), що мають один прихований шар, який утворено, так званими, R-нейронами, при цьому навчання цих мереж реалізується на рівні вихідного шару, що представляє собою адаптивний лінійний асоціатор [42]–[45]. На відміну від P-нейронів, R-нейрони мають, як правило, дзвонувату функцію активації $f_j(x)$, аргументом якої є відстань (зазвичай в евклідовій метриці) між поточним значенням вхідного сигналу $x(k)$ та центром c_j j-го нейрона, тобто

$$\varphi_j(x(k)) = \varphi_j\left(\sum_{i=1}^{n}(x_i(k) - c_{ji})^2\right) = \varphi_j\left(\left\|x(k) - c_j\right\|^2\right). \quad (1.27)$$

Основною перевагою РБНМ є висока швидкість навчання у вихідному шарі, яка обумовлена тим, що параметри, які адаптуються, входять в опис мережі лінійно. У той же час залишається відкритою проблема розміщення центрів R-нейронів, невдале рішення якої веде до виникнення «прокльону розмірності». Застосування методів кластеризації, хоча і дозволяє зменшити розміри мережі, але виключає можливість роботи в реальному часі. Тут же слід зазначити, що в [46] описана градієнтна рекурентна процедура покомпонентного налаштування параметрів c_{ji}, однак вона характеризується низькою швидкістю збіжності.

Одночасно з нейронними мережами для обробки сигналів різної природи останнім часом досить часто використовується теорія вейвлет-перетворення [11], [12], що забезпечує компактне локальне подання сигналів як у частотній, так і в часовій області. На стику теорій штучних нейронних мереж і вейвлетів виникли вейвлет-нейронні мережі [19], [21], [25], [40], що підтвердили свою ефективність у задачах обробки нестаціонарних нелінійних сигналів і процесів.

Елементарними вузлами вейвлет-нейронних мереж є, так звані, радіальні вейвлони [47], активаційними функціями яких є парні вейвлети з аргументом у вигляді евклідової відстані між $x(k)$ і центром вейвлета c_j, при цьому кожна компонента відстані $\left|x_i(k) - c_{ji}\right|$ нормується на параметр ширини σ_{ji} так, що

$$\varphi_j(x(k)) = \varphi_j\left(\sum_{i=1}^{n}\left((x_i(k) - c_{ji})/\sigma_{ji}\right)^2\right), \quad (1.28)$$

де $\varphi_j(\bullet)$ – активаційна функція-вейвлет.

Рецепторними полями таких вейвлонів є гіпереліпсоїди з осями колінеарними координатним осям простору X.

З огляду на еквівалентність радіально-базисних штучних нейронних мереж і систем нечіткого виведення [48], [49], а також можливість використання як функції належності парних вейвлетів [50], [51], у рамках парадигми уніфікації [47] можна говорити про таку гібридну систему як адаптивний W-нейрон, що має можливість швидкого навчання радіально-базисних нейронних мереж, інтерпретовність систем нечіткого виведення та локальних властивостей вейвлетів.

Розглянемо синтез робастного алгоритму навчання адаптивного W-нейрона (вейвлона), що має регульований рівень нечутливості до різного роду викидів, грубих похибок, неґаусівських збурень та має підвищену швидкість збіжності й забезпечує поліпшені апроксимуючі властивості в порівнянні із традиційними системами обчислювального інтелекту.

Розглянемо двошарову архітектуру [52]–[54], що наведено на рис. 1.8 і співпадаючу із традиційною радіально-базисною нейронною мережею.

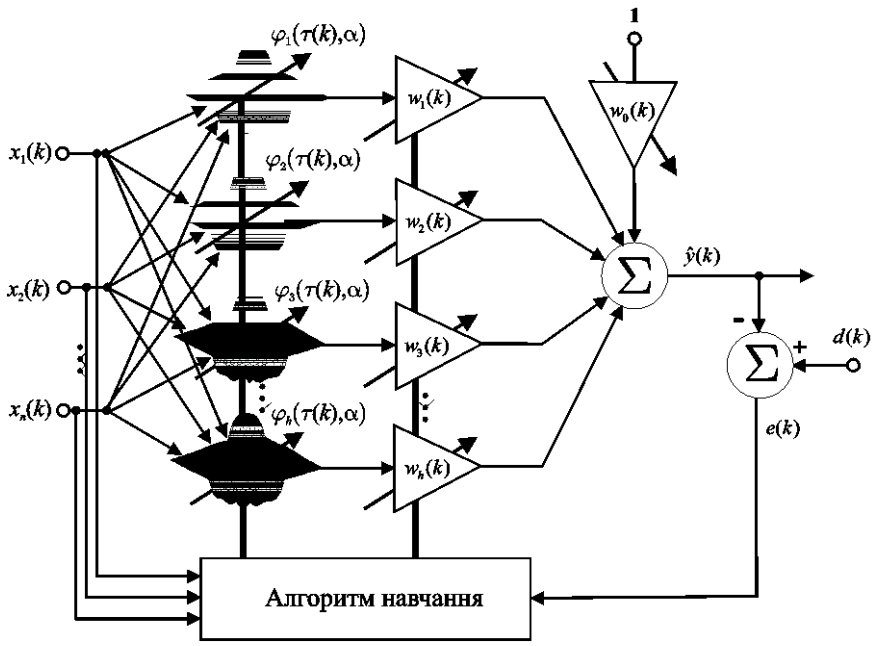


Рисунок 1.8 – Структура адаптивного W-нейрона

Нульовий шар архітектури є рецепторним і в поточний момент часу k на нього подається вхідний сигнал у формі вектора $x(k) = (x_1(k), x_2(k), \ldots, x_n(k))^T$.

Прихований шар на відміну від радіально-базисних мереж утворений не R-нейронами, а вейвлонами з активаційними функціями-вейвлетами вигляду

$$\varphi_j(x(k)) = \varphi_j\left((x(k)-c_j)^T Q_j^{-1}(k)(x(k)-c_j)\right),\ \ j=1,2,\ldots,h, \qquad (1.29)$$

у які замість параметрів зсуву σ_{ji} в (1.28) використовується матриця розтягань (ширин) Q_j, тобто використовується не евклідова метрика, а метрика Ітакури-Саїто [55]. Це приводить до того, що рецепторні поля – гіпереліпсоїди W-нейронів (1.29) можуть мати довільну орієнтацію щодо координатних осей простору X, що розширює функціональні можливості адаптивного W-нейрона.

Ґрунтуючись на вейвлет-функції «Mexican Hat» [14], введемо нову адаптивну активаційну функцію у структуру складеного адаптивного вейвлона, що має вигляд

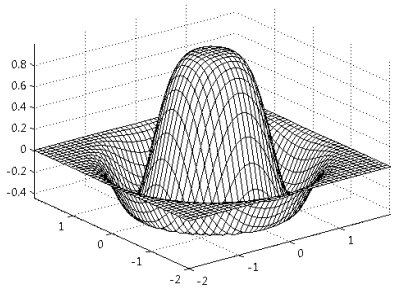
$$\varphi_j(\tau_j(x(k))) = (1-\alpha_j\tau_j{}^2)\exp\left(-\frac{\tau_j{}^2}{2}\right), \qquad (1.30)$$

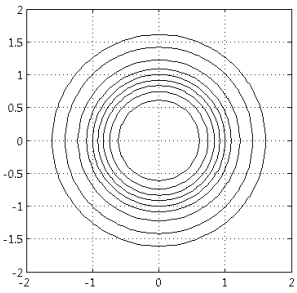
де $\tau_j(x(k)) = \tau_j\left((x(k)-c_j(k))^T Q_j^{-1}(k)(x(k)-c_j(k)),\alpha_j\right)$;

α_j – адаптивний параметр, що налаштувується ($0 \le \alpha \le 1$).

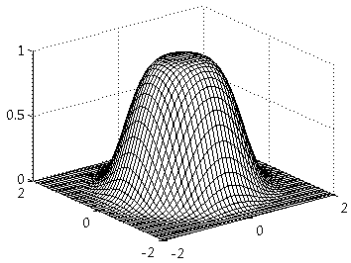
Параметр α_j, що уточнюється, дозволяє налаштовувати форму активаційної функції в процесі навчання складеного адаптивного вейвлона, при цьому якщо $\alpha = 0$ одержуємо ґаусову функцію активації, при $\alpha = 1$ одержуємо вейвлет-функцию «Mexican Hat», а при $0 < \alpha < 1$ – гібридну функцію активації.

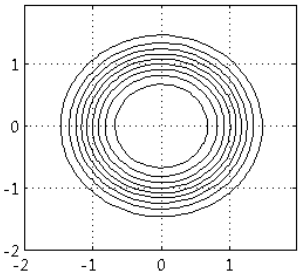
На рис. 1.9 наведено активаційні функції вейвлонів (1.29) при різних матрицях Q_j і параметрі α_j.



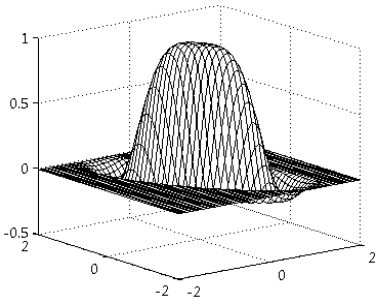


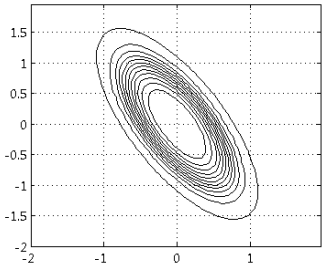
а) $Q^{-1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \alpha = 1$



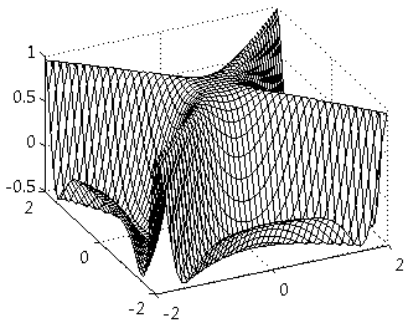


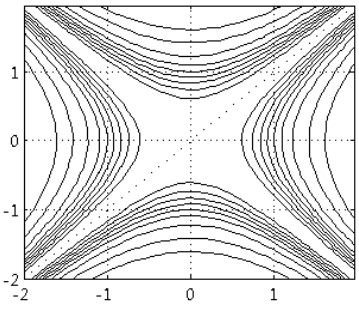
б) $Q^{-1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \alpha = 0$



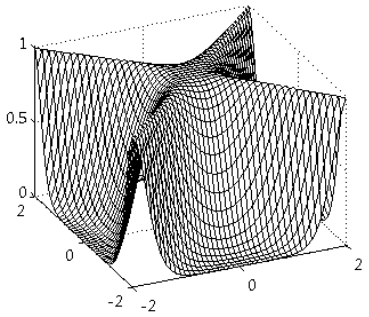
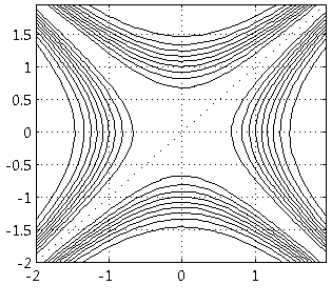


в) $Q^{-1} = \begin{pmatrix} 4 & 3 \\ 1 & 2 \end{pmatrix}, \alpha = 0.5$

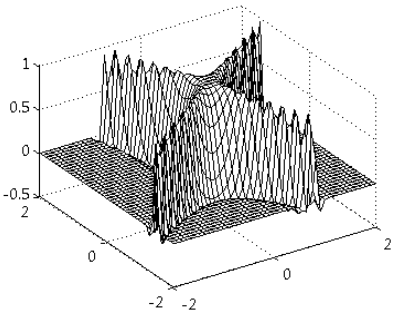
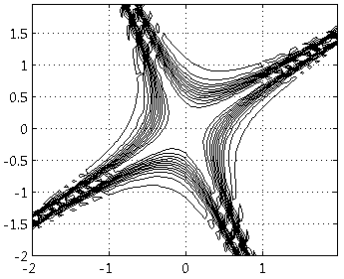




г) $Q^{-1} = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}, \alpha = 1$

д) $Q^{-1} = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}, \alpha = 0$

е) $Q^{-1} = \begin{pmatrix} -4 & 3 \\ 1 & 2 \end{pmatrix}, \alpha = 0.5$

Рисунок 1.9 – Активаційні функції W-нейронів при різних матрицях Q_j^{-1} і параметрі α

І, нарешті, вихідний шар є звичайний адаптивний лінійний асоціатор із синаптичними вагами w_j, що налаштовуються

$$\begin{aligned} \hat{y}(k) &= w_0(k) + \sum_{j=1}^{h} w_j(k)\varphi\Big((x(k)-c_j)^T Q_j^{-1}(k)(x(k)-c_j), \alpha_j(k)\Big) = \\ &= w^T(k)\varphi(\tau_j(x(k))), \end{aligned} \tag{1.31}$$

де $\varphi_0(\tau_j(x(k))) \equiv 1$;

$w(k) = (w_0(k), w_1(k), \ldots, w_h(k))^T$;

$\varphi(\tau_j(x(k))) = (1, \varphi_1(\tau_j(x(k))), \ \ldots, \varphi_h(\tau_j(x(k))))^T$.

Параметрами архітектури, що адаптуються та підлягають визначенню в процесі навчання, будемо вважати $h+1$ синаптичних ваг w_j, h $(n\times 1)$-векторів c_j і h $(n\times n)$-матриць Q_j^{-1}.

Усього ж така мережа містить h $(1+n+n^2)+1$ параметрів, що налаштовуються.

Розглянемо робастний алгоритм навчання адаптивного вейвлона. Введемо до розгляду похибку навчання адаптивного W-нейрона

$$e(k)=y(k)-\hat{y}(k)=y(k)-w^T(k)\varphi(k) \tag{1.32}$$

і робастний критерій оптимізації (1.18).

Далі розглянемо процес синтезу алгоритмів навчання [56]–[58]. Для налаштування синаптичних ваг і параметрів W-нейрона (векторів c_j і матриць Q_j^{-1}) використовуємо градієнтну мінімізацію критерію (1.18), при цьому на відміну від покомпонентного навчання, що розглянуто в [46], будемо проводити уточнення у векторно-матричній формі, що, по-перше, простіше з обчислювальної точки зору, а, по-друге, дозволяє оптимізувати процес навчання за швидкодією.

У загальному випадку алгоритм навчання може бути записаний у вигляді

$$\begin{cases} w(k+1)=w(k)-\eta_w\nabla_w E(k), \\ c_j(k+1)=c_j(k)-\eta_{c_j}\nabla_{c_j}E(k),\ \ j=1,2,\ldots,h, \\ Q_j^{-1}(k+1)=Q_j^{-1}(k)-\eta_{Q_j^{-1}}\left\{\partial E(k)\big/\partial Q_j^{-1}\right\},\ \ j=1,2,\ldots,h, \\ \alpha_j(k+1)=\alpha_j(k)-\eta_\alpha\nabla_{\alpha_j}E(k),\ \ j=1,2,\ldots,h, \end{cases} \tag{1.33}$$

де $\nabla_{c_j}E$, $\nabla_{\alpha_j}E$ – $(n\times 1)$-вектори-градієнти критерію (1.18) по c_j та α_j відповідно;

$\left\{\partial E(k)/\partial Q_j^{-1}\right\}$ – $(n\times n)$-матриця, яку утворено частковими похідними E по компонентах Q_j^{-1};

$\eta_{c_j}, \eta_{Q_j^{-1}}, \eta_{\alpha_j}$ – параметри кроку алгоритму навчання.

Для довільно взятого вейвлета $\varphi((x(k)-c_j)^T Q_j^{-1}(k)(x(k)-c_j),\alpha_j)$ можна записати

$$
\begin{cases}
\dfrac{\partial E(k)}{\partial w_j} = -\beta\tanh\left(e(k)/\beta\right)\varphi_j((x(k)-c_j(k))^T Q_j^{-1}(k)(x(k)-c_j(k)),\alpha_j) = \\
\qquad = -\beta\tanh\left(e(k)/\beta\right)J_w(k), \\
\nabla_{c_j}E(k) = \beta\tanh\left(e(k)/\beta\right)w_j(k)\varphi_j'((x(k)-c_j(k))^T Q_j^{-1}(k)(x(k)-c_j(k)),\alpha_j)\cdot \\
\qquad \cdot Q_j^{-1}(k)(x(k)-c_j(k)) = \beta\tanh\left(e(k)/\beta\right)J_{c_j}(k), \\
\left\{\dfrac{\partial E(k)}{\partial Q_j^{-1}}\right\} = -\beta\tanh\left(e(k)/\beta\right)w_j(k)\cdot \\
\qquad\qquad \cdot\varphi_j'((x(k)-c_j(k))^T Q_j^{-1}(k)(x(k)-c_j(k)),\alpha_j)\cdot \\
\qquad\qquad \cdot(x(k)-c_j(k))(x(k)-c_j(k))^T = -\beta\tanh\left(e(k)/\beta\right)J_{Q_j^{-1}}(k), \\
\dfrac{\partial E(k)}{\partial \alpha_j} = -\beta\tanh\left(e(k)/\beta\right)w_j(k)\cdot \\
\qquad \cdot\dfrac{\partial\varphi_j((x(k)-c_j(k))^T Q_j^{-1}(k)(x(k)-c_j(k)),\alpha_j)}{\partial\alpha_j} = \\
\qquad = -\beta\tanh\left(e(k)/\beta\right)J_{\alpha_j}(k),
\end{cases}
\tag{1.34}
$$

де $\varphi_j'(\bullet)$ – похідна j-го вейвлета по аргументу $(x(k)-c_j(k))^T Q_j^{-1}(k)(x(k)-c_j(k))$.

Тоді алгоритм навчання прихованого шару W-нейрона з урахуванням (1.34) набуває вигляду

$$\begin{cases} w(k+1) = w(k) + \eta_w \beta \tanh\left(e(k)/\beta\right) J_w(k), \\ c_j(k+1) = c_j(k) - \eta_{c_j} \beta \tanh\left(e(k)/\beta\right) J_{c_j}(k), \\ Q_j^{-1}(k+1) = Q_j^{-1}(k) + \eta_{Q_j^{-1}} \beta \tanh\left(e(k)/\beta\right) J_{Q_j^{-1}}(k), \\ \alpha_j(k+1) = \alpha_j(k) + \eta_{\alpha_j} \beta \tanh\left(e(k)/\beta\right) J_{\alpha_j}(k), \end{cases} \tag{1.35}$$

при цьому швидкість збіжності до оптимальних значень c_j, α_j і Q_j^{-1} повністю визначається параметрами кроку η_{c_j}, η_{α_j} та $\eta_{Q_j^{-1}}$. Підвищення швидкості збіжності може бути досягнуте шляхом використання більш складних, ніж градієнтні, процедур типу Хартлі або Марквардта, які для першого співвідношення (1.35) можуть бути записані в узагальненій формі [59]

$$w(k+1) = w(k) - \lambda_w (J_w(k) J_w^T(k) + \eta_w I)^{-1} J_w(k) \beta \tanh\left(e(k)/\beta\right), \tag{1.36}$$

де I – $(n \times n)$-одинична матриця;

λ_w – додатний параметр;

η_w – параметр регуляризації.

Скориставшися лемою обернення матриць, після нескладних перетворень можна одержати простий і ефективний алгоритм навчання параметрів у вигляді

$$\begin{cases} w(k+1) = w(k) + \lambda_w \dfrac{\beta \tanh\left(e(k)/\beta\right) J_w(k)}{\eta_w + \left\| J_w(k) \right\|^2}, \\ c_j(k+1) = c_j(k) - \lambda_c \dfrac{\beta \tanh\left(e(k)/\beta\right) J_{c_j}(k)}{\eta_c + \left\| J_{c_j}(k) \right\|^2}, \\ Q_j^{-1}(k+1) = Q_j^{-1}(k) + \lambda_{Q_j^{-1}} \dfrac{\beta \tanh\left(e(k)/\beta\right) J_{Q_j^{-1}}(k)}{\eta_{Q_j^{-1}} + Tr(J_{Q_j^{-1}}^T(k) J_{Q_j^{-1}}(k))}, \\ \alpha_j(k+1) = \alpha_j(k) + \lambda_\alpha \dfrac{\beta \tanh\left(e(k)/\beta\right) J_{\alpha_j}(k)}{\eta_\alpha + \left\| J_{\alpha_j}(k) \right\|^2}. \end{cases} \tag{1.37}$$

Для надання процесу навчання додаткових згладжуючих властивостей та використовуючи підхід, що запропоновано в [60], можна ввести модифіковану процедуру навчання [56, 57]:

$$\begin{cases} w(k+1) = w(k) + \lambda_w \dfrac{\tanh\left(e(k)/\beta\right) J_w(k)}{\eta_w(k)}, \\ \eta_w(k+1) = \alpha_w \eta_w(k) + \left\| J_w(k+1) \right\|^2, \\ c_j(k+1) = c_j(k) - \lambda_{c_j} \dfrac{\tanh\left(e(k)/\beta\right) J_{c_j}(k)}{\eta_{c_j}(k)}, \\ \eta_{c_j}(k+1) = \alpha_c \eta_{c_j}(k) + \left\| J_{c_j}(k+1) \right\|^2, \\ Q_j^{-1}(k+1) = Q_j^{-1}(k) + \lambda_{Q_j^{-1}} \dfrac{\tanh\left(e(k)/\beta\right) J_{Q_j^{-1}}(k)}{\eta_{Q_j^{-1}}(k)}, \\ \eta_{Q_j^{-1}}(k+1) = \alpha_{Q_j^{-1}} \eta_{Q_j^{-1}}(k) + Tr\left(J_{Q_j^{-1}}^T(k+1) J_{Q_j^{-1}}(k+1) \right), \\ \alpha_j(k+1) = \alpha_j(k) + \lambda_{\alpha_j} \dfrac{\tanh\left(e(k)/\beta\right) J_{\alpha_j}(k)}{\eta_{\alpha_j}(k)}, \\ \eta_{\alpha_j}(k+1) = \gamma_\alpha \eta_{\alpha_j}(k) + \left\| J_{\alpha_j}(k+1) \right\|^2 \end{cases} \tag{1.38}$$

(тут $0 \le \alpha_w \le 1, 0 \le \alpha_{c_j} \le 1, 0 \le \alpha_{Q_j^{-1}} \le 1, 0 \le \alpha_{\alpha_j} \le 1$ – параметри зважування застарілої інформації), що є нелінійним гібридом алгоритмів Качмажа–Уідроу–Хоффа та Ґудвина–Ремеджа–Кейнеса та мають як слідкуючі, так і фільтруючі властивості.

1.1.4 Робастний алгоритм навчання адаптивної нейро-фаззі системи та гібридних вейвлет-нейро-фаззі систем

У цей час все більше поширення одержують вейвлет-нейро-фаззі технології обробки нестаціонарних процесів за умов невизначеності, що забруднені викидами з невідомим законом розподілу. Такі методи дозволяють вирішувати широкий клас задач обробки інформації й, насамперед, задачі

сегментації, прогнозування, емуляції, часових рядів довільної природи за умов структурної та параметричної невизначеності. Найбільш відомою і популярною є адаптивна нейро-фаззі система Такаґі-Суґено-Канґа [61], відомі алгоритми навчання якої не дозволяють обробляти сигнали, що забруднені завадами, в реальному часі. Таким чином, актуальним є синтез робастних алгоритмів навчання, а також гібридних архітектур та їхніх алгоритмів навчання на основі об'єднання вейвлет-нейро-фаззі методів, які надалі будемо іменувати вейвлет-нейро-фаззі системами [19], [47]. Такі системи з успіхом можуть бути використані в задачах прогнозування і сегментації стохастичних і хаотичних сигналів і послідовностей зі складним нелінійним трендом і нестаціонарними параметрами.

Розглянемо три модифікації адаптивних вейвлет-нейро-фаззі систем:

– адаптивна нейро-фаззі система Такаґі–Суґено–Канґа з робастним алгоритмом навчання – відома стандартна адаптивна нейро-фаззі система, в яку введено робастний шостий шар (див. рис. 1.10, де ψ_{ji} – стандартні функції належності) [62], [63];

– адаптивна гібридна вейвлет-нейро-фаззі система з лінійним консеквентом [64]–[66] і робастним алгоритмом навчання всіх її параметрів. Така архітектура є модифікацією нейро-фаззі системи Такаґі–Суґено–Канґа, у якій в першому шарі введені замість звичайних функцій належності вейвлет-функції активації-належності (див. рис. 1.10, де ψ_{ji} – вейвлет-функції активації-належності);

– адаптивна гібридна вейвлет-нейро-фаззі система з консеквентом на основі адаптивного W-нейрона з робастним алгоритмом навчання всіх її параметрів (див. рис. 1.11) [67], [68]. Ця архітектура є модифікацією відомої адаптивної нейро-фаззі системи та структури, представленої у [69], в якій у консеквенті замість лінійних функцій використовуються W-нейрони.

Оскільки архітектура стандартної адаптивної нейро-фаззі системи розглянута докладно в багатьох статтях та монографіях, далі наведемо лише опис адаптивних гібридних вейвлет-нейро-фаззі систем.

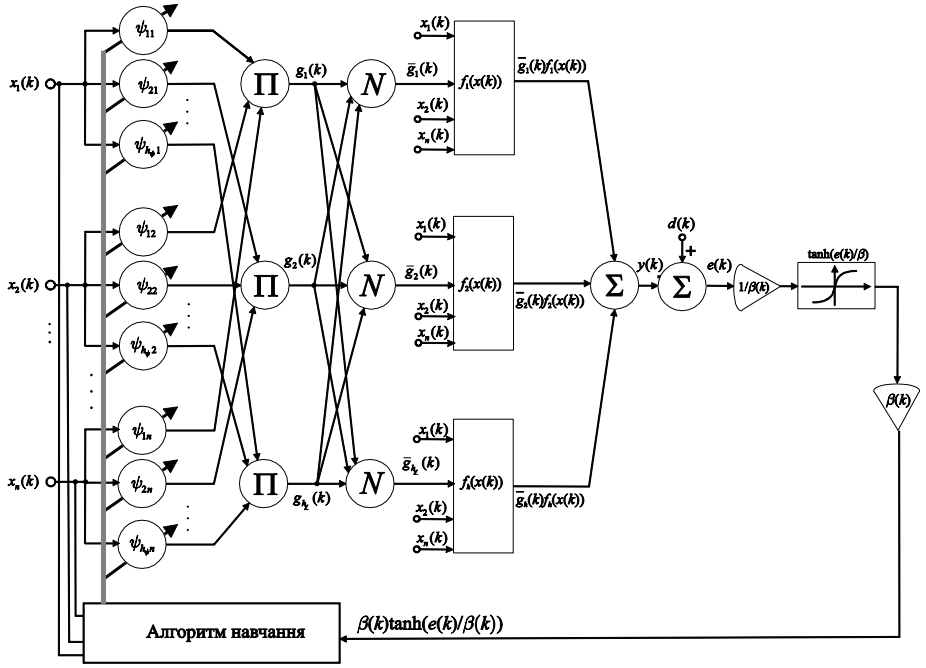


Рисунок 1.10 – Адаптивна робастна гібридна вейвлет-нейро-фаззі система з лінійним консеквентом

Перший прихований шар утворений не традиційними додатними функціями належності, а набором з $h_\psi n$ вейвлетів з $2h_\psi n$ параметрами центра c_{ji}^{ψ} і ширини σ_{ji}, що налаштовуються.

Тут можна відзначити, що коливальний характер вейвлет-функцій не суперечить уніполярності функцій належності, оскільки від'ємні значення $\psi_{ji}(k)$ можуть трактуватися в сенсі малих рівнів належності або неналежності [51].

У робастній адаптивній нейро-фаззі системі будемо використовувати стандартні функції належності (наприклад, функції Ґауса), а в робастній вейвлет-нейро-фаззі системі, будемо використовувати запропоновану в [70]–[72] адаптивну активаційну функцію, що має вигляд

$$\psi_j(t_j(x(k))) = (1-\alpha_j t_j^2(x(k)))\exp\left(-t_j^2(x(k))/2\right), \qquad (1.39)$$

$$\varphi_j(\tau_j(x(k))) = (1-\alpha_j\tau_j^2(x(k)))\exp\left(-\tau_j^2(x(k))/2\right), \qquad (1.40)$$

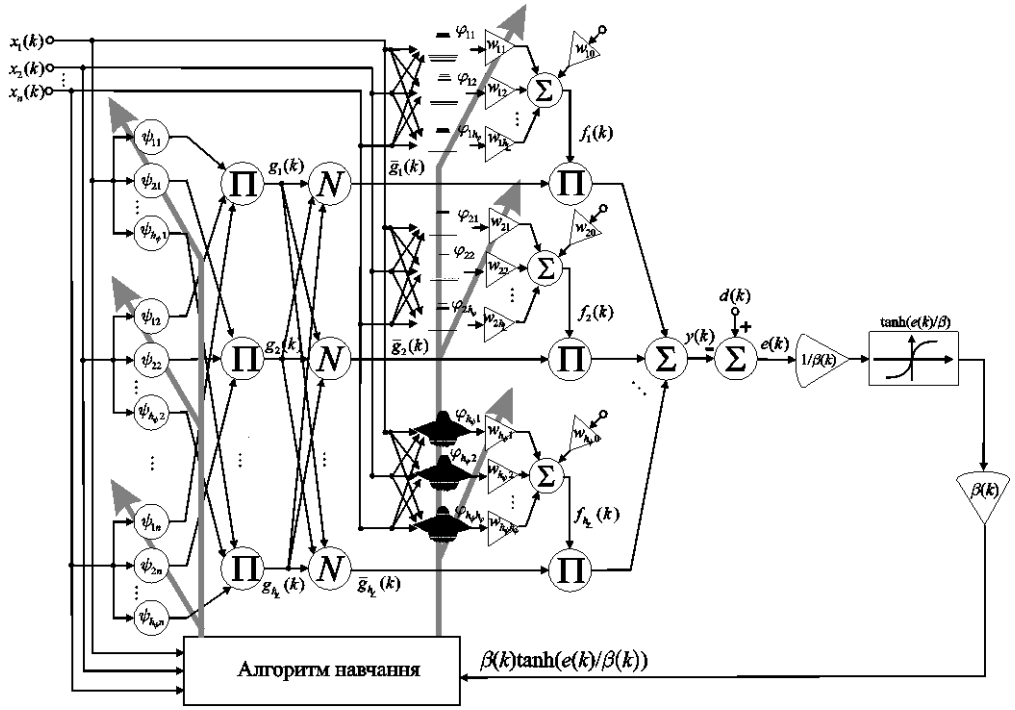


Рисунок 1.11 – Адаптивна робастна гібридна вейвлет-нейро-фаззі система з консеквентом на основі адаптивного W -нейрона

де α_j – адаптивний параметр ($0 \le \alpha \le 1$);

$$t_j(x(k)) = \left((x(k) - c_j^{\psi}(k))\sigma_j^{-1}(k)\right);$$

$$\tau_j(x(k)) = \left((x(k) - c_j^{\varphi}(k))^T Q_j^{-1}(k)(x(k) - c_j^{\varphi}(k))\right).$$

Другий прихований шар реалізує операцію, аналогічну обчисленню нечіткої T -норми

$$g_j(k) = \prod_{i=1}^{n}\psi_{ji}(k),\ j = 1,2,\ldots,h_{\psi}\ , \qquad (1.41)$$

після чого в третьому прихованому шарі відбувається нормалізація

$$\bar{g}_j(k) = \prod_{i=1}^{n}\psi_{ji}(k)\Big/\sum_{j=1}^{h_{\psi}}\prod_{i=1}^{n}\psi_{ji}(k), \qquad (1.42)$$

що забезпечує виконання умови

$$\sum_{j=1}^{h_\psi} \bar{g}_j(k) = 1. \qquad (1.43)$$

Четвертий прихований шар реалізує операцію аналогічну обчисленню консеквента у системах нечіткого виведення (лінійний консеквент), а в третій вейвлет-нейро-фаззі системі замість лінійних вихідних функцій $f_j(X(k))$ використовується структура адаптивного W-нейрона [72], що дозволяє поліпшити апроксимуючі властивості системи

$$f_j(X(k)) = w_{j0}(k) + \sum_{m=1}^{h_\varphi} w_{jm}(k)\varphi_{jm}(\tau_{jm}(k)) = w_j^T(k)\varphi_j(\tau_j(k)), \qquad (1.44)$$

де $\varphi_{j0} = 1$;

$\varphi_{jm}(\tau_{jm}(k)) = \varphi_{jm}((X(k) - c_{jm}^{\varphi}(k))^T Q_{jm}^{-1}(k)(X(k) - c_{jm}^{\varphi}(k)), \alpha_{jm}(k))$;

$w_j(k) = (w_{j0}(k), \ldots, w_{jh_\varphi}(k))^T$, $\varphi_j(\tau_j(k)) = (1, \varphi_{j1}(\tau_{j1}(k)), \ldots, \varphi_{jh_\varphi}(\tau_{jh_\varphi}(k)))$.

У цьому випадку в четвертому шарі обчислюються сигнали вигляду

$$\bar{g}_j(k)\left(w_{j0}(k) + \sum_{m=1}^{h_\varphi} w_{jm}(k)\varphi_{jm}(\tau_{jm}(k)) \right) = \bar{g}_j(k) w_j^T(k)\varphi_j(\tau_j(k)), \qquad (1.45)$$

де $h_\varphi(n+1)$ параметрів w_{jm}, $j = 1, 2, \ldots, h_\varphi$, $m = 0, 1, 2, \ldots, h_\psi$ підлягають визначенню.

У п’ятому вихідному шарі обчислюється вихідний сигнал системи, який можна представити у векторній формі

$$y(k) = \bar{g}^T(k) f(X(k)), \qquad (1.46)$$

де $f(X(k)) = (w_1^T(k)\varphi_1(\tau_1(k)),\ldots,w_{h_\psi}^T(k)\varphi_{h_\psi}(\tau_{h_\psi}(k)))$.

Вводячи позначення $F(X(k)) = (\bar{g}_1(k)\varphi_1(\tau_1(k)),\ldots,\bar{g}_{h_\psi}(k)\varphi_{h_\psi}(\tau_{h_\psi}(k)))^T$ і $w = (w_1,\ldots,w_{h_\psi})^T$, вихід такої архітектури можна записати в компактній формі

$$y(k) = w^T(k)F(X(k)). \tag{1.47}$$

Для налаштування параметрів прихованих шарів у запропонованих архітектурах використовується алгоритм зворотного поширення похибок, що заснований на градієнтній оптимізації робастного критерію

$$E(k) = \beta^2 \ln(\cosh(e(k)/\beta)), \tag{1.48}$$

де $e(k) = d(k) - y(k) = d(k) - w^T(k)F(X(k))$.

Кінцева форма алгоритму навчання параметрів вейвлет-функцій належності має вигляд

$$\begin{cases} \Psi(k+1) = \Psi(k) + \lambda\left(\dfrac{J^\Psi(k)e(k)}{\eta(k)}\right), \\ \eta(k+1) = \alpha\eta(k) + \left\|J^\Psi(k+1)\right\|^2, \end{cases} \tag{1.49}$$

де $\Psi(k) = (c_{11}^\psi(k),\sigma_{11}^{-1}(k),c_{21}^\psi(k),\sigma_{21}^{-1}(k),\ldots,c_{h_\psi n}^\psi(k),\sigma_{h_\psi n}^{-1}(k))^T$ – $(2h_\psi n\times 1)$-вектор параметрів, що налаштовуються;

η – скалярний регуляризуючий параметр;

λ – додатний скалярний коефіцієнт підсилення;

$$J^{\Psi}(k)=\left(\beta\frac{\partial y(k)}{\partial c_{11}^{\psi}},\beta\frac{\partial y(k)}{\partial \sigma_{11}^{-1}},\ldots,\beta\frac{\partial y(k)}{\partial c_{h_\psi n}^{\psi}},\beta\frac{\partial y(k)}{\partial \sigma_{h_\psi n}^{-1}}\right) \quad – \quad \text{вектор-градієнт,}$$

компоненти якого обчислюються за допомогою співвідношень

$$\begin{cases}\dfrac{\partial y(k)}{\partial c_{ji}^{\psi}}=f_j(X(k))\bar{g}_j(k)(1-\bar{g}_j(k))\dfrac{\partial \psi_{ji}(x_i(k))}{\partial c_{ji}^{\psi}}\cdot\dfrac{1}{\psi_{ji}(x_i(k))},\\ \dfrac{\partial y(k)}{\partial \sigma_{ji}^{-1}}=f_j(X(k))\bar{g}_j(k)(1-\bar{g}_j(k))\dfrac{\partial \psi_{ji}(x_i(k))}{\partial \sigma_{ji}^{-1}}\cdot\dfrac{1}{\psi_{ji}(x_i(k))}.\end{cases} \tag{1.50}$$

Далі розглянемо алгоритм навчання параметрів четвертого шару. У перших двох системах для навчання параметрів можуть бути використані стандартні градієнтні методи навчання. Для налаштування параметрів W-нейрона (векторів w_j, c_j, матриць Q_j^{-1}, параметра α_j) будемо використовувати градієнтну мінімізацію критерію (1.49). Таким чином, остаточно алгоритм навчання параметрів W-нейрона прихованого шару в оптимальному по швидкодії варіанті може бути записаний у вигляді системи

$$
\begin{cases}
w_j(k+1) = w_j(k) + \lambda_w \dfrac{\tanh(e(k)/\beta) J_w^{\varphi}(k)}{\eta_w(k)}, \\
\eta_w(k+1) = \gamma_w \eta_w(k) + \left\| J_w^{\varphi}(k+1) \right\|^2, \\
c_j^{\varphi}(k+1) = c_j^{\varphi}(k) + \lambda_{c_j^{\varphi}} \dfrac{\tanh(e(k)/\beta) J_{c_j^{\varphi}}^{\varphi}(k)}{\eta_{c_j^{\varphi}}(k)}, \\
\eta_{c_j^{\varphi}}(k+1) = \gamma_{c_j^{\varphi}} \eta_{c_j^{\varphi}}(k) + \left\| J_{c_j^{\varphi}}^{\varphi}(k+1) \right\|^2, \\
Q_j^{-1}(k+1) = Q_j^{-1}(k) + \lambda_{Q_j^{-1}} \dfrac{\tanh(e(k)/\beta) J_{Q_j^{-1}}^{\varphi}(k)}{\eta_{Q_j^{-1}}(k)}, \\
\eta_{Q_j^{-1}}(k+1) = \gamma_{Q_j^{-1}} \eta_{Q_j^{-1}}(k) + Tr((J_{Q_j^{-1}}^{\varphi}(k+1))^T J_{Q_j^{-1}}^{\varphi}(k+1)), \\
\alpha_j(k+1) = \alpha_j(k) + \lambda_{\alpha_j} \dfrac{\tanh(e(k)/\beta) J_{\alpha_j}^{\varphi}(k)}{\eta_{\alpha_j}(k)}, \\
\eta_{\alpha_j}(k+1) = \gamma_{\alpha_j} \eta_{\alpha_j}(k) + \left\| J_{\alpha_j}^{\varphi}(k+1) \right\|^2,
\end{cases}
\tag{1.51}
$$

де $0 \le \gamma_w, \gamma_{c_j^{\varphi}}, \gamma_{Q_j^{-1}}, \gamma_{\alpha_j} \le 1$ – параметри зважування застарілої інформації;

$$
J_w^{\varphi}(k) = \beta_w (1 - \alpha_j(k) \tau_j^2(x(k))) \exp(-\tau_j^2(x(k))/2);
$$

$$
J_{c_j}^{\varphi}(k) = \beta_c w_j(k) Q_j^{-1}(k)(x(k) - c_j^{\varphi}(k)) \times
$$

$$
\times (\alpha_j(k) \tau_j^3(x(k)) - (2\alpha_j(k) + 1) \tau_j(x(k))) \exp(-\tau_j^2(x(k))/2);
$$

$$
J_{Q_j^{-1}}^{\varphi}(k) = \beta_Q w_j(k)(x(k) - c_j^{\varphi}(k))(x(k) - c_j^{\varphi}(k))^T \times
$$

$$
\times (\alpha_j(k) \tau_j^3(x(k)) - (2\alpha_j(k) + 1) \tau_j(x(k))) \exp(-\tau_j^2(x(k))/2);
$$

$$
J_{\alpha_j}^{\varphi}(k) = \beta_{\alpha} w_j(k) \tau_j^2(x(k)) \exp(-\tau_j^2(x(k))/2).
$$

Розглянуті робастні архітектури адаптивних нейро-фаззі та вейвлет-нейро-фаззі систем забезпечують високу якість апроксимації сигналів, що забруднені завадами та викидами невідомої природи. Модифікації нейро-фаззі систем дозволяють досягти підвищення швидкості обробки сигналів у реальному часі.

Такі вейвлет-нейро-фаззі системи можуть бути використані для вирішення задач діагностики, прогнозування, емуляції та ідентифікації нестаціонарних процесів у реальних системах.

Експериментальне дослідження розробленого робастного алгоритму навчання адаптивного вейвлона проводилося на основі сигналу, що забруднено інтенсивними викидами. Сигнал був отриманий, використовуючи нелінійний динамічний об'єкт Нарендри [73], чий вихідний сигнал був штучно забруднений випадковими викидами з розподілом Коші, що має вигляд

$$F_X^{-1}(x) = x_0 + \gamma tg[\pi(x - 0.5)], \tag{1.52}$$

де x_0 – параметр локалізації;

γ – параметр масштабу $(\gamma > 0)$

x – носій ($x \in (-\infty; +\infty)$).

Нелінійний динамічний об’єкт було сгенеровано рівнянням

$$y(k+1) = 0.3y(k) + 0.6y(k-1) + f(u(k)), \tag{1.53}$$

де $f(u(k)) = 0.6\sin(u(k)) + 0.3\sin(3u(k)) + 0.1\sin(5u(k))$ та $u(k) = \sin(2k/250)$.

Значення $x(k-3), x(k-2), x(k-1), x(k)$ були використані для емуляції $x(k+1)$. У реальному часі адаптивний W-нейрон був навчений процедурою на 20000 ітераціях.

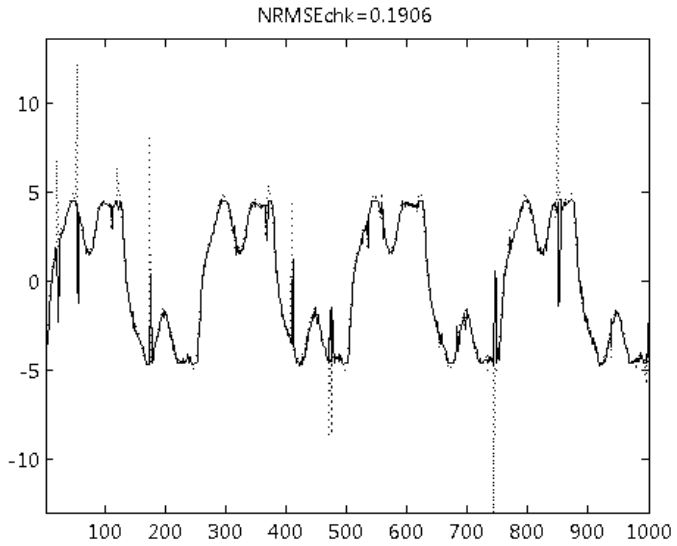
Рис. 1.12,а показує результати емуляції сигналу, що забруднено завадами. Рис. 1.12,б показує сегмент процесу навчання: можна побачити, що викид з великою амплітудою, що виникає на початку вибірці, не вплинув на алгоритм навчання.

Порівняння результатів прогнозування на основі робастного алгоритму навчання проводилося з результатами прогнозування на основі градієнтного алгоритму та алгоритму на основі рекурентного методу найменших квадратів, де

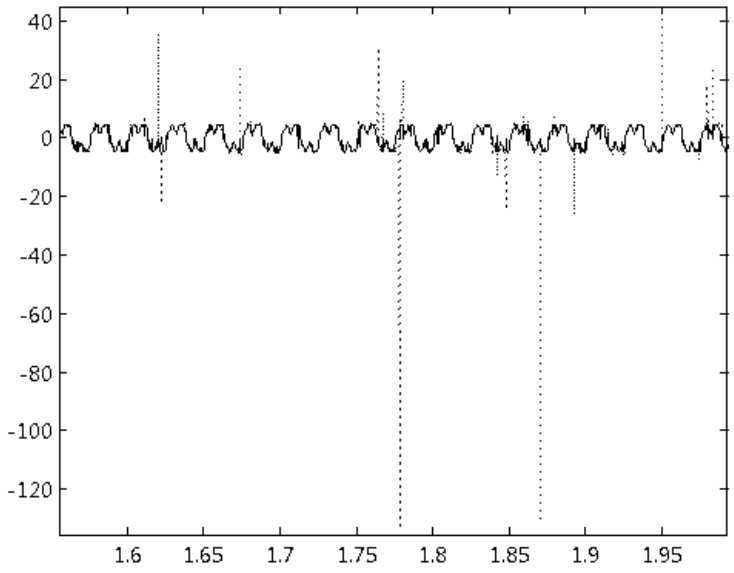
структура мережі, і кількість параметрів, що настроювалися, були однаковими. Результати прогнозування наведені у таблиці 1.1.

При навчанні адаптивного W-нейрона градієнтним алгоритмом перший же викид на початку вибірки сильно вплинув на процес навчання і як результат – велика похибка емуляції. При навчанні адаптивного вейвлона рекурентним методом найменших квадратів при першому викиді відбувся так званий «вибух параметрів» коваріаційної матриці і як результат – неможливість емуляції сигналів, що забруднені аномальними викидами.

У такий спосіб видно, що запропонований робастний алгоритм навчання адаптивного вейвлона дозволяє обробляти сигнали за умов істотного забруднення викидами.



а)



б)

Рисунок 1.12 – Результати емуляції

Таблиця 1.1 – Результати прогнозування

Нейронна мережа / Алгоритм навчання	NRMSE
Адаптивний W-нейрон / Запропонований робастний алгоритм навчання	0.1906
Адаптивний W-нейрон / Градієнтний алгоритм навчання	1.1242
Адаптивний W-нейрон / Рекурентний метод найменших квадратів	∞

Наступний експеримент довів ефективність розробленої адаптивної гібридної вейвлет-нейро-фаззі системи на адаптивних W-нейронах. Типову схему задачі емуляції наведено на рис. 1.13. Входами адаптивної гібридної вейвлет-нейро-фаззі системи є зовнішні вхідні сигнали об'єкту та їхні затримані значення, а також затримані значення сигналів об'єкта емуляції на виході.

Рисунок 1.13 – Типова схема емуляції

В якості об’єкта емуляції був також взятий динамічний нелінійний об'єкт іншого порядку [73], що описується різницевим рівнянням

$$y(k)=\theta(y(k-1),y(k-2),y(k-3),u(k),u(k-1)), \tag{1.54}$$

де

$$\theta(x_1,x_2,x_3,x_4,x_5)=\frac{x_1x_2x_3x_5(x_3-1)+x_4}{1+x_3^2+x_2^2}. \tag{1.55}$$

Як керуючий сигнал на вході використовується послідовність

$$u(k)=\begin{cases}\sin(\pi k/25),\ для\ k<250,\\ 1.0,\ для\ 250\le k<500,\\ -1.0,\ для\ 500\le k<750,\\ 0.3\sin(\pi k/25)+0.1\sin(\pi k/32)+\\ \quad+0.6\sin(\pi k/10),\ для\ 750\le k\le 1000.\end{cases} \tag{1.56}$$

Для емуляції динамічного об'єкту застосовувалася запропонована гібридна вейвлет-нейро-фаззі система на W-нейронах з кількістю входів $n=5$. Кількість вейвлет-активаційних функцій у прихованому шарі і їхні початкові параметри

були отримані за допомогою методу субтрактивної кластеризації [74]. Застосування цього методу дозволяє отримати не тільки вихідну матрицю координат центрів кластерів, але також вектор, компоненти якого визначають діапазон впливу центра кластера. Початкові значення синаптичних ваг були прийняті нульовими.

В якості критерію прогнозу було використано середньоквадратичну похибку (RMSE)

$$RMSE = \frac{1}{N}\sum_{k=1}^{N}(y(k) - y_{wnfs}(k))^2 .$$

У таблиці 1.2 наведено порівняльний аналіз процесу емуляції на основі запропонованої гібридної вейвлет-нейро-фаззі системи на основі W-нейронів з налаштуванням всіх її параметрів з іншими підходами, описаними в літературних джерелах, а саме рекурентною фаззі-нейро мережею (Recurrent Fuzzy Neural Network, RFNN), рекурентною самоорганізовною нейро-фаззі мережею (recurrent self-organizing neural fuzzy inference network, RSONFIN) [75], нейро-фаззі системою із прямою передачею інформації (feedforward neural fuzzy systems, NFS), рекурентною фаззі мережею TSK-типу (TSK-type recurrent fuzzy network, TSK-RFN) [76] і фаззі-вейвлет нейронною мережею [77].

Таблиця 1.2 – Порівняльний аналіз результатів емуляції динамічного об’єкта

Нейронна мережа / Алгоритм навчання	Кількість епох навчання	RMSE	
		Навчальна вибірка	Тестова вибірка
Запропонована гібридна вейвлет-нейро-фаззі система на основі адаптивних W-нейронів (Wavelet-Neuro-Fuzzy Systems based on W-neurons, WNFS-W) /	30	0,0183	**0,02004**

Запропонований алгоритм навчання всіх параметрів мережі			
Фаззі-вейвлет нейронна мережа [77]	200	0,0282	0,0301
Рекурентна фаззі-нейро-мережа (Recurrent Fuzzy Neural Network, RFNN)	200	0,0114	0,0575
Рекурентна самоорганізована нейро-фаззі-мережа (Recurrent Self-Organizing Neural Fuzzy Inference Network, RSONFIN) [75]	200	0,0248	0,0780
Нейро-фаззі система із прямою передачею інформації (Feedforward Neural Fuzzy Systems, NFS)	200	0,0203	0,0521
Рекурентна фаззі-мережа ТСК-типа (TSK-type Recurrent Fuzzy Network, TSK-RFN) [76]	200	**0,0084**	0,0346

Результати емуляції динамічного об’єкту (1.54)–(1.56) наведено на рис. 1.14. Як видно, дві криві, що представляють реальні (пунктирна лінія) і модельні (суцільна лінія) значення, практично ідентичні.

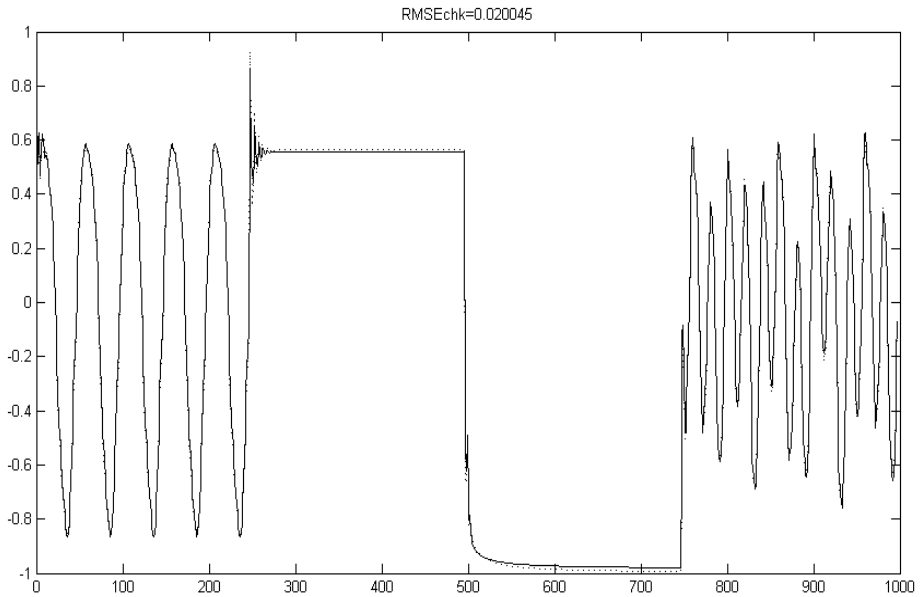


Рисунок 1.14 – Результати емуляції динамічного об’єкта

З наведених вище результатів можна бачити, що запропонована архітектура гібридної вейвлет-нейро-фаззі системи та алгоритм навчання всіх її параметрів забезпечують більш високу точність емуляції в порівнянні з іншими підходами за менший час навчання.

1.2 Нечітка сегментація зображень

Традиційно початковою інформацією для задачі кластеризації є вибірка спостережень, що складається з N n-вимірних векторів ознак $X = \{x(1), x(2), ..., x(k), ..., x(N)\}$, $x(k) = (x_1(k), ..., x_n(k))^T \in R^n$, $k = 1, 2, ..., N$, а результатом роботи алгоритму є розподіл початкового масиву даних на m класів з деяким рівнем $w_j(k)$ належності k-ого вектора ознак j-ому кластеру.

В той же час існує широкий клас задач, коли початкова інформація надходить не в векторній, а в матричній формі, тобто $x(k) = \{x_{i_1 i_2}(k)\}$; $i_1 = 1, 2, ..., n_1$; $i_2 = 1, 2, ..., n_2$; $k = 1, 2, ..., N$. Така ситуація є характерною, наприклад, при обробці зображень [78], коли початкова $(N_1 \times N_2)$-матриця розбивається на $N = N_1 \cdot N_2 \cdot (n_1 \cdot n_2)^{-1}$ $(n_1 \times n_2)$-матриць-фрагментів, які підлягають кластеризації, в результаті якої формуються однорідні в деякому сенсі сегменти цього зображення. Традиційно ця задача вирішуються шляхом попередньої векторизації фрагментів і використання вже відомих процедур, найбільш популярною з яких є метод кластеризації нечітких С-середніх [8], [79].

Для обробки матричних даних необхідно ввести матричні методи кластеризації-сегментації для чого доцільно ввести до розгляду матричний метод нечітких С-середніх, який є узагальненям FCM. Такий метод дозволить уникнути зайвих операцій векторизації-девекторизації при обробці даних, що задані у формі двовимірних масивів, та забезпечує обробку інформації в on-line режимі.

Отже, нехай задана вибірка спостережень $x(k)=\{x_{i_1 i_2}(k)\}\in R^{n_1\times n_2}$, $k=1,2,...,N$, при цьому для зручності подальшої обробки ці дані попередньо відцентровані відносно середнього:

$$\overline{x}=\frac{1}{N}\sum_{k=1}^{N}x(k) \tag{1.57}$$

і пронормовані на свою сферичну норму (Frobenius norm):

$$\|x(k)\|=\sqrt{Tr\left(x(k)x^T(k)\right)}. \tag{1.58}$$

Як цільова функція кластеризації використовується матричний імовірнісний критерій:

$$\begin{aligned} E(w_j(k),c_j)&=\sum_{k=1}^{N}\sum_{j=1}^{m}w_j^{\beta}(k)D^2(x(k),c_j)= \\ &=\sum_{k=1}^{N}\sum_{j=1}^{m}w_j^{\beta}(k)Tr\left((x(k)-c_j)(x(k)-c_j)^T\right) \end{aligned} \tag{1.59}$$

за наявності обмежень:

$$\sum_{j=1}^{m}w_j(k)=1,\ (\text{або}\ \sum_{j=1}^{m}w_j(k)-1=0),\ k=1,2,...,N,$$

$$0<\sum_{j=1}^{m}w_j(k)<N,\ \ j=1,2,...,m.$$

Вводячи функцію Лагранжа:

$$L(w_j(k),c_j,\lambda(k))=\sum_{k=1}^{N}\sum_{j=1}^{m}w_j^{\beta}(k)D^2(x(k),c_j)+\sum_{k=1}^{N}\lambda(k)\left(\sum_{j=1}^{m}w_j(k)-1\right)=$$
$$=\sum_{k=1}^{N}\left(\sum_{j=1}^{m}w_j^{\beta}(k)D^2(x(k),c_j)+\lambda(k)\left(\sum_{j=1}^{m}w_j(k)-1\right)\right), \quad (1.60)$$

де $\lambda(k)$ – невизначений множник Лагранжа, і вирішуючи систему рівнянь Куна-Таккера

$$\begin{cases}\dfrac{\partial L(w_j(k),c_j,\lambda(k))}{\partial w_j(k)}=\beta w_j^{\beta-1}(k)D^2(x(k),c_j)+\lambda(k)=0,\\ \dfrac{\partial L(w_j(k),c_j,\lambda(k))}{\partial \lambda_j(k)}=\sum_{j=1}^{m}w_j(k)-1=0,\\ \left\{\dfrac{\partial L(w_j(k),c_j,\lambda(k))}{\partial c_j(k)}\right\}=-2\sum_{k=1}^{N}w_j^{\beta}(k)(x(k)-c_j)=\mathrm{O},\end{cases}$$

де $\left\{\dfrac{\partial L(w_j(k),c_j,\lambda(k))}{\partial c_j(k)}\right\}$ – $(n_1\times n_2)$-матриця, що формується з частинних похідних $\dfrac{\partial L(w_j(k),c_j,\lambda(k))}{\partial c_{ji_1i_2}}$;

O – матриця тієї ж розмірності, що утворена нулями, таким чином, приходимо до кінцевого вигляду алгоритму:

$$\begin{cases} w_j(k) = \dfrac{(D^2(x(k), c_j))^{\frac{1}{1-\beta}}}{\sum\limits_{l=1}^{m}(D^2(x(k), c_l))^{\frac{1}{1-\beta}}}, \\ \lambda(k) = -\left(\sum\limits_{l=1}^{m}\left(\beta D^2(x(k), c_l)^{\frac{1}{1-\beta}} \right)^{1-\beta} \right), \\ c_j = \dfrac{\sum\limits_{k=1}^{N} w_j^{\beta}(k) x(k)}{\sum\limits_{k=1}^{N} w_j^{\beta}(k)}. \end{cases} \quad (1.61)$$

Отримана система породжує широкий клас процедур кластеризації. Так, якщо покласти $\beta = 2$, отримуємо простий і ефективний алгоритм матричної кластеризації [80], який є узагальненням популярної процедури Дж. Бездека [8]:

$$\begin{cases} w_j(k) = \dfrac{(Tr(x(k) - c_j)(x(k) - c_j)^T)^{-1}}{\sum\limits_{l=1}^{m}(Tr(x(k) - c_l)(x(k) - c_l)^T)^{-1}}, \\ c_j = \dfrac{\sum\limits_{k=1}^{N} w_j^2(k) x(k)}{\sum\limits_{k=1}^{N} u_j^2(k)}, \end{cases} \quad (1.62)$$

де Tr – символ сліду матриці.

Основна різниця між імовірнісним та можливісним підходами полягає в тому, що імовірнісні алгоритми використовують відносні подібності між об’єктами і кластерами, в той час як можливісні алгоритми використовують абсолютні подібності.

Замість матриці нечіткого розбиття в алгоритмі нечітких С-середніх, можливісний алгоритм С-середніх використовує $(N \times m)$-матрицю можливостей (matrix of possibilities або typicality matrix) $T = \{t_j(k)\}$, де $t_j(k) \in [0,1]$ –

можливість того, що об’єкт $x(k)$ належить кластеру j. Матриця можливостей має лише такі обмеження:

$$0 < \sum_{j=1}^{m} t_j(k) \le m,\ k = 1,2,\ldots,N. \tag{1.63}$$

Це означає, що об’єкт може мати вектор можливостей, який містить тільки значення, близькі до нуля (зазвичай такі об’єкти вважають шумом), чи тільки одиниці.

Кришнапурам, Келлер та інші запропонували можливісний алгоритм С-середніх (PCM) та два алгоритми, в яких імовірнісний та можливісний підходи об’єднані: імовірнісно-можливісний алгоритм С-середніх (FPCM) та можливісно-імовірнісний алгоритм С-середніх (PFCM) [81]–[83].

В алгоритмі PCM формула (1.62) була змінена виразом:

$$\begin{cases} t_j(k) = \dfrac{1}{1 + \left(\dfrac{Tr\left(x(k) - c_j\right)\left(x(k) - c_j\right)^T}{\gamma_j} \right)^{\frac{1}{\beta - 1}}}, \\ c_j = \dfrac{\sum_{k=1}^{N} t_j^{\beta}(k) x(k)}{\sum_{k=1}^{N} t_j^{\beta}(k)}, \end{cases} \tag{1.64}$$

де $\gamma_j > 0$ – константа, що задається емпіричним шляхом.

Можна помітити, що обчислення прототипу кластеру в формулах (1.62) та (1.64) ідентично, з тією лише різницею, що матриця нечіткого розбиття змінена на матрицю можливостей. Обчислення можливості належності об’єкту до кластеру у формулі (1.64) може бути обґрунтовано як дзвонувата функція, що представлена на рисунку 1.15.

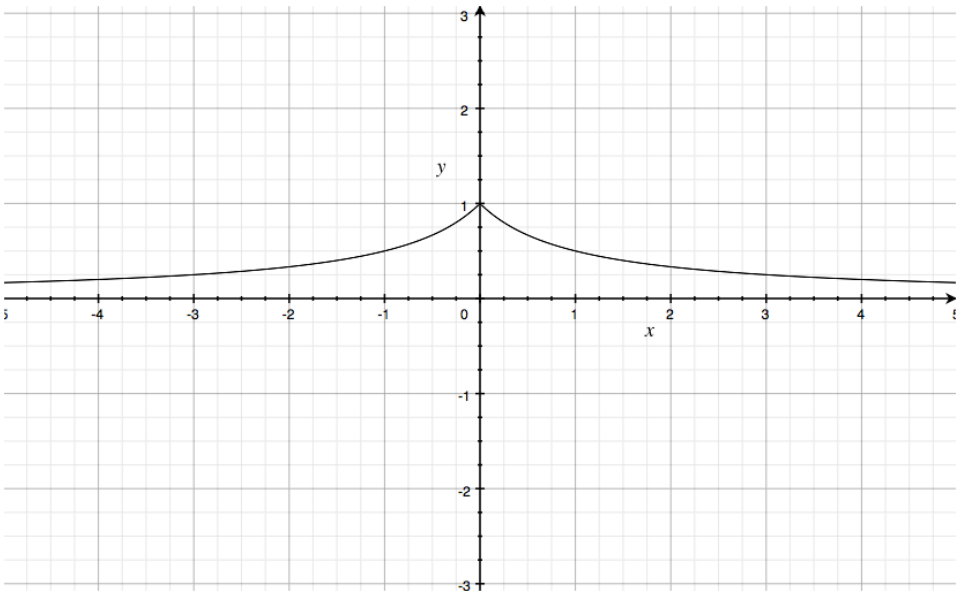


Рисунок 1.15 – Дзвонувата функція, що показує залежность між відстанями належності (для $\beta = 2$ та $\gamma_j = 1$)

Келлер та Кришнапурам запропонували обирати параметр γ_j у формі

$$\gamma_j = K \frac{\sum_{k=1}^{N} w_j^{\beta}(k) Tr\left(x(k) - c_j\right)\left(x(k) - c_j\right)^T}{\sum_{k=1}^{N} w_j^{\beta}(k)}, \qquad (1.65)$$

де $K > 0$ (частіше всього $K = 1$).

Але обчислення γ_j по формулі (1.65) потребує пам’яті для зберігання матриці нечіткого розбиття, а також час для її використання.

Алгоритм PCM добре справляється з придушенням завад і зазвичай може бути застосований коли необхідно покращити результати, що отримані за допомогою інших алгоритмів. Також цей алгоритм може об’єднати близькі кластери в один, з чого випливає, що початкова кількість кластерів, що була задана заздалегідь, занадто велика (в той же час, алгоритм PCM може об’єднати кластери, які повинні бути розділені).

Алгоритми FPCM та PFCM використовують як матрицю нечіткого розбиття, так і матрицю можливостей, намагаючись використовувати переваги обох підходів.

FPCM алгоритм використовує такі формули:

$$\begin{cases} w_j(k) = \dfrac{\left(Tr\left(x(k)-c_j\right)\left(x(k)-c_j\right)^T\right)^{\frac{1}{1-\beta}}}{\sum_{l=1}^{m}\left(Tr\left(x(k)-c_l\right)\left(x(k)-c_l\right)^T\right)^{\frac{1}{1-\beta}}}, \\ t_j(k) = \dfrac{\left(Tr\left(x(k)-c_j\right)\left(x(k)-c_j\right)^T\right)^{\frac{1}{1-\eta}}}{\sum_{l=1}^{N}\left(Tr\left(x(l)-c_j\right)\left(x(l)-c_j\right)^T\right)^{\frac{1}{1-\eta}}}, \\ c_j = \dfrac{\sum_{k=1}^{N}\left(w_j^{\beta}(k)+t_j^{\eta}(k)\right)x(k)}{\sum_{k=1}^{N}\left(w_j^{\beta}(k)+t_j^{\eta}(k)\right)}, \end{cases} \tag{1.66}$$

де $\eta > 0$ (в більшості випадках $\eta = 2$).

Алгоритм FPCM використовує стандартну процедуру обчислення матриці нечіткого розбиття, але матриця можливостей обчислюється за новою формулою. Прототипи кластерів обчислюються використовуючи суму обох матриць.

Метод PFCM використовує стандартну процедуру обчислення матриці нечіткого розбиття (як в формулі (1.62)). Процедура обчислення матриці можливостей була взята з PCM (1.64) і незначно змінена. Центроїди обчислюються як і в алгоритмі FPCM, але обидві матриці мають свої ваги:

$$\begin{cases} w_j(k) = \dfrac{\left(Tr\left(x(k)-c_j\right)\left(x(k)-c_j\right)^T\right)^{\frac{1}{1-\beta}}}{\sum\limits_{l=1}^{m}\left(Tr\left(x(k)-c_l\right)\left(x(k)-c_l\right)^T\right)^{\frac{1}{1-\beta}}}, \\ t_j(k) = \dfrac{1}{1+\left(b\dfrac{Tr\left(x(k)-c_j\right)\left(x(k)-c_j\right)^T}{\gamma_j}\right)^{\frac{1}{\beta-1}}}, \\ c_j = \dfrac{\sum\limits_{k=1}^{N}\left(aw_j^{\beta}(k)+bt_j^{\eta}(k)\right)x(k)}{\sum\limits_{k=1}^{N}\left(aw_j^{\beta}(k)+bt_j^{\eta}(k)\right)}, \end{cases} \tag{1.67}$$

де $a>0,\ b>0$.

Константи a та b визначають відносну важливість матриці нечіткого розбиття і матриці можливостей в функції обчислення центроїдів. Задавши $a=0$, алгоритм (1.67) переходить у PCM, а задавши $b=0$, алгоритм (1.67) переходить у FCM.

Аналізуючи всі представлені методи, можна зробити декілька висновків. По-перше, функція належності алгоритму FCM з його обмеженнями є занадто «сильною», що дозволяє відносити outlier-об’єкти до одного чи більше кластерам, що, в свою чергу, може сильно вплинути на основну структуру набору даних. З іншого боку, обмеження методу PCM для можливостей є занадто слабким – він дозволяє відноситься до кластеру незалежно від решти даних. Також PCM дуже чутливий до ініціалізації матриці можливостей. Метод PFCM є ефективною комбінацією двох підходів та результати кластеризації залежать від задання параметрів $a,\ b,\ \beta,\ \eta$.

Алгоритм (1.62) може бути розширено на випадок коли дані на обробку надходять послідовно в online режимі. Для цього, застосовуючи до лагранжіану (1.60) процедуру пошуку сідлової точки Ерроу–Ґурвіца–Удзави, при

надходженні $(k+1)$-ого спостереження оцінки рівнів належностей і центроїдів можуть бути уточнені за допомогою рекурентних співвідношень [84]

$$\begin{cases} w_j(k+1) = \dfrac{(D^2(x(k+1), c_j(k))^{\frac{1}{1-\beta}})}{\sum\limits_{l=1}^{m}(D^2(x(k+1), c_l(k))^{\frac{1}{1-\beta}})}, \\ c_j(k+1) = c_j(k) - \eta(k)\left\{\dfrac{\partial L(w_j(k+1), c_j, \lambda(k+1))}{\partial c_j}\right\} = \\ = c_j(k) + \eta(k) w_j^{\beta}(k+1)(x(k+1) - c_j(k)) \end{cases} \quad (1.68)$$

для довільного значення фаззіфікатора β та

$$\begin{cases} w_j(k+1) = \dfrac{(Tr(x(k+1) - c_j(k))(x(k+1) - c_j(k))^T)^{-1}}{\sum\limits_{l=1}^{m}(Tr(x(k+1) - c_l(k))(x(k+1) - c_l(k))^T)^{-1}}, \\ c_j(k+1) = c_j(k) + \eta(k) w_j^2(k+1)(x(k+1) - c_j(k)) \end{cases} \quad (1.69)$$

для $\beta = 2$. Нескладно побачити, що вираз (1.68) є адаптивною версією процедури (1.61), а (1.69) – відповідно (1.62).

Для тестування реалізованих матричних модифікацій алгоритму кластеризації нечітких С-середніх була використана вибірка з UCI-репозиторію Iris (ірис) [85], а також цифрові зображення, у тому числі сателітні знімки міста Харкова. Вибірки не мають пропущених атрибутів і є числовими.

Результатом роботи алгоритму є кінцева матриця нечіткого розбиття для усіх об’єктів вибірки і прототипи класів.

При обробці цифрових зображень, об’єкти (матриці чи вектори однакової розмірності) формуються з фрагментів цього зображення, а кожен піксель з кольорової моделі RGB (Red-Green-Blue) переводиться до моделі Grayscale, де

яскравість пікселя виражається скалярним значенням з інтервалу [0, 1]. Переведення з моделі RGB до моделі Grayscale виконується згідно з формули $Y = (0.299R + 0.587G + 0.114B)/255$, де Y – яскравість світіння пікселя, R, G, B – яскравості світіння красного, зеленого і синього тонів відповідно, значення яких знаходяться в інтервалі [0, 255].

Набори спостережень, що сформовані з цифрових зображень, оброблюються за тим же принципом, що і стандартні кількісні вибірки. Після обробки зображень, кожному кластеру присвоюється кольори моделі Grayscale, а кожен об'єкт забарвлюється в колір найближчого кластера. Для оцінки якості роботи алгоритму використовувались такі критерії: Partition Coefficient (PC), Classification Entropy (CE), Partition Index (PI) при однаковій ініціалізованій матриці нечіткого розбиття U^0. В табл. 1.3 наведено результати точності і швидкості роботи алгоритмів кластеризації на вибірці Iris, а в табл. 1.4 – на сателітному цифровому зображені міста Харкова. Час наведено в середньому для одній ітерації з урахуванням операції векторизації-девекторизації.

Таблиця 1.3 – Результати кластерного аналізу на вибірці Iris

Алгоритм кластеризації	PC	CE	PI	Час, сек.
Нечітких С-середніх	0.531	0.811	12.19	0.003
Матричний алгоритм нечітких С-середніх	0.531	0.811	12.19	0.0025

Таблиця 1.4 – Результати кластерного аналізу на цифровому зображенні

Алгоритм кластеризації	PC	CE	PI	Час, сек.
Нечітких С-середніх	0.697	0.419	8.23	1.9
Матричний алгоритм нечітких С-середніх	0.697	0.419	8.23	1.8

На рисунках 1.16–1.18 відповідно наведені початкове зображення, передоброблена вибірка (20% об’єктів), результат кластерного аналізу і процес роботи алгоритму.

Рисунок 1.16 – Початкове цифрове зображення для кластеризації

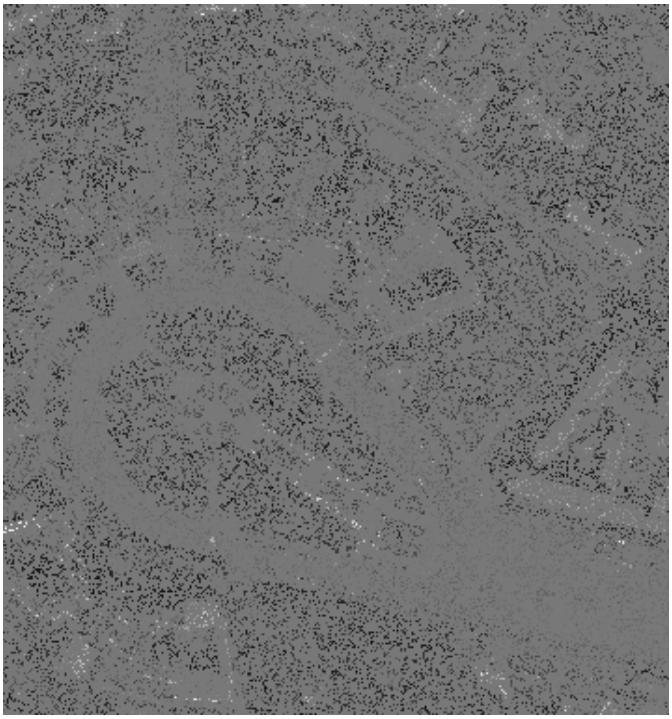


Рисунок 1.17 – Передоброблена вибірка (20% об’єктів) для кластеризації

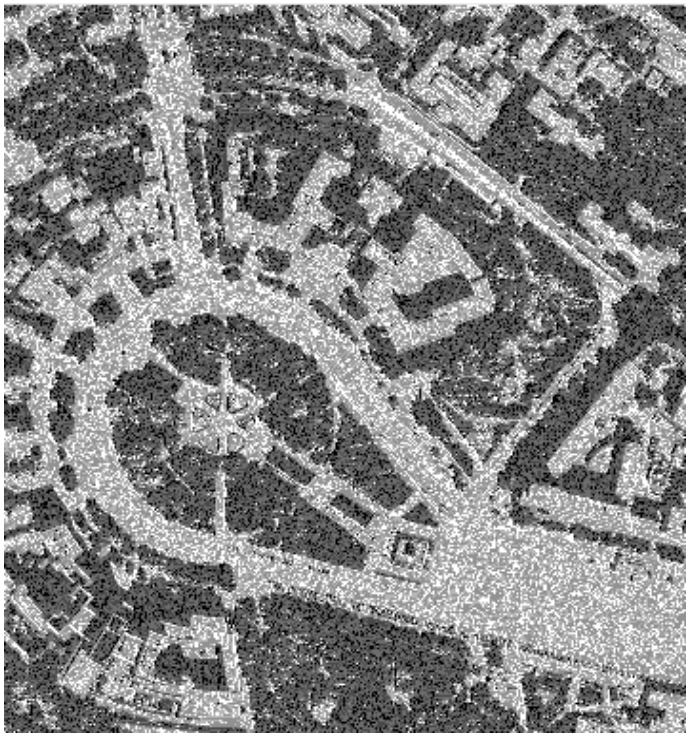


Рисунок 1.18 – Вихідне зображення кластерного аналізу

На рисунку 1.19 наведено результат кластеризації цифрового зображення адаптивним матричним алгоритмом нечітких С-средніх.

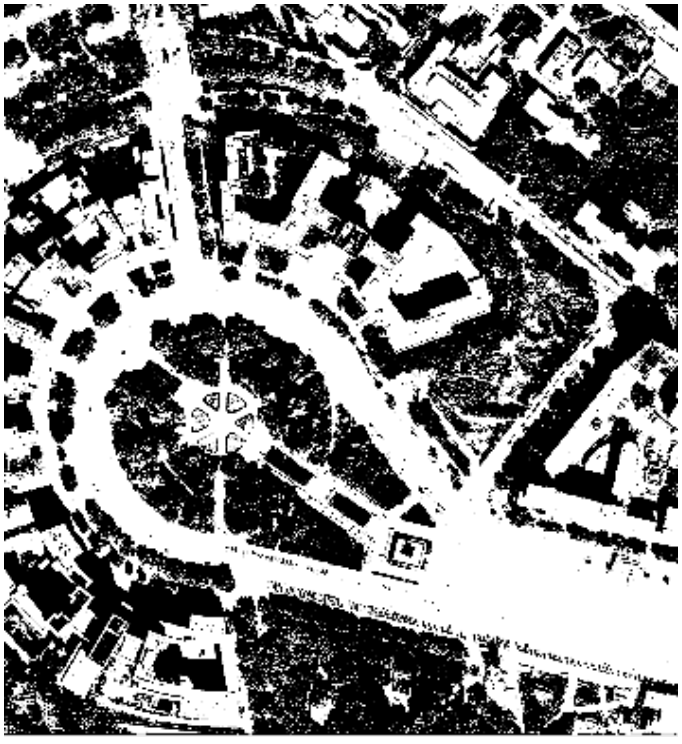


Рисунок 1.19 – Вихідне зображення після адаптивного кластерного аналізу

1.3 Гібридна каскадна нейро-фаззі мережа з оптимізацією пулу нейронів

Зазвичай під «навчанням» розуміють процес коригування синаптичних вагових коефіцієнтів за допомогою певної процедури оптимізації, що ґрунтується на пошуку екстремуму заданого критерію навчання. Якість процесу навчання може бути поліпшена шляхом коригування топології мережі поспіль з синаптичними вагами [87], [88]. Ця ідея полягає в основі систем обчислювального інтелекту, що еволюціонують [89], [90].

Мабуть, найбільш відомою реалізацією цього підходу є каскадно-кореляційні нейронні мережі [91], [92], привабливі високою ефективністю та простотою налаштування як синаптичних вагових коефіцієнтів, так і топології мережі. Така мережа на початку містить лише один пул (ансамбль) нейронів, які навчаються незалежно один від іншого (перший каскад). Кожен нейрон у пулі може мати відмінні функції активації та метод навчання. Доки навчання триває, нейрони у пулі не взаємодіють один з одним. Після того, як процес налаштування вагових коефіцієнтів завершився для всіх нейронів пулу першого каскаду, кращий нейрон відповідно до обраного критерію навчання формує перший каскад і коефіцієнти його синаптичних ваг більше не коригуються. Далі формується другий каскад зазвичай з нейронів, подібних до нейронів першого каскаду. Різниця лише в тому, що нейрони, які навчаються в пулі другого каскаду, мають додатковий вхід (і, отже, додатковий синаптичний ваговий коефіцієнт) – вихід першого каскаду. Подібно до першого каскаду, у другому каскаді залишається лише один найбільш продуктивний нейрон і його синаптичні вагові коефіцієнти фіксуються. Аналогічним чином нейрони третього каскаду матимуть два додаткових входи, а саме виходи першого та другого каскадів. Еволюційна мережа продовжуватиме розширяти свою архітектуру новими каскадами, доки вона не досягне бажаної якості вирішення завдання для заданого набору даних.

Автори найбільш поширеної каскадної нейронної мережі, що еволюціонує,

CasCorLA, Фальман та Лєб'єр, використовували елементарні персептрони Розенблатта з традиційними сигмоїдальними функціями активації і коригували синаптичні вагові коефіцієнти за допомогою QuickProp-алгоритму, що є модифікацією δ-правила. Оскільки вихідний сигнал таких нейронів нелінійно залежить від синаптичних ваг, швидкість навчання не може бути суттєво збільшена для таких нейронів.

Для уникнення багатоепошного навчання доцільно в якості вузлів системи використовувати такі типи нейронів, що їх виходи лінійно залежать від синаптичних ваг, що дозволить використовувати оптимальні за швидкодією методи навчання та обробляти дані в онлайн режимі.

Проте варто зазначити, що у випадку послідовного навчання системи, неможливо визначити найкращий нейрон у пулі, адже при оброблянні нестаціонарних об'єктів певний нейрон може бути кращим для однієї частини тренувальної вибірки, проте поступатися у точності іншому нейрону на іншій частині вибірки. Отже доцільно зберегти усі нейрони пулу та використовувати певну оптимізуючу процедуру (відповідно обраному критерію якості) задля визначення нейрона-переможця на кожному кроці обробляння даних.

1.3.1 Архітектура оптимізованої каскадної нейронної мережі

Архітектура пропонованої гібридної системи з оптимізованим пулом нейронів у кожному каскаді наведена на рис. 1.20.

На вхід такої системи (так званий «рецептивний» шар) подається векторний сигнал

$$x(k) = \left(x_1(k), x_2(k), \ldots, x_n(k)\right)^T, \tag{1.70}$$

де $k = 1,2, \ldots$ – кількість образів у таблиці «об'єкт – властивість» або поточний дискретний час.

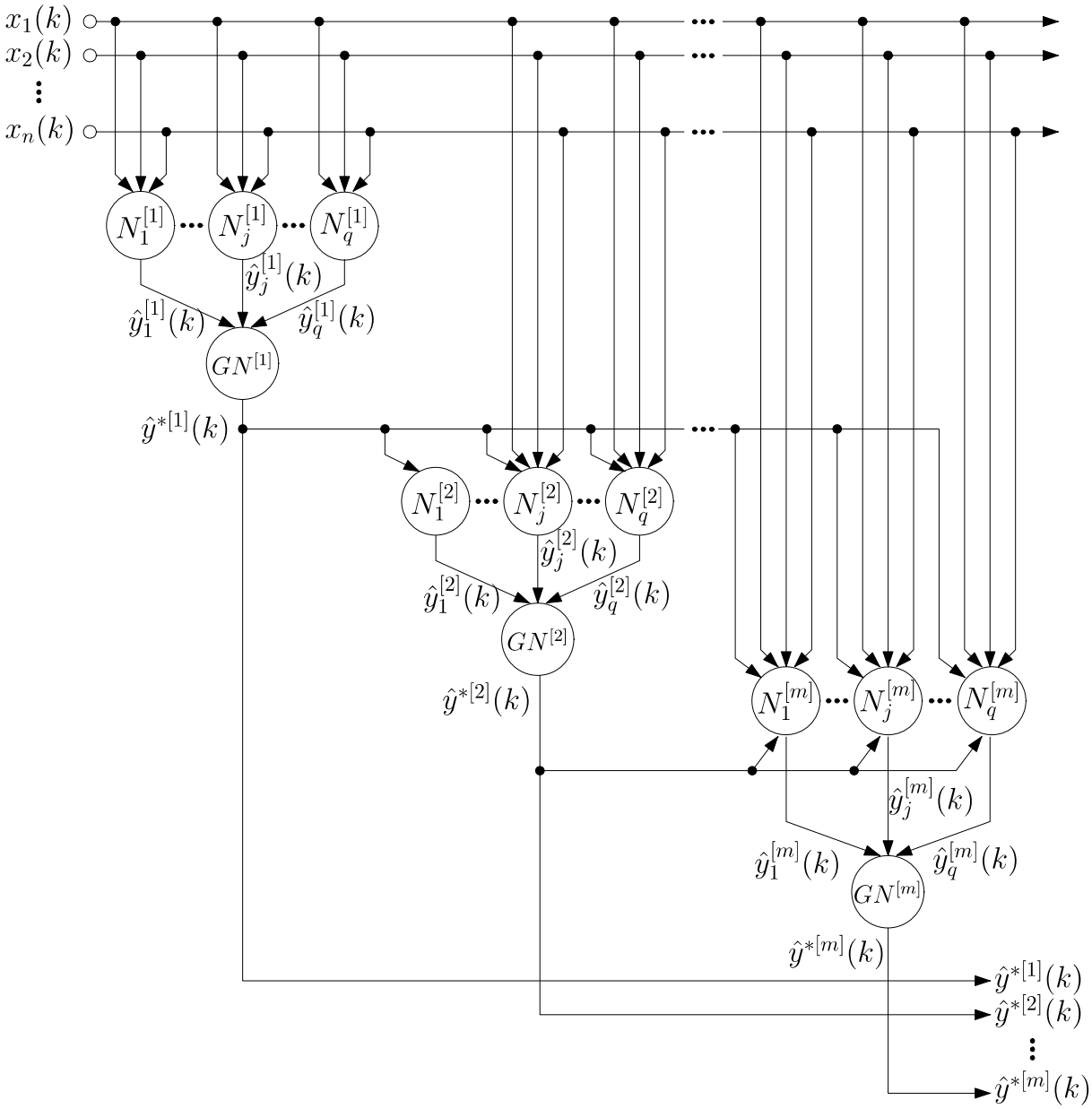


Рисунок 1.20 – Архітектура гібридної системи з оптимізованим пулом нейронів

Ці сигнали подаються на входи кожного нейрона в мережі $N_j^{[m]}$ $j = 1, 2, \dots, q$ – кількість нейронів у тренувальному пулі, $m = 1,2, \dots$ – номер каскаду) з вихідним сигналом $\hat{y}_j^{[m]}(k)$ Далі вихідні сигнали кожного каскаду надходять до «узагальнюючого» вузлу $GN^{[m]}$, який генерує поточно-оптимальний вихідний сигнал відповідного каскаду $\hat{y}^{*[m]}$. Слід зауважити, що вхідними сигналами першого каскаду є вектор $x(k)$ (що може містити опціональне порогове значення

$x_0(k) \equiv 1$), другий каскад має додатковий вхід для сгенерованого першим каскадом вихідного сигналу $\hat{y}^{*[1]}(k)$, нейрони третього каскаду оброблятимуть два додаткових сигнали $\hat{y}^{*[1]}(k)$, $\hat{y}^{*[2]}(k)$, нейрони m-ого каскаду матимуть $(m-1)$ додаткових вхідних сигналів: $\hat{y}^{*[1]}(k), \hat{y}^{*[2]}(k), \ldots, \hat{y}^{*[m-1]}(k)$. Під час тренування системи нові каскади додаються доки не буде досягнута бажана точність.

1.3.2 Навчання елементарних персептронів Розенблатта у каскадній оптимізованій системі

Наразі вважатимемо j-й вузол m-ого каскаду елементарним персептроном Розенблатта з активаційною функцією

$$0 < \sigma_j^{[m]}\left(\gamma_j^{[m]} u_j^{[m]}\right) = \frac{1}{1 + e^{-\gamma_j^{[m]} u_j^{[m]}}} < 1, \tag{1.71}$$

де $u_j^{[m]}$ – внутрішній активаційний сигнал j-ого нейрону m-ого каскаду;

$\gamma_j^{[m]}$ – параметр посилення.

У такому випадку вихідні сигнали нейронів тренувального пулу першого каскаду матимуть вигляд

$$\hat{y}_j^{[1]} = \sigma_j^{[1]}\left(\gamma_j^{[1]} \sum_{i=0}^{n} w_{ji}^{[1]} x_i\right) = \sigma_j^{[1]}\left(\gamma_j^{[1]} w_j^{[1]T} x\right), \tag{1.72}$$

де $w_{ji}^{[1]}$ – i-й ваговий коефіцієнт j-ого нейрону першого каскаду.

Вихідні сигнали другого каскаду дорівнюватимуть

$$\hat{y}_j^{[2]} = \sigma_j^{[2]}\left(\gamma_j^{[2]}\left(\sum_{i=0}^{n} w_{ji}^{[2]} x_i + w_{j,n+1}^{[2]}\hat{y}^{*[1]}\right)\right), \tag{1.73}$$

вихідні сигнали m-ого каскаду матимуть вигляд

$$\begin{aligned}\hat{y}_j^{[m]} = \sigma_j^{[m]}\Bigg(\gamma_j^{[m]}\Bigg(\sum_{i=0}^{n} w_{ji}^{[m]} x_i + w_{j,n+1}^{[m]}\hat{y}^{*[1]} + w_{j,n+2}^{[m]}\hat{y}^{*[2]} + \cdots + \\ + w_{j,n+m-1}^{[m]}\hat{y}^{*[m-1]}\Bigg)\Bigg) = \sigma_j^{[m]}\left(\gamma_j^{[m]}\sum_{i=0}^{n+m-1} w_{ji}^{[m]} x_i^{[m]}\right) = \\ = \sigma_j^{[m]}\left(w_j^{[m]T} x^{[m]}\right),\end{aligned} \tag{1.74}$$

де $x^{[m]} = \left(x^T, \hat{y}^{*[1]}, \dots, \hat{y}^{*[m-1]}\right)^T$.

Таким чином, нейронна мережа з персептронами Розенблатта у якості вузлів, що містить m каскадів, залежить від $\left(m(n+2) + \sum_{p=1}^{m-1} p\right)$параметрів, у тому числі від параметрів посилення $\gamma_j^{[p]}, p = 1,2, \dots, m$.

У якості критерію навчання можна використовувати загальноприйняту квадратичну функцію

$$\begin{aligned}E_j^{[m]} = \frac{1}{2}\left(e_j^{[m]}(k)\right)^2 = \\ = \frac{1}{2}\left(y(k) - \hat{y}_j^{[m]}(k)\right)^2 = \\ = \frac{1}{2}\left(y(k) - \sigma_j^{[m]}\left(\gamma_j^{[m]} w_j^{[m]T} x^{[m]}(k)\right)\right)^2,\end{aligned} \tag{1.75}$$

де $y(k)$– бажане значення вихідного сигналу.

Градієнтну оптимізацію критерію (1.75) відносно $\gamma_j^{[m]}$ можна записати у вигляді

$$
\begin{aligned}
&w_j^{[m]}(k+1) = w_j^{[m]}(k) + \eta_j^{[m]}(k+1)e_j^{[m]}(k+1)\gamma_j^{[m]}\hat{y}_j^{[m]}(k+1) \times \\
&\times \left(1 - \hat{y}_j^{[m]}(k+1)\right) x^{[m]}(k+1) = \\
&= w_j^{[m]}(k) + \eta_j^{[m]}(k+1)e_j^{[m]}(k+1)\gamma_j^{[m]}J_j^{[m]}(k+1),
\end{aligned}
\tag{1.76}
$$

де $\eta_j^{[m]}(k+1)$ – параметр кроку навчання.

Мінімізувати критерій (**Помилка! Джерело посилання не знайдено.**) відносно $\gamma_j^{[m]}$ можна за допомогою алгоритму Крушке–Мовеланна [93]

$$
\begin{aligned}
\gamma_j^{[m]}(k+1) = \gamma_j^{[m]}(k) + \eta_j^{[m]}(k+1)e_j^{[m]}(k+1)\hat{y}_j^{[m]}(k+1) \times \\
\times \left(1 - \hat{y}_j^{[m]}(k+1)\right) u_j^{[m]}(k+1).
\end{aligned}
\tag{1.77}
$$

Поєднуючи (1.75) та (1.76), отримаємо алгоритм навчання для j-ого нейрону m-ого каскаду

$$
\begin{aligned}
&\begin{pmatrix} w_j^{[m]}(k+1) \\ \cdots \\ \gamma_j^{[m]}(k+1) \end{pmatrix} \\
&= \begin{pmatrix} w_j^{[m]}(k) \\ \cdots \\ \gamma_j^{[m]}(k) \end{pmatrix} + \eta_j^{[m]}(k+1)e_j^{[m]}(k+1)\hat{y}_j^{[m]}(k+1) \times \\
&\times \left(1 - \hat{y}_j^{[m]}(k+1)\right) \begin{pmatrix} \gamma_j^{[m]} x^{[m]}(k+1) \\ \cdots \\ u_j^{[m]}(k+1) \end{pmatrix},
\end{aligned}
\tag{1.78}
$$

або, вводячи нові змінні, у більш компактній формі

$$\widetilde{w}_j^{[m]}(k+1) = \widetilde{w}_j^{[m]}(k) + \eta_j^{[m]}(k+1)e_j^{[m]}(k+1)\hat{y}_j^{[m]}(k+1)\tilde{x}^{[m]}(k+1) = \quad (1.79)$$
$$= \widetilde{w}_j^{[m]}(k) + \eta_j^{[m]}(k+1)e_j^{[m]}(k+1)\tilde{J}_j^{[m]}(k+1).$$

Використовуючи регуляризуючий параметр (momentum term) [94], [95], можна удосконалити процес корегування синаптичних вагових коефіцієнтів під час навчання. Тоді, замість критерію (1.75) слід використовувати функцію

$$E_j^{[m]}(k) = \frac{\eta}{2}\left(e_j^{[m]}(k)\right)^2 + \frac{1-\eta}{2}\left\|\widetilde{w}_j^{[m]}(k) - \widetilde{w}_j^{[m]}(k-1)\right\|^2, \quad (1.80)$$
$$0 < \eta < 1.$$

Тоді метод навчання приймає вигляд

$$\widetilde{w}_j^{[m]}(k+1) = \widetilde{w}_j^{[m]}(k) + \eta_j^{[m]}(k+1)\left(\eta e_j^{[m]}(k+1)\tilde{J}_j^{[m]}(k+1) + (1-\eta)\left(\widetilde{w}_j^{[m]}(k) - \widetilde{w}_j^{[m]}(k+1)\right)\right), \quad (1.81)$$

що є модифікацією процедури Сільви-Альмейди [96].

Доцільно вдосконалити алгоритм, використовуючи підхід, запропонований у [97], тоді алгоритм (1.81**Помилка! Джерело посилання не знайдено.**) набуває слідкуючих та фільтруючих властивостей. Таким чином, кінцева модифікація методу набуває вигляду

$$\begin{cases} \widetilde{w}_j^{[m]}(k+1) = \widetilde{w}_j^{[m]}(k) + \dfrac{\eta e_j^{[m]}(k+1)\tilde{J}_j^{[m]}(k+1)}{r_j^{[m]}(k+1)} + \\ \qquad + \dfrac{(1-\eta)\left(\widetilde{w}_j^{[m]}(k) - \widetilde{w}_j^{[m]}(k-1)\right)}{r_j^{[m]}(k+1)}, \\ r_j^{[m]}(k+1) = r_j^{[m]}(k) + \left\|\tilde{J}_j^{[m]}(k+1)\right\|^2 - \left\|\tilde{J}_j^{[m]}(k-s)\right\|^2, \end{cases} \tag{1.82}$$

де s – розмір ковзного вікна.

Цікаво, що при $s = 1$ та $\eta = 1$ отримуємо нелінійну версію загально-відомого алгоритму Качмажа-Уідроу-Хоффа:

$$\widetilde{w}_j^{[m]}(k+1) = \widetilde{w}_j^{[m]}(k) + \frac{e_j^{[m]}(k+1)\tilde{J}_j^{[m]}(k+1)}{\left\|\tilde{J}_j^{[m]}(k+1)\right\|^2}, \tag{1.83}$$

який широко використовується для навчання штучних нейронних мереж і відомий високою швидкістю збіжності.

1.3.3 Навчання нео-фаззі нейронів у оптимізованій каскадній нейронній мережі

Низька швидкість навчання персептронів Розенблатта у поєднанні з труднощами інтерпретації результатів (властиві всім штучним нейронним мережам в цілому) спонукає шукати альтернативні підходи до синтезу еволюційних нейронних мереж. Як відомо, нейро-фаззі системи відомі високою інтерпретовністю і прозорістю, а також високими апроксимаційними властивостями, та є основою гібридних систем штучного інтелекту. У [98], [99] розглядаються гібридні каскадні системи штучного інтелекту побудовані на нео-фаззі нейронах, що дозволяє їм суттєво підвищити швидкість корегування синаптичних вагових коефіцієнтів.

Нео-фаззі нейрон (NFN), що його архітектуру наведено на рис. 1.21, – це нелінійна система, що реалізує нечітке висновування

$$\hat{y} = \sum_{i=1}^{n} f_i(x_i), \tag{1.84}$$

де x_i – i-й вхідний сигнал $i = 1,2, \ldots$;

$\hat{y}$ – вихідний сигнал нео-фаззі нейрону.

Структурними елементами нео-фаззі нейрона є нелінійні синапси NS_i які трансформують вхідні сигнали в такий спосіб:

$$f_i(x_i) = \sum_{l=1}^{h} w_{li}\mu_{li}(x_i), \tag{1.85}$$

де w_{li} – l-й ваговий коефіцієнт i-ого нелінійного синапсу;

$l = 1,2, \ldots, h$ – кількість синаптичних вагових коефіцієнтів, а отже і функцій належності у $\mu_{li}(x_i)$ синапсі.

Таким чином, нелінійний синапс NS_i реалізує нечітке висновування

$$\text{Якщо } x_i - \text{ це } X_{li}, \text{ тоді вихід } - w_{li}, \tag{1.86}$$

де X_{li} – нечітка множина з функцією належності μ_{li};

w_{li} – сінглтон (синаптичний ваговий коефіцієнт у консеквенті).

Тобто нелінійній синапс фактично є системою висновування Такаґі–Суґено нульового порядку.

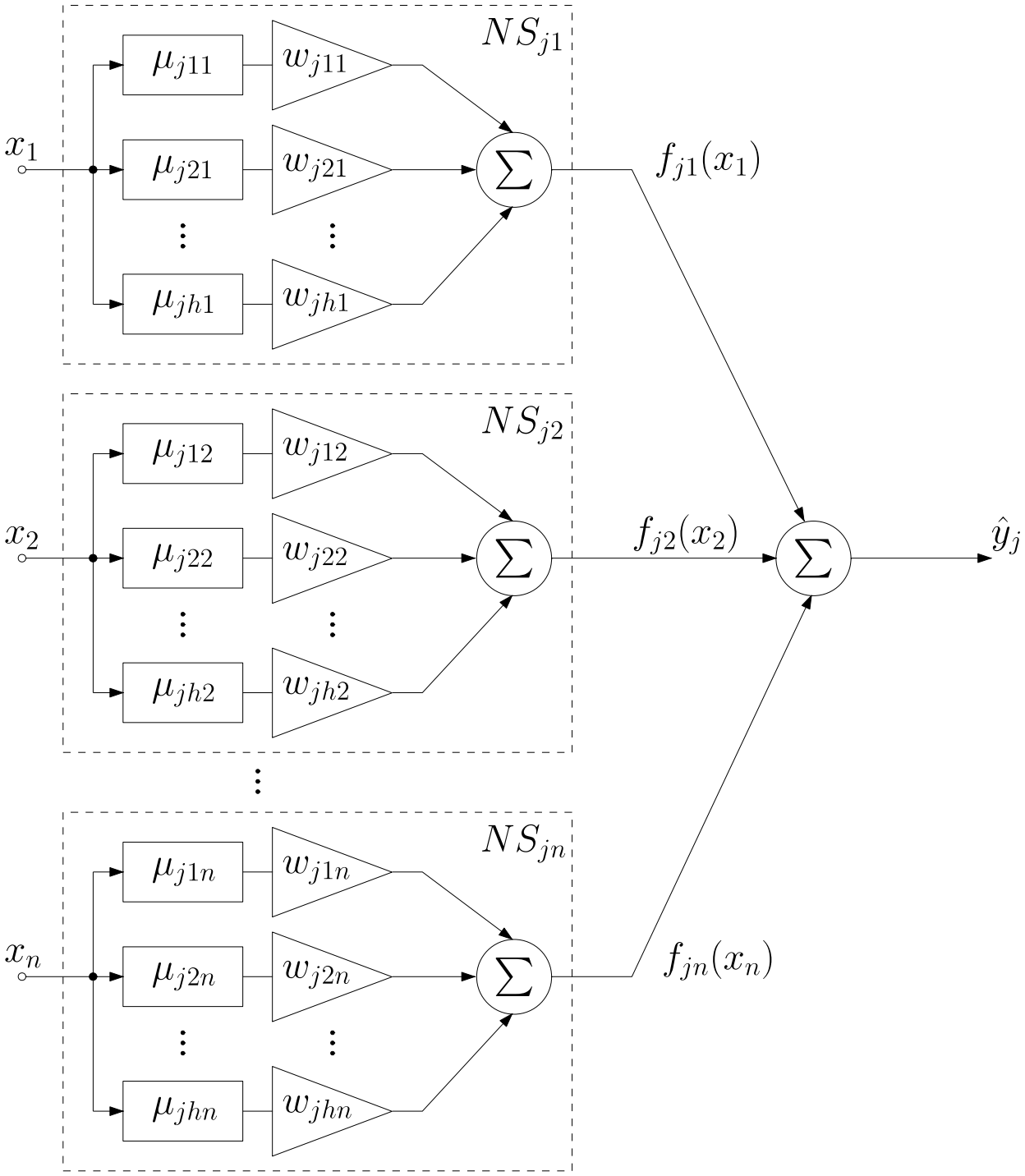


Рисунок 1.21 –Архітектура нео-фаззі нейрону

Запишемо вихідні сигнали для нейронів першого каскаду у такому вигляді:

$$\begin{cases} \hat{y}_j^{[1]}(k) = \sum_{i=1}^{n} f_{ji}^{[1]}\big(x(k)\big) = \sum_{i=1}^{n}\sum_{l=1}^{h} w_{jli}^{[1]}\mu_{jli}^{[1]}\big(x_i(k)\big), \\ \text{якщо } x_i - \text{ це } X_{li}, \text{тоді вихід} - w_{li}. \end{cases} \tag{1.87}$$

Автори нео-фаззі нейрона в якості функцій належності використовували традиційні трикутні структури, які задовольняють умові розбиття Руспіні:

$$\mu_{jli}^{[1]}(x_i) = \begin{cases} \dfrac{x_i - c_{j,l-1,i}^{[1]}}{c_{jli}^{[1]} - c_{j,l-1,i}^{[1]}}, & \text{якщо } x_i \in \left[c_{j,l-1,i}^{[1]}, c_{jli}^{[1]}\right], \\ \dfrac{c_{j,l+1,i}^{[1]} - x_i}{c_{j,l+1,i}^{[1]} - c_{jli}^{[1]}}, & \text{якщо } x_i \in \left[c_{jli}^{[1]}, c_{j,l+1,i}^{[1]}\right], \\ 0 & \text{у протилежному випадку,} \end{cases} \tag{1.88}$$

де $c_{jli}^{[1]}$ – довільно обрані центри параметрів функцій належності на інтервалі $[0,1]$, зазвичай рівномірно розподілені.

Такий вибір функцій належності гарантує, що вхідний сигнал x_i активує лише два сусідні функції, а сума їх значень завжди дорівнюватиме 1:

$$\mu_{jli}^{[1]}(x_i) + \mu_{j,l+1,i}^{[1]}(x_i) = 1, \tag{1.89}$$

$$f_{jl}^{[1]}(x_i) = w_{jli}^{[1]}\mu_{jli}^{[1]}(x_i) + w_{j,l+1,i}^{[1]}\mu_{j,l+1,i}^{[1]}(x_i). \tag{1.90}$$

Апроксимуючі властивості системи можна поліпшити використовуючи кубічні сплайни [100] замість трикутних функцій належності:

$$\mu_{jli}^{[1]}(x_i)=\begin{cases}\frac{1}{4}\left(2+3\frac{2x_i-c_{jli}^{[1]}-c_{j,l-1,i}^{[1]}}{c_{jli}^{[1]}-c_{j,l-1,i}^{[1]}}-\left(\frac{2x_i-c_{jli}^{[1]}-c_{j,l-1,i}^{[1]}}{c_{jli}^{[1]}-c_{j,l-1,i}^{[1]}}\right)^3\right),\\ \text{якщо } x_i\in\left[c_{j,l-1,i}^{[1]},c_{jli}^{[1]}\right],\\ \frac{1}{4}\left(2-3\frac{2x_i-c_{j,l+1,i}^{[1]}-c_{jli}^{[1]}}{c_{j,l+1,i}^{[1]}-c_{jli}^{[1]}}+\left(\frac{2x_i-c_{j,l+1,i}^{[1]}-c_{jli}^{[1]}}{c_{j,l+1,i}^{[1]}-c_{jli}^{[1]}}\right)^3\right),\\ \text{якщо } x_i\in\left[c_{jli}^{[1]},c_{j,l+1,i}^{[1]}\right],\\ 0 \text{ у протилежному випадку}\end{cases} \tag{1.91}$$

або B-сплайни [101]:

$$\mu_{jli}^{g[1]}(x_i)=\begin{cases}\left.\begin{matrix}1, \text{якщо } x_i\in\left[c_{jli}^{[1]},c_{j,l+1,i}^{[1]}\right]\\ 0 \text{ у протилежному випадку}\end{matrix}\right\} \text{для } g=1\\ \frac{x_i-c_{jli}^{[1]}}{c_{j,l+g-1,i}^{[1]}-c_{jli}^{[1]}}\mu_{jli}^{g-1,[1]}(x_i)+\frac{c_{j,l+g,i}^{[1]}-x_i}{c_{j,l+g,i}^{[1]}-c_{j,l+1,i}^{[1]}}\mu_{j,l+1,i}^{g-1,[1]}\\ \text{для } g>1\end{cases} \tag{1.92}$$

де $\mu_{jli}^{g[1]}(x_i)$ – l-й сплайн g-ого порядку.

Нескладно помітити, що при $g=2$ отримуємо трикутні функції належності (1.88).

B-сплайни, як і трикутні функції належності, забезпечують розбиття Руспіні, але в загальному випадку вони можуть активувати довільне число функцій належності за межами інтервалу $[0,1]$, що може стати у нагоді для подальших каскадів.

Також у якості функцій належності нелінійних синапсів можна використовувати інші структури, такі, як поліноміальні, гармонійні функції, вейвлети, ортогональні функції, тощо. Проте не можна сказати наперед, які з функцій забезпечать кращі результати, тому ідея використання не одного

нейрона, а пулу нейронів з різними функціями належності та активації виглядає доречною та перспективною.

За аналогією до (1.87) визначаємо вихідні сигнали інших каскадів. Так, для другого каскаду можемо записати вихідні сигнали у формі

$$\hat{y}_j^{[2]} = \sum_{i=1}^{n}\sum_{l=1}^{h} w_{jli}^{[2]}\mu_{jli}^{[2]}(x_i) + \sum_{l=1}^{h} w_{j,l,n+1}^{[2]}\mu_{j,l,n+1}^{[2]}\left(\hat{y}^{*[1]}\right), \tag{1.93}$$

вихідні сигнали для нейронів m-ого каскаду

$$\hat{y}_j^{[m]} = \sum_{i=1}^{n}\sum_{l=1}^{h} w_{jli}^{[m]}\mu_{jli}^{[m]}(x_i) + \sum_{p=n+1}^{n+m-1}\sum_{l=1}^{h} w_{jlp}^{[m]}\mu_{jlp}^{[m]}\left(\hat{y}^{*[p-n]}\right) \tag{1.94}$$

Таким чином, каскадна нейронна мережа з нео-фаззі нейронів, що сформована m каскадами, містить $h(\sum_{p=1}^{m-1} p)$ параметрів.

Введемо вектор функцій належності для j-ого нео-фаззі нейрона m-ого каскаду

$$\begin{aligned}&\mu_j^{[m]}(k)\\ &= \left(\mu_{j11}^{[m]}(x_1(k), \dots, \mu_{jh1}^{[m]}(x_1(k), \mu_{j12}^{[m]}(x_2(k)), \dots, \mu_{jh2}^{[m]}(x_2(k)), \dots,\right.\\ &\left.\mu_{jhn}^{[m]}\left(x_i(k)\right), \dots, \mu_{j1,n+1}^{[m]}\left(\hat{y}^{*[1]}(k)\right), \dots, \mu_{jh,n+m-1}^{[m]}\left(\hat{y}^{*[m-1]}(k)\right)\right)^T\end{aligned} \tag{1.95}$$

та відповідний вектор синаптичних вагових коефіцієнтів

$$\begin{aligned}w_j^{[m]} = &\left(w_{j11}^{[m]}, \dots, w_{jh1}^{[m]}, \dots, w_{j12}^{[m]}, \dots, w_{jh2}^{[m]}, \dots,\right.\\ &\left.\dots, w_{jli}^{[m]}, \dots, w_{jhn}^{[m]}, \dots, w_{j1,n+1}^{[m]}, \dots, w_{jh,n+m-1}^{[m]}\right)^T.\end{aligned} \tag{1.96}$$

Тоді можемо компактно записати вихідні сигнали для j-ого нейрону m-ого каскаду

$$\hat{y}_j^{[m]}(k) = w_k^{[m]T}\mu_j^{[m]}(k). \tag{1.97}$$

У такому разі критерій навчання приймає вигляд

$$E_j^{[m]}(k) = \frac{1}{2}\left(e_j^{[m]}(k)\right)^2 = \frac{1}{2}\left(y(k) - w_j^{[m]T}\mu_j^{[m]}(k)\right)^2, \tag{1.98}$$

а мінімізувати його можна використавши модифікацію процедури [102] для ковзного вікна

$$\begin{cases} w_j^{[m]}(k+1) = w_j^{[m]}(k) + \dfrac{e_j^{[m]}(k+1)\mu_j^{[m]}(k+1)}{r_j^{[m]}(k+1)} \\ r_j^{[m]}(k+1) = r_j^{[m]}(k) + \left\|\mu_j^{[m]}(k+1)\right\|^2 - \left\|\mu_j^{[m]}(k-s)\right\|^2 \end{cases} \tag{1.99}$$

або для випадку, коли $s = 1$,

$$w_j^{[m]}(k+1) = w_j^{[m]}(k) + \frac{e_j^{[m]}(k+1)\mu_j^{[m]}(k+1)}{\left\|\mu_j^{[m]}(k+1)\right\|^2}, \tag{1.100}$$

що збігається з однокроковим оптимальним алгоритмом Качмажа–Уідроу–Хоффа.

Вочевидь, замість (1.99) можна скористатися іншими алгоритмами, як-от експоненційно зважений рекурентний метод найменших квадратів (EWRLSM), що використовується у DENFIS та FLEXFIS. Та варто зауважити, що EWRLSM

може бути нестійким при малому коефіцієнті забування.

При використанні критерія навчання з регуляризуючим параметром замість (1.98) отримуємо остаточний метод навчання нео-фаззі нейрона

$$\begin{cases} w_j^{[m]}(k+1) = w_j^{[m]}(k) + \dfrac{\eta e_j^{[m]}(k+1)\mu_j^{[m]}(k+1)}{r_j^{[m]}(k+1)} + \\ + \dfrac{(1-\eta)\left(w_j^{[m]}(k) - w_j^{[m]}(k-1)\right)}{r_j^{[m]}(k+1)}, \\ r_j^{[m]}(k+1) = r_j^{[m]}(k) + \left\|\mu_j^{[m]}(k+1)\right\|^2 - \left\|\mu_j^{[m]}(k-s)\right\|^2. \end{cases} \quad (1.101)$$

Варто наголосити, що оскільки вихідні сигнали нео-фаззі нейрона лінійно залежать від його синаптичних вагових коефіцієнтів, можна використовувати будь-які методи адаптивної лінійної ідентифікації (наприклад, рекурентний метод найменших квадратів, робасні методи, методи, що ігнорують застарілі данні, тощо), що дозволяє обробляти нестаціонарні сигнали в online режимі.

1.3.4 Розширені нео-фаззі нейрони в якості елементів гібридної каскадної мережі, що еволюціонує

Як зазначалося вище, розглядаючи нелінійний синапс нео-фаззі нейрону з позицій нечіткої логіки, нескладно побачити, що він є вельми схожим на шар фаззіфікування таких нейро-фаззі систем як мережі Такаґі–Суґено–Канґа, Дженґа, Ванґа–Менделя, і, фактично реалізує нечітке висновування Такаґі-Суґено нульового порядку. Та задля поліпшення апроксимуючих властивостей таких систем видається доцільним запропонувати удосконалений нелінійний синапс такий, що реалізує нечітке висновування довільного порядку, далі «розширений нелінійний синапс» (ENS), та зсинтезувати «розширений нео-фаззі нейрон» (ENFN), що містить такі структури замість традиційних нелінійних синапсів NS_i.

Архітектури розширеного нелінійного синапсу та розширеного нео-фазі нейрону наведено на рис. 1.22 та рис. 1.23 відповідно.

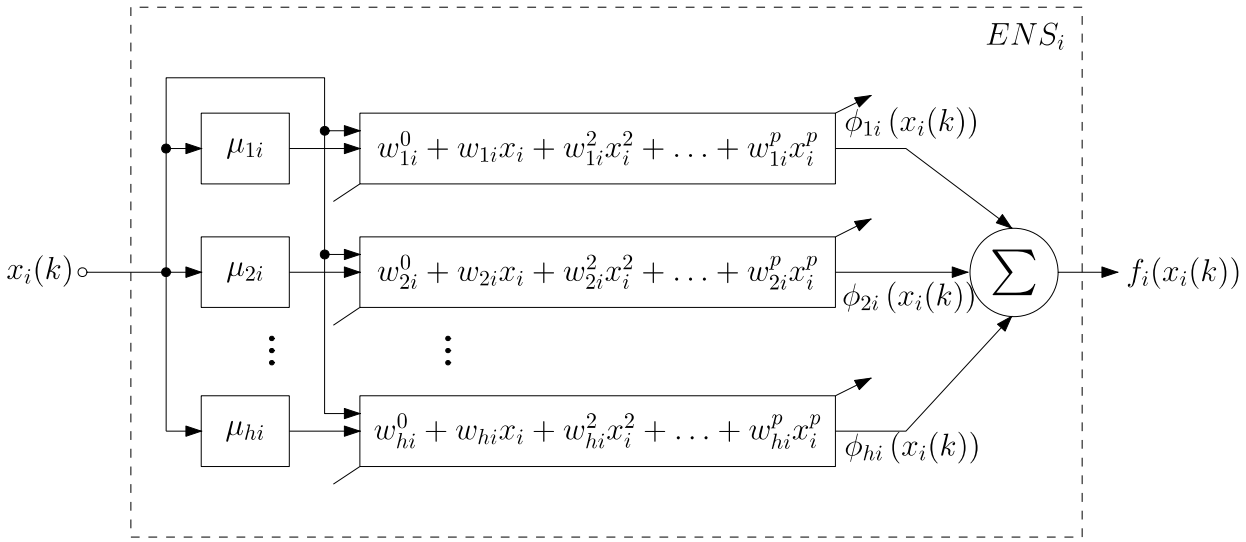


Рисунок 1.22 – Синапс розширеного нео-фаззі нейрону

Вводячі нові змінні

$$\phi_{li}(x_i) = \mu_{li}(x_i)\left(w_{li}^0 + w_{li}^1x_i + w_{li}^2x_i^2 + \cdots + w_{li}^px_i^p\right), \tag{1.102}$$

$$\begin{aligned} f_i(x_i) &= \sum_{l=1}^{h} \mu_{li}(x_i)\left(w_{li}^0 + w_{li}^1x_i + w_{li}^2x_i^2 + \cdots + w_{li}^px_i^p\right) = \\ &= w_{1i}^0\mu_{1i}(x_i) + w_{1i}^1x_i\mu_{1i}(x_i) + \cdots + w_{1i}^px_i^p\mu_{1i}(x_i) + \\ &\quad + w_{2i}^0\mu_{2i}(x_i) + w_{2i}^px_i^p\mu_{2i}(x_i) + \cdots + w_{hi}^px_i^p\mu_{hi}(x_i) \end{aligned} \tag{1.103}$$

$$\begin{aligned} \tilde{\mu}_i(x_i) = \big(&\mu_{1i}(x_i), x_i\left(\mu_{1i}(x_i)\right), \ldots, x_i^p\left(\mu_{1i}(x_i)\right), \\ &\mu_{2i}(x_i), \ldots, x_i^p\left(\mu_{2i}(x_i)\right), \ldots, x_i^p\left(\mu_{hi}(x_i)\right)\big)^T, \end{aligned} \tag{1.104}$$

можна представити вихідні сигнали розширеного нео-фаззі нейрона у вигляді

$$f_i(x_i) = w_i^T \tilde{\mu}_i(x_i), \tag{1.105}$$

$$\hat{y} = \sum_{i=1}^{n} f_i(x_i) = \sum_{i=1}^{n} w_i^T \tilde{\mu}_i(x_i) = \tilde{w}^T \tilde{\mu}(x), \tag{1.106}$$

де $\tilde{w}^T = \left(w_1^T, \ldots, w_i^T, \ldots, w_n^T\right)^T$;

$\tilde{\mu}^T = \left(\tilde{\mu}_1^T(x_1), \ldots, \tilde{\mu}_i^T(x_i), \ldots, \tilde{\mu}_n^T(x_n)\right)^T$.

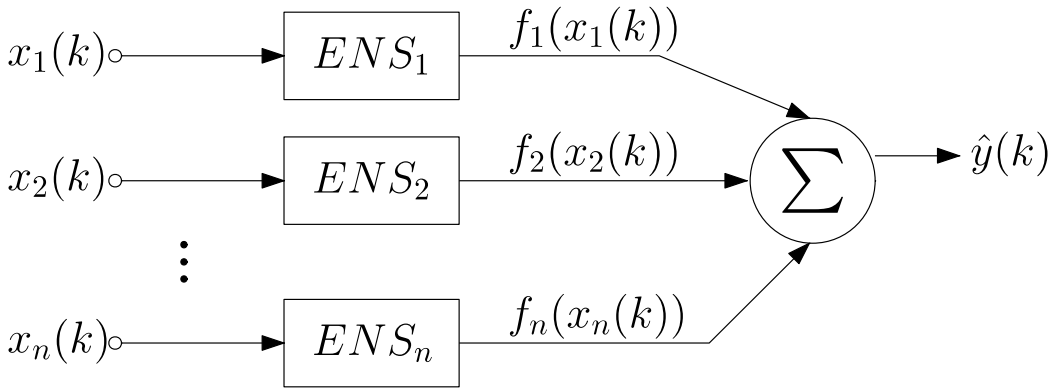


Рисунок 1.23 – Розширений нео-фаззі нейрон

Таким чином, $ENFN$ містить $(p+1)hn$ вагових коефіцієнтів та реалізує нечітке висновування Такагі-Суґено p-ого порядку, а висновування, що його реалізує кожний розширений нелінійний синапс ENS_i можна записати у формі

$$\text{Якщо } x_i \text{ це } X_{li} \text{тоді вихід} - w_{li}^0 + w_{li}^1 x_i + w_{li}^p x_p, l = 1,2, \ldots, h, \tag{1.107}$$

що збігається з нечітким висновуванням Такагі-Суґено p-ого порядку.

Коли подати векторний сигнал $x(k)$ на вхід ENFN першого каскаду, на виході отримуємо скалярне значення

$$\hat{y}^{[1]}(k) = \tilde{w}^{[1]T}(k-1)\tilde{\mu}^{[1]}\big(x(k)\big), \tag{1.108}$$

що відрізняється від виразу (1.107) для звичайних NFN тим, що містить у $p+1$ більше параметрів, що корегуються.

Вочевидь, будь-які методи навчання нео-фаззі нейронів підійдуть і для розширених нео-фаззі нейронів. Так, вирази (1.99) та (1.100) для j-ого нейрону m-ого каскаду приймають вигляд

$$\begin{cases} \widetilde{w}_j^{[m]}(k+1) = \widetilde{w}_j^{[m]}(k) + \dfrac{e_j^{[m]}(k+1)\tilde{\mu}_j^{[m]}(k+1)}{\tilde{r}_j^{[m]}(k+1)}, \\ \tilde{r}_j^{[m]}(k+1) = \tilde{r}_j^{[m]}(k) + \left\|\tilde{\mu}_j^{[m]}(k+1)\right\|^2 - \left\|\tilde{\mu}_j^{[m]}(k-s)\right\|^2 \end{cases} \tag{1.109}$$

та

$$\widetilde{w}_j^{[m]}(k+1) = \widetilde{w}_j^{[m]}(k) + \frac{e_j^{[m]}(k+1)\tilde{\mu}_j^{[m]}(k+1)}{\left\|\tilde{\mu}_j^{[m]}(k+1)\right\|^2} \tag{1.110}$$

відповідно.

1.3.5 Оптимізація пулу нео-фаззі нейронів

Вихідні сигнали, згенеровані нейронами пулу кожного з каскадів, можна об'єднати у окремому вузлі-нейроні $GN^{[m]}$, з точністю $\hat{y}_j^{*[m]}(k)$, не меншою від точності будь-якого нейрону пулу $\hat{y}_j^{[m]}(k)$. Це завдання можна вирішити за допомогою підходу, пов’язаного з ансамблями нейронних мереж.

Хоча відомі алгоритми не призначені для роботи в онлайн-режимі, варто розглянути методи адаптивного узагальнюючого прогнозування [103].

Введемо вектор вхідних сигналів для m-ого каскаду:

$$\hat{y}_j^{[m]}(k) = \left(\hat{y}_1^{[m]}(k), \hat{y}_2^{[m]}(k), \dots, \hat{y}_q^{[m]}(k)\right)^T. \quad (1.111)$$

Тоді оптимальний вихідний сигнал, що його генерує нейрон $GN^{[m]}$ (що, власне, є адаптивним лінійним асоціатором), можна записати у формі

$$\hat{y}_j^{*[m]}(k) = \sum_{j=1}^{q} c_j^{[m]} \hat{y}_j^{[m]}(k) = c^{[m]T} \hat{y}^{[m]}(k) \quad (1.112)$$

з обмеженнями на незміщенність

$$\sum_{j=1}^{q} c_j^{[m]} = E^T c^{[m]} = 1, \quad (1.113)$$

де $c^{[m]} = \left(c_1^{[m]}, c_2^{[m]}, .., c_q^{[m]}\right)^T$;

$E = (1,1, \dots,1)^T$ – $(q \times 1)$-вектори.

Введемо критерій навчання на ковзному вікні

$$E^{[m]}(k) = \frac{1}{2} \sum_{\tau=k-s+1}^{k} \left(y(\tau) - \hat{y}^{*[m]}(\tau)\right)^2 = \\ = \frac{1}{2} \sum_{\tau=k-s+1}^{k} \left(y(\tau) - c^{[m]T} \hat{y}^{[m]}(\tau)\right)^2, \quad (1.114)$$

зважаючи на обмеження (1.113), функція Лаґранжа матиме вигляд

$$L^{[m]}(k) = E^{[m]}(k) - \lambda\left(1 - E^T c^{[m]}\right), \quad (1.115)$$

де λ – невизначений Лагранжів множник.

Мінімізуючи (1.115) відносно $c^{[m]}$, отримуємо

$$\begin{cases} \hat{y}^{*[m]}(k+1) = \dfrac{\hat{y}^{[m]T}(k+1)P^{[m]}(k+1)E}{E^T P^{[m]}(k+1)E}, \\ P^{[m]}(k+1) = \left(\sum\limits_{\tau=k-s+2}^{k+1} \hat{y}^{[m]}(\tau)\hat{y}^{[m]T}(\tau) \right)^{-1} \end{cases} \tag{1.116}$$

або у рекурентній формі

$$\begin{cases} \tilde{P}^{[m]}(k+1) = P^{[m]}(k) - \dfrac{P^{[m]}(k)\hat{y}^{[m]}(k+1)\hat{y}^{[m]T}(k+1)P^{[m]}(k)}{1+\hat{y}^{[m]T}(k+1)P^{[m]}(k)\hat{y}^{[m]}(k+1)}, \\ P^{[m]}(k+1) = \tilde{P}^{[m]}(k+1) + \\ \qquad + \dfrac{\tilde{P}^{[m]}(k+1)\hat{y}(k-s+1)\hat{y}^{[m]T}(k-s+1)\tilde{P}^{[m]}(k+1)}{1-\hat{y}^{[m]T}(k-s+1)\tilde{P}^{[m]}(k+1)\hat{y}^{[m]}(k-s+1)}, \\ \hat{y}^{*[m]}(k+1) = \dfrac{\hat{y}^{[m]T}(k+1)P^{[m]}(k+1)E}{E^T P^{[m]}(k+1)E}. \end{cases} \tag{1.117}$$

У випадку одиничного ковзного вікна $s = 1$, (1.116) та (1.117) приймають доволі простий вигляд:

$$\begin{aligned} \hat{y}^{*[m]}(k+1) = \frac{\hat{y}^{[m]T}(k+1)\hat{y}^{[m]}(k+1)}{E^T\hat{y}^{[m]}(k+1)} = \frac{\left\|\hat{y}^{[m]}(k+1)\right\|^2}{E^T\hat{y}^{[m]}(k+1)} = \\ = \frac{\sum_{j=1}^{q}\left(\hat{y}^{[m]}(k+1)\right)^2}{\sum_{j=1}^{q}\hat{y}^{[m]}(k+1)}. \end{aligned} \tag{1.118}$$

Важливо зазначити, що навчання як нео-фаззі нейронів, так і нейронів-узагальнювачів можна організувати в онлайн-режимі. Таким чином, вагові коефіцієнти нейронів попередніх каскадів (на відміну від CasCorLA) можна не заморожувати, а постійно корегувати. Так само, число каскадів не має бути

фіксованим і може змінюватись у часі, що відрізняє пропоновану нейронну мережу від інших відомих каскадних систем.

1.4 Багатовимірна каскадна нео-фаззі система, що еволюціонує

Задача апроксимації та екстраполяції багатовимірних часових рядів доволі часто виникає у багатьох технічних, медико-біологічних та інших дослідженнях, де якість прийнятих рішень істотно залежить від точності синтезованих прогнозів. У багатьох реальних задачах часові ряди характеризуються високим рівнем нелінійності та нестаціонарності своїх параметрів, наявністю аномальних викидів. Зрозуміло, що традиційні методи аналізу часових рядів, засновані на регресійному, кореляційному та інших подібних підходах, що мають на меті апріорну наявність репрезентативної вибірки спостережень, є неефективними. Альтернативою традиційним статистичним методам може слугувати математичний апарат обчислювального інтелекту, зокрема штучні нейронні мережі та нейро-фаззі-системи, завдяки своїм універсальним апроксимувальним властивостям. Водночас з апроксимувальних властивостей зовсім не витікають екстраполюючі, оскільки врахування давньої передісторії для побудови прогнозувальної моделі може погіршити якість прогнозу. У зв’язку з цим під час оброблення нестаціонарних процесів треба відмовитися від процедур навчання, що базуються на зворотному поширенні похибки (багатошарові персептрони, рекурентні нейронні мережі, адаптивні нейромережеві системи нечіткого виведення – ANFIS або методі найменших квадратів (радіально-базисні та функціонально пов’язані нейронні мережі) та скористатися процедурами на основі локальних критеріїв та «короткої» пам’яті. При цьому використані алгоритми навчання мусять забезпечувати не лише високу швидкодію, але й фільтруючі якості для придушення стохастичної «шумової» компоненти в оброблюваному сигналі. У зв’язку з цим синтез спеціалізованих гібридних систем обчислювального інтелекту для розв’язання задач прогнозування істотно нестаціонарних часових рядів за умов невизначеності, що забезпечують разом з

високою швидкістю навчання і фільтрацію завад, є досить цікавою та перспективною задачею.

Таким чином, цей розділ присвячено синтезу багатовимірної гібридної системи обчислювального інтелекту, що здатна реалізувати нелінійне відображення $R^n \to R^g$ в онлайн режимі.

1.4.1 Багатовимірна каскадна система, що еволюціонує, побудована на нео-фаззі нейронах

Для вирішення задачі прогнозування та ідентифікації багатовимірних даних в умовах апріорної і поточної структурної та параметричної невизначеності як ніколи доречні переваги каскадно-кореляційної архітектури, адже системи з такою архітектурою успадковують всі переваги елементів, які використовуються в їх вузлах, а в процесі навчання автоматично підбирається необхідна кількість каскадів для того, щоб отримати модель адекватної складності для вирішення поставленого завдання. Однак, слід зазначити, що каскадно-кореляційна мережа у формі, що її запропонували С. Фальман і К. Лєб'єр, є системою з одним виходом, тобто не здатна реалізувати нелінійне відображення $R^n \to R^g$. Це досить серйозне обмеження, оскільки більшість практичних завдань містять кілька вихідних сигналів. Тож пропонуймо такі модифікації до архітектури каскадно-кореляційної мережі CasCorLA:

1. Замість елементарних персептронів Розенблатта використовувати нео-фаззі нейрони.

2. Кількість нейронів у кожному каскаді відтепер має дорівнювати розмірності вектору вихідного сигналу системи.

Схему пропонованої архітектури наведено на рис. 1.24.

Тоді вихідний сигнал системи формується з векторів, що його складають вихідні сигнали кращих нейронів останнього каскаду:

$$\hat{y}(k) = \left(\hat{y}_1^{*[m]}(k), \hat{y}_2^{*[m]}(k), \ldots \hat{y}_g^{*[m]}(k)\right)^T \tag{1.119}$$

де g – кількість елементів вихідного вектору даних, що їх треба спрогнозувати чи ідентифікувати.

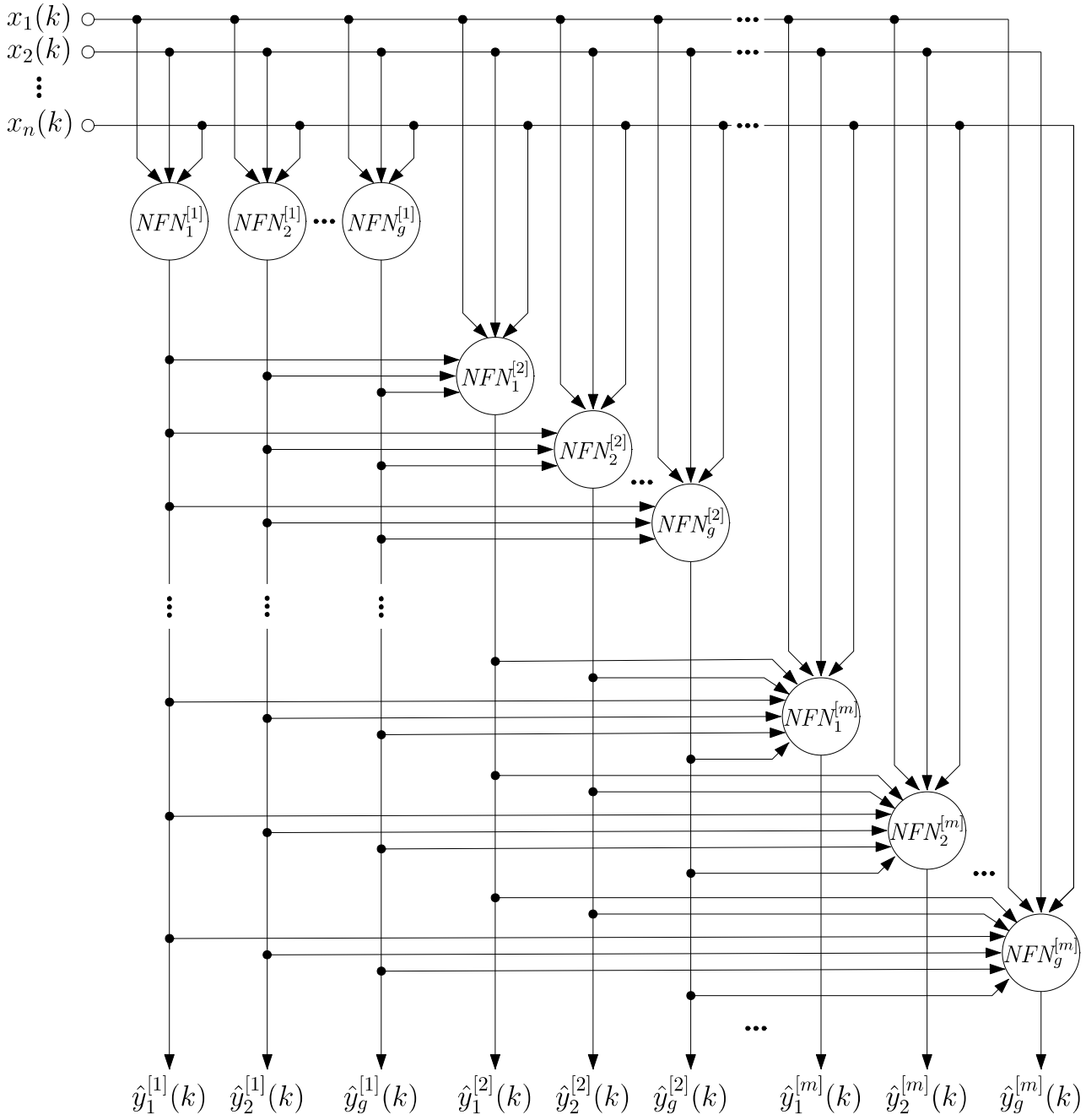


Рисунок 1.24 – Архітектура гібридної MIMO системи, побудованої на нео-фаззі нейронах

Для кожного з нео-фаззі нейронів системи в якості функцій належності можна використовувати трикутні конструкції:

$$\mu_{dli}^{[1]j}(x_i) = \begin{cases} \dfrac{x_i - c_{d,l-1,i}^{[1]j}}{c_{dli}^{[1]j} - c_{d,l-1,i}^{[1]j}}, & \text{якщо } x_i \in \left[c_{d,l-1,i}^{[1]j}, c_{dli}^{[1]j}\right], \\ \dfrac{c_{d,l+1,i}^{[1]j} - x_i}{c_{d,l+1,i}^{[1]j} - c_{dli}^{[1]}}, & \text{якщо } x_i \in \left[c_{dli}^{[1]j}, c_{d,l+1,i}^{[1]j}\right], \\ 0 & \text{у протилежному випадку,} \end{cases} \tag{1.120}$$

кубічні сплайни:

$$\mu_{dli}^{[1]j}(x_i) = \begin{cases} \dfrac{1}{4}\left(2 + 3\dfrac{2x_i - c_{dli}^{[1]j} - c_{d,l-1,i}^{[1]j}}{c_{dli}^{[1]j} - c_{d,l-1,i}^{[1]j}} - \left(\dfrac{2x_i - c_{dli}^{[1]j} - c_{d,l-1,i}^{[1]j}}{c_{dli}^{[1]j} - c_{d,l-1,i}^{[1]j}}\right)^3\right), \\ \text{якщо } x_i \in \left[c_{d,l-1,i}^{[1]j}, c_{dli}^{[1]j}\right], \\ \dfrac{1}{4}\left(2 - 3\dfrac{2x_i - c_{d,l+1,i}^{[1]j} - c_{dli}^{[1]j}}{c_{d,l+1,i}^{[1]j} - c_{dli}^{[1]j}} + \left(\dfrac{2x_i - c_{d,l+1,i}^{[1]j} - c_{dli}^{[1]j}}{c_{d,l+1,i}^{[1]j} - c_{dli}^{[1]j}}\right)^3\right), \\ \text{якщо } x_i \in \left[c_{dli}^{[1]j}, c_{d,l+1,i}^{[1]j}\right], \\ 0 \text{ у протилежному випадку} \end{cases} \tag{1.121}$$

або B-сплайни:

$$\mu_{dli}^{g[1]j}(x_i) = \begin{cases} \left.\begin{matrix} 1, \text{якщо } x_i \in \left[c_{dli}^{[1]j}, c_{d,l+1,i}^{[1]j}\right] \\ 0 \text{ у протилежному випадку} \end{matrix}\right\} \text{для } g = 1, \\ \dfrac{x_i - c_{dli}^{[1]j}}{c_{d,l+g-1,i}^{[1]j} - c_{dli}^{[1]j}} \mu_{dli}^{g-1,[1],j}(x_i) + \dfrac{c_{d,l+g,i}^{[1]j} - x_i}{c_{d,l+g,i}^{[1]j} - c_{j,l+1,i}^{[1]}} \mu_{d,l+1,i}^{g-1,[1],j} \\ \text{для } g > 1, \end{cases} \tag{1.122}$$

де d – індекс елементу вихідного вектору даних.

Варто зауважити, що всі ці конструкції задовольняють умовам одиничного розбиття Руспіні.

Запишемо вихідний сигнал j-ого нео-фаззі нейрону d-ого виходу першого каскаду у вигляді

$$\begin{cases} \hat{y}_d^{[1]j}(k) = \sum_{i=1}^{n} f_{di}^{[1]j}\big(x(k)\big) = \sum_{i=1}^{n}\sum_{l=1}^{h} w_{dli}^{[1]j}\mu_{dli}^{[1]j}\big(x_i(k)\big), \\ \text{якщо } x_i - \text{ це } X_{dli}^{j}, \text{тоді вихід} - w_{dli}^{[1]j}, \end{cases} \tag{1.123}$$

вихідні сигнали нео-фаззі нейронів другого каскаду:

$$\hat{y}_d^{[2]j}(k) = \sum_{i=1}^{n}\sum_{l=1}^{h} w_{dli}^{[2]j}\mu_{dli}^{[2]j}\big(x_i(k)\big) + \\ + \sum_{d=1}^{g}\sum_{l=1}^{h} w_{dl,n+1}^{[2]j}\mu_{dl,n+1}^{[2]j}\left(\hat{y}_d^{*[1]}(k)\right) \;\forall\, d = 1,2,\dots,g, \tag{1.124}$$

вихідні сигнали m-ого каскаду:

$$\hat{y}_d^{[m]j}(k) = \sum_{i=1}^{n}\sum_{l=1}^{h} w_{dli}^{[m]j}\mu_{dli}^{[m]j}\big(x_i(k)\big) + \\ + \sum_{d=1}^{g}\sum_{p=n+1}^{n+m-1}\sum_{l=1}^{h} w_{dlp}^{[m]j}\mu_{dlp}^{[m]j}\left(\hat{y}_d^{*[p-n]}(k)\right) \;\forall\, d = 1,2,\dots,g. \tag{1.125}$$

Введемо до розгляду надалі вектор функцій належності j-ого нейрону d-ого виходу m-ого каскаду:

$$\mu_d^{[m]j} = \left(\mu_{d11}^{[m]j}(x_1(k)), \dots, \mu_{dh1}^{[m]j}(x_1(k)), \mu_{d12}^{[m]j}(x_2(k)), \dots,\right.$$

$$\dots, \mu_{dh2}^{[m]j}(x_2(k)), \dots, \mu_{dli}^{[m]j}(x_i(k)), \dots, \mu_{dhn}^{[m]j}(x_n(k)),$$
$$\left.\dots, \mu_{dh,n+1}^{[m]j}\left(\hat{y}^{*[1]}(k)\right), \dots, \mu_{dh,n+1}^{[m]j}\left(\hat{y}^{*[m-1]}(k)\right)\right)^T \quad (1.126)$$

та відповідний йому вектор синаптичних вагових коефіцієнтів

$$w_d^{[m]j} = \left(w_{d11}^{[m]j}, \dots, w_{dh1}^{[m]j}, w_{d12}^{[m]j}, \dots, w_{dh2}^{[m]j}, \dots, w_{dli}^{[m]j},\right.$$
$$\left.\dots, w_{dhn}^{[m]j}, w_{dh,n+1}^{[m]j}, \dots, w_{dh,n+m-1}^{[m]j}\right)^T \quad (1.127)$$

щоб записати вихідний сигнал системи у компактній формі:

$$\hat{y}_d^{[m]j}(k) = \left(w_d^{[m]j}\right)^T \mu_d^{[m]j}(k) \quad (1.128)$$

Для навчання нео-фаззі нейронів може бути використаний будь-який з методів адаптивної ідентифікації, що ми пропонували використовувати для навчання вузлів одновимірної нео-фаззі системи.

Так, корегувати вагові коефіцієнти можна за допомогою експоненційно-зваженого рекурентного методу найменших квадратів, що можна записати у вигляді:

$$\begin{cases} w_d^{[m]j}(k+1) = w_d^{[m]j}(k) + \\ \quad + \dfrac{P_d^{[m]j}(k)\left(y^d(k+1) - \left(w_d^{[m]j}(k)\right)^T \mu_d^{[m]j}(k+1)\right)}{\alpha + \left(\mu_d^{[m]j}(k+1)\right)^T P_d^{[m]j}(k)\mu_d^{[m]j}(k+1)}\mu_d^{[m]j}(k+1), \\ P_d^{[m]j}(k+1) = \dfrac{1}{\alpha}\left(P_d^{[m]j}(k) - \dfrac{P_d^{[m]j}(k)\mu_d^{[m]j}(k+1)\left(\mu_d^{[m]j}(k+1)\right)^T P_d^{[m]j}(k)}{\alpha + \left(\mu_d^{[m]j}(k+1)\right)^T P_d^{[m]j}(k)\mu_d^{[m]j}(k+1)}\right) \end{cases} \quad (1.129)$$

де $y^d(k+1), d = 1,2,\dots,q$ – зовнішній навчальний сигнал;

$0 < \alpha \leq 1$ – фактор забування;

Також можна скористатися градієнтним методом навчання, що, як зазначалося, відрізняється як згладжувальними, так і слідкуючими властивостями:

$$\begin{cases} w_d^{[m]j}(k+1) = w_d^{[m]j} + \dfrac{y^d(k+1) - \left(w_d^{[m]j}(k)\right)^T \mu_d^{[m]j}(k+1)}{r^{[1]j}(k+1)}, \\ r_d^{[m]j}(k+1) = \alpha r_d^{[m]j}(k) + \left\|\mu_d^{[m]j}(k+1)\right\|^2, \ 0 \leq \alpha \leq 1. \end{cases} \quad (1.130)$$

1.4.2 Оптимізація пулу нео-фаззі нейронів багатовимірної каскадної системи, що еволюціонує

Оскільки за мету було поставлено синтез такої багатовимірної каскадної системи, що б могла працювати саме в режимі реального часу, було б дуже доречно, якби система могла самостійно визначати найліпшу кількість функцій належності та їх форму, адже ці параметри також можуть змінюватися у часі. Тому у цьому підрозділі пропонується у кожному каскаді збільшити кількість нео-фаззі нейронів до такої, що є кратною (а не дорівнює, як пропонувалося у попередньому підрозділі) розмірності вектору вихідного сигналу та ввести

узагальнюючі нейрони, що для пулу кожного каскаду визначатимуть локально оптимальні вихідні сигнали (тут під «локально оптимальним вихідними сигналами» слід розуміти сигнали, оптимальні у конкретний поточний момент часу). Таким чином, коли g–розмірність вихідного векторного сигналу, а z– кількість відмінних типів нейронів (що відрізняються за кількістю чи характером функцій належності) системи, у пулі першого каскаду знаходиться zq нео-фаззі нейронів та g нейронів-узагальнювачів $GN_d^{[1]}$, пул другого каскаду містить $z(g+1)$ нейронів та $g+1$ нейронів $GN_d^{[2]}$, останній каскад – $z(g+m-1)$ нейронів та $g+m-1$ нейронів $GN_d^{[m]}$.

Схему такої оптимізованої MIMO (Multiple Input Multiple Output) архітектури зображено на рис. 1.25.

1.4.3 Метод визначення локально оптимальних вихідних сигналів пулу нео-фаззі нейронів багатовимірної каскадної системи, що еволюціонує

Вихідні сигнали нейронів пулу кожного каскаду пропонується об’єднати узагальнюючим нейроном $GN^{[m]}$, що його було введено вище.

Таким чином, у кожному каскаді системи маємо g $GN_d^{[m]}$ елементів, що узагальнюють вихідні сигнали нейронів пулу для кожного елементу вихідного вектору:

$$\hat{y}^{*[m]}(k) = \left(\hat{y}_1^{[m]}(k), \hat{y}_2^{[m]}(k), \ldots, \hat{y}_q^{[m]}(k)\right)^T. \qquad (1.131)$$

До першого узагальнюючого елементу першого каскаду $GN_1^{[1]}$ подаються сигнали

$$\left(\hat{y}_1^{[1]}(k), \hat{y}_{g+1}^{[1]}(k), \ldots, \hat{y}_{2g+1}^{[1]}(k), \ldots, \hat{y}_{(z-1)(g+1)}^{[1]}(k)\right)^T, \qquad (1.132)$$

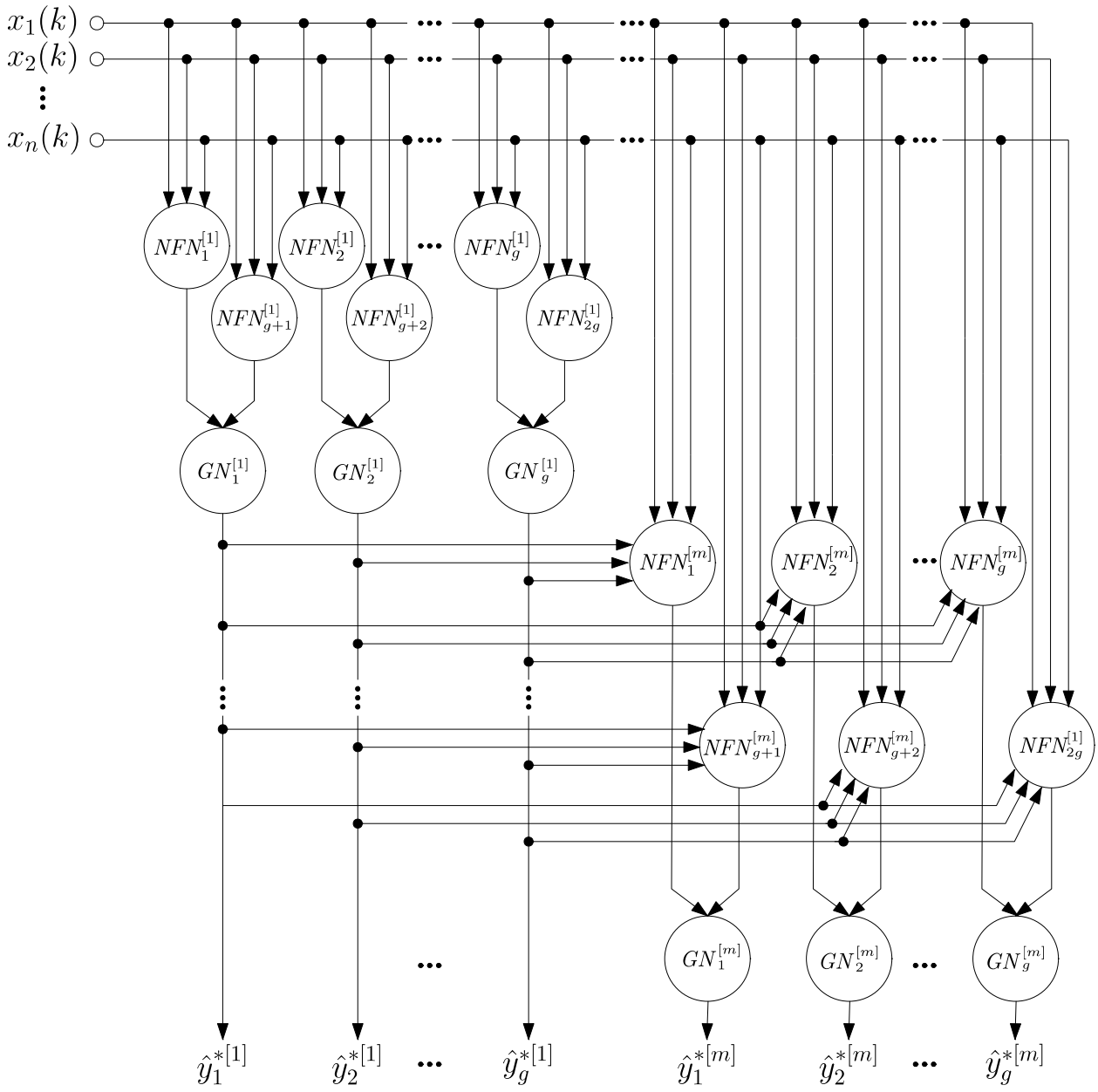


Рисунок 1.25 – Архітектура гібридної оптимізованої MIMO системи, побудованої на нео-фаззі нейронах

до другого узагальнювача $GN_2^{[1]}$ –

$$\left(\hat{y}_2^{[1]}(k), \hat{y}_{g+2}^{[1]}(k), \dots, \hat{y}_{2g+2}^{[1]}(k), \dots, \hat{y}_{(z-1)(g+2)}^{[1]}(k)\right)^T, \qquad (1.133)$$

і, нарешті, вектор вхідних сигналів останнього узагальнюючого елементу першого каскаду $GN_g^{[1]}$:

$$\left(\hat{y}_g^{[1]}(k), \hat{y}_{2g}^{[1]}(k), \ldots, \hat{y}_{(z-1)g}^{[1]}(k)\right)^T. \tag{1.134}$$

Нагадаємо, що точність вихідного сигналу узагальнюючих елементів має бути не гіршою за точність будь-якого сигналу, що узагальнюється (подається на вхід до $GN_g^{[m]}$).

Рекурентна форма методу навчання на ковзному вікні елементів $GN_g^{[m]}$ кожного каскаду має вигляд

$$\begin{cases} \tilde{P}_d^{[m]}(k+1) = \tilde{P}_d^{[m]}(k) - \dfrac{P_d^{[m]}(k)\hat{y}_d^{[m]}(k+1)\hat{y}_d^{[m]T}(k+1)P_d^{[m]}(k)}{1+\hat{y}_d^{[m]}(k+1)P_d^{[m]}(k)\hat{y}_d^{[m]}(k+1)}, \\ P_d^{[m]}(k+1) = P_d^{[m]}(k) + \\ \quad + \dfrac{\tilde{P}_d^{[m]}(k+1)\hat{y}_d^{[m]}(k-s+1)\hat{y}_d^{[m]T}(k-s+1)\tilde{P}_d^{[m]}(k+1)}{1-\hat{y}_d^{[m]T}(k-s+1)\tilde{P}_d^{[m]}(k+1)\hat{y}_d^{[m]}(k-s+1)}, \\ \hat{y}_d^{*[m]}(k+1) = \dfrac{\hat{y}_d^{[m]T}(k+1)P_d^{[m]}(k+1)E}{E^T P_d^{[m]}(k+1)E}, \end{cases} \tag{1.135}$$

а у випадку, коли $s = 1$:

$$\hat{y}_d^{*[m]}(k+1) = \frac{\hat{y}_d^{[m]T}(k+1)\hat{y}_d^{[m]}(k+1)}{E^T\hat{y}_d^{[m]}(k+1)} = \frac{\left\|\hat{y}_d^{[m]}(k+1)\right\|^2}{E^T\hat{y}_d^{[m]}(k+1)} = \frac{\sum_{j=1}^{q}\left(\hat{y}_d^{[m]}(k+1)\right)^2}{\sum_{j=1}^{q}\hat{y}_d^{[m]}(k+1)}. \tag{1.136}$$

1.4.4 Багатовимірна каскадна система, що еволюціонує, побудована на багатовимірних нео-фаззі нейронах

Архітектура багатовимірної каскадної системи, яка ґрунтується на звичайних нео-фаззі нейронах є надмірною, адже вектор вхідних сигналів $x(k)$ (для першого каскаду) подається на однотипні нелінійні синапси $NS_{di}^{[1]j}$ нео-фаззі нейронів, кожен з яких на виході генерує сигнал $\hat{y}_d^{[1]j}(k), d = 1,2,..,g$. У результаті компоненти вихідного вектора

$$\hat{y}^{[1]j}(k) = \left(\hat{y}_1^{[1]j}(k), \hat{y}_2^{[1]j}(k), \ldots, \hat{y}_g^{[1]j}(k)\right)^T \tag{1.137}$$

обчислюються незалежно один від одного, хоча при цьому

$$\mu_{1il}\big(x_i(k)\big) = \mu_{2il}\big(x_i(k)\big) = \mu_{jil}\big(x_i(k)\big) = \mu_{nil}\big(x_i(k)\big). \tag{1.138}$$

Надмірність архітектури, що її наведено на рис. 1.25, проілюстрована на рис. 1.26, де позначені неодноразово обчислювані тотожні значення функцій належності μ_{111} та μ_{j11}, тотожні μ_{12n} та μ_{j2n}. Уникнути цього можна, якщо ввести до розгляду багатовимірний нео-фаззі нейрон.

Вузлами багатовимірного нео-фаззі нейрону MNFN (схему наведено на рис. 1.27) є складені нелінійні синапси $MNS_i^{[1]j}$, кожен з яких містить h функцій належності $\mu_{li}^{[1]j}$ та gh настроюваних синаптичних вагових коефіцієнтів, але тільки hn функцій належності, що в g разів менше, ніж у випадку, коли каскад сформований із звичайних нео-фаззі нейронів. Введемо надалі до розгляду

$(hn \times 1)$- вектор функцій належності

$$\mu^{[1]j}(k) = \left(\mu_{11}^{[1]j}\left(x_1(k)\right), \mu_{21}^{[1]j}\left(x_1(k)\right), \dots, \mu_{h1}^{[1]j}\left(x_1(k)\right), \dots \right.$$
$$\left. \dots, \mu_{hn}^{[1]j}\left(n(k)\right)\right)^T \quad (1.139)$$

та $(g \times hn)$ - матрицю синаптичних вагових коефіцієнтів

$$W^{[1]j} = \begin{pmatrix} w_{111}^{[1]j} & w_{112}^{[1]j} & \cdots & w_{1li}^{[1]j} & \cdots & w_{1hn}^{[1]j} \\ w_{211}^{[1]j} & w_{212}^{[1]j} & \cdots & w_{2li}^{[1]j} & \cdots & w_{2hn}^{[1]j} \\ \vdots & \vdots & & \vdots & & \vdots \\ w_{g11}^{[1]j} & w_{g12}^{[1]j} & \cdots & w_{gli}^{[1]j} & \cdots & w_{ghn}^{[1]j} \end{pmatrix} \quad (1.140)$$

і запишемо сигнал на виході $MN^{[1]j}$ k-й момент часу у вигляді

$$\hat{y}^{[1]j}(k) = W^{[1]j}\mu^{[1]j}(k). \quad (1.141)$$

Навчання багатовимірного нео-фаззі нейрону можна реалізувати за допомогою матричної модифікації експоненційно-зваженого рекурентного методу найменших квадратів у формі

$$\begin{cases} W^{[1]j}(k+1) = W^{[1]j}(k) + \\ \qquad + \dfrac{\left(y(k+1) - W^{[1]j}(k)\mu^{[1]j}(k+1)\right)\left(\mu^{[1]j}(k+1)\right)^T P^{[1]j}(k)}{\alpha + \left(\mu^{[1]j}(k+1)\right)^T P^{[1]j}(k)\mu^{[1]j}(k+1)}, \\ P^{[1]j}(k+1) = \dfrac{1}{\alpha}\left(P^{[1]j}(k) - \dfrac{P^{[1]j}(k)\mu^{[1]j}(k+1)\left(\mu^{[1]j}(k+1)\right)^T P^{[1]j}(k)}{\alpha + \left(\mu^{[1]j}(k+1)\right)^T P^{[1]j}(k)\mu^{[1]j}(k+1)}\right), \\ 0 < \alpha \le 1 \end{cases} \quad (1.142)$$

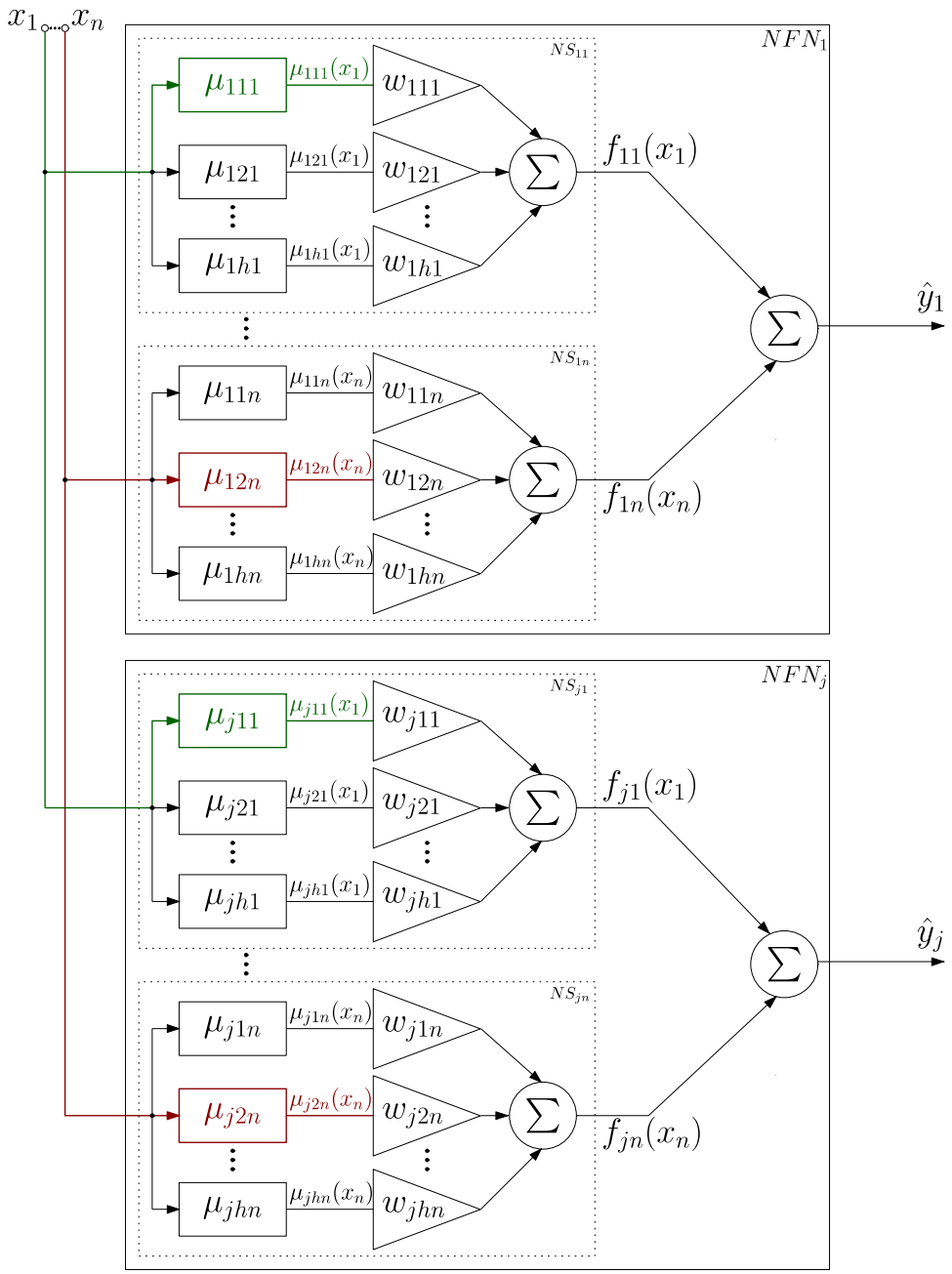


Рисунок 1.26 – Ілюстрація надмірності MIMO системи, побудованої на нео-фаззі нейронах

або багатовимірного варіанту методу (1.130)

$$\begin{cases} W^{[1]j}(k+1) = W^{[1]j}(k) + \dfrac{y(k+1) - W^{[1]j}(k)\mu^{[1]j}(k+1)\left(\mu^{[1]j}(k+1)\right)^T}{r^{[1]j}(k+1)} \\ r^{[1]j}(k+1) \quad = \alpha W^{[1]j}(k) + \left\|\mu^{[1]j}(k+1)\right\|^2, \;\; 0 \le \alpha \le 1. \end{cases}, \quad (1.143)$$

де $y(k+1) = \left(y^1(k+1), y^2(k+1), \ldots, y^g(k+1)\right)^T$.

Аналогічним чином проводиться навчання інших каскадів, при цьому вектор функцій належності m-го каскаду $\mu^{[m]j}(k+1)$ збільшує свою розмірність на $(m-1)g$ компоненти, що їх утворили виходи попередніх каскадів.

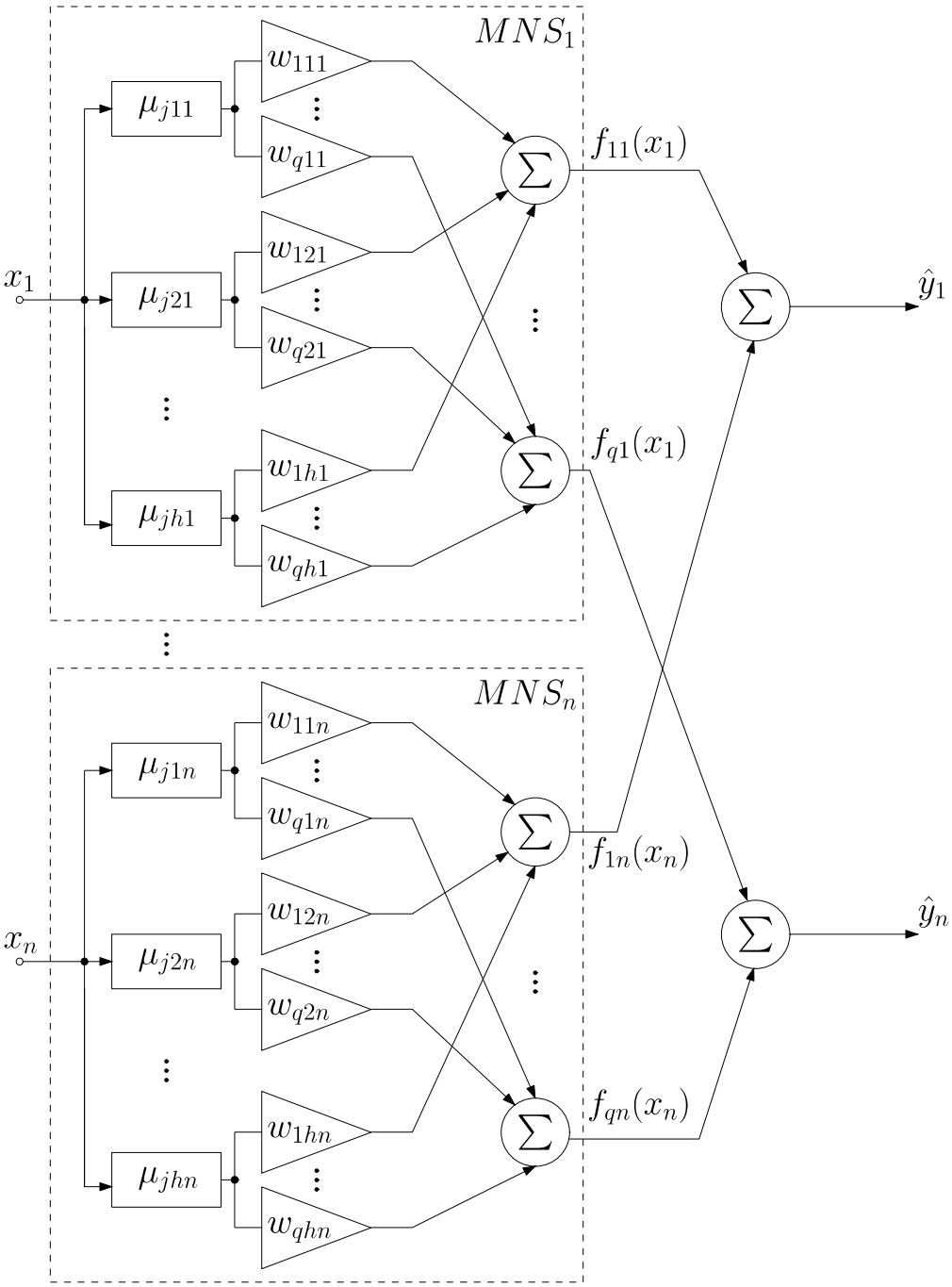


Рисунок 1.27 – Багатовимірний нео-фаззі нейрон

1.4.5 Метод визначення локально оптимального вихідного сигналу пулу багатовимірних нео-фаззі нейронів каскадної системи, що еволюціонує

У цьому підрозділі запропоновано узагальнюючий нейрон $GMN^{[m]}$ та рекурентний метод його навчання, який б об'єднував усі вихідні сигнали нейронів $MNFN^{[m]}$ пулу каскаду у вихідний сигнал

$$\hat{y}^{*[m]}(k) = \left(\hat{y}_1^{*[m]}(k), \hat{y}_2^{*[m]}(k), \ldots, \hat{y}_g^{*[m]}(k)\right)^T \tag{1.144}$$

з точністю не меншою від точності будь-якого з сигналів $\hat{y}^{[m]}(k)$.

Розв'язати це завдання можна, знову скориставшись апаратом невизначених множників Лагранжа та адаптивного багатовимірного узагальненого прогнозування. Введемо до розгляду вихідний сигнал нейрону $GMN^{[m]}$ у вигляді

$$\hat{y}^{*[m]}(k) = \sum_{j=1}^{q} c_j^{[m]} \hat{y}_j^{[m]}(k) = \hat{y}^{[m]}(k) c^{[m]}, \tag{1.145}$$

де $\hat{y}^{[m]}(k) = \left(\hat{y}_1^{[m]}(k), \hat{y}_2^{[m]}(k), \ldots, \hat{y}_q^{[m]}(k)\right)^T$ – $(g \times q)$-матриця;

$c^{[m]}$ – $(q \times 1)$-вектор коефіцієнтів, що відповідають умовам незміщенності:

$$\sum_{j=1}^{q} c_j^{[m]} = E^T c^{[m]} = 1, \tag{1.146}$$

де $E = (1,1, \ldots,1)^T$ – вектор, утворений одиницями.

Введемо критерій навчання

$$E^{[m]}(k) = \sum_{\tau=1}^{k} \left\| y(\tau) - \hat{y}^{[m]}(\tau) c^{[m]} \right\|^2 = \\ = Tr\left(\left(Y(k) - \hat{Y}^{[m]}(k) I \otimes c^{[m]} \right)^T \left(Y(k) - \hat{Y}^{[m]}(k) I \otimes c^{[m]} \right) \right), \tag{1.147}$$

де $Y(k) = \left(y^T(1), y^T(2), \dots, y^T(k) \right)^T$ – $(k \times s)$-матриця спостережень;

I – одинична $(g \times g)$-матриця;

$\otimes$ – символ тензорного добутку;

$$\hat{Y}^{[m]}(k) = \begin{pmatrix} \hat{y}_1^{[m]T}(1) & \hat{y}_1^{[m]T}(1) & \cdots & \hat{y}_q^{[m]T}(1) \\ \hat{y}_1^{[m]T}(2) & \hat{y}_2^{[m]T}(2) & \cdots & \hat{y}_q^{[m]T}(2) \\ \vdots & \vdots & & \vdots \\ \hat{y}_1^{[m]T}(k) & \hat{y}_2^{[m]T}(k) & \cdots & \hat{y}_q^{[m]T}(k) \end{pmatrix}. \tag{1.148}$$

З урахуванням обмежень запишемо функцію Лагранжа

$$L^{[m]}(k) = E^{[m]}(k) + \lambda\left(E^T c^{[m]} - 1 \right) = \\ = \left\| y(\tau) - \hat{y}^{[m]}(\tau) c^{[m]} \right\|^2 + \lambda\left(E^T c^{[m]} - 1 \right) = \\ = \sum_{\tau=1}^{q} r\left(\left(Y(k) - \hat{Y}^{[m]}(k) I \otimes c^{[m]} \right)^T \left(\left(Y(k) - \hat{Y}^{[m]}(k) I \otimes c^{[m]} \right) \right) \right) + \\ + \lambda\left(E^T c^{[m]} - 1 \right) = \\ = Tr\left(V^{[m]T}(k) V^{[m]}(k) + \lambda\left(E^T c^{[m]} - 1 \right) \right), \tag{1.149}$$

де $V^{[m]}(k) = Y(k) - \hat{Y}^{[m]}(k) I \otimes c^{[m]}$ – $(k \times g)$-матриця оновлень.

Розв’язання системи рівнянь Каруша-Куна-Таккера

$$\begin{cases} \nabla_{c^{[m]}} L^{[m]}(k) = \vec{0}, \\ \dfrac{\partial L^{[m]}(k)}{\partial \lambda} = 0 \end{cases} \tag{1.150}$$

призводить до очевидного результату

$$\begin{cases} c^{[m]}(k) = \left(R^{[m]}(k)\right)^{-1} E \left(E^T \left(R^{[m]}(k)\right)^{-1}\right)^{-1}, \\ \lambda = -2E^T \left(R^{[m]}(k)\right)^{-1} E, \end{cases} \tag{1.151}$$

де $R^{[m]}(k) = V^{[m]T}(k)V^{[m]}(k)$.

Таким чином, можна організувати оптимальне об'єднання виходів усіх нейронів пулу кожного каскаду. Зрозуміло, що в якості таких нейронів можуть використовуватися не тільки багатовимірні нео-фаззі нейрони, але й будь-які інші конструкції, що реалізують нелінійне відображення $R^{n+(m-1)g} \to R^{g}$.

1.5 Еволюційна МГУА-нейро-фаззі система з малою кількістю параметрів, що налаштовуються

1.5.1 Архітектура еволюційної багатошарової МГУА-нейро-фаззі системи

Метод групового урахування аргументів (МГУА) розроблявся академіком О. Г. Івахненком та його школою. МГУА є методом індуктивного моделювання і одним з найбільш ефективних методів структурно-параметричної ідентифікації складних об’єктів, процесів і систем за даними спостережень за умов неповноти інформації.

Серед основних переваг методу групового урахування аргументів можна виділити наступні:

– він добре працює на коротких вибірках даних, тобто коли кількість параметрів, що налаштовуються, більша за кількість спостережень;

– система, що заснована на МГУА, конструюється сама у процесі роботи і не потребує точного задання кількості шарів.

Архітектура еволюційної МГУА-системи наведена на рис. 1.28.

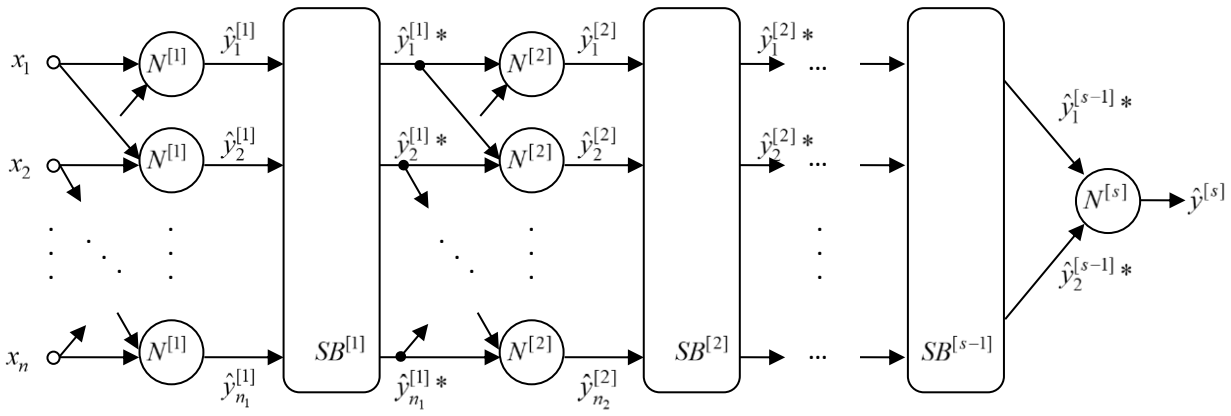


Рисунок 1.28 – Еволюційна МГУА-система

На нульовий (рецепторний) шар системи надходить $(n\times 1)$-вимірний вектор вхідних сигналів $x=(x_1, x_2, ..., x_n)^{\mathrm{T}}$, який далі передається на перший прихованих шар, що містить $n_1 = c_n^2$ вузлів-нейронів, кожен з яких має всього два входи. На виходах вузлів $N^{[1]}$ першого прихованого шару формуються вихідні сигнали $\hat{y}_l^{[1]}$, $l=1, 2, ..., 0,5n(n-1)=c_n^2$. Далі ці сигнали надходять до блоку селекції першого прихованого шару $SB^{[1]}$, який відбирає з множини вихідних сигналів $\hat{y}_l^{[1]}$ n_1* ($n_1*\le n$) найбільш точних з них згідно з прийнятим критерієм, найчастіше середнім квадратом похибки $\sigma^2_{y_l^{[1]}}$.

З цих n_1* найкращих виходів першого прихованого шару $\hat{y}_l^{[1]}*$ формується n_2 (зазвичай $n\le n_2\le 2n$) попарних сполучень $\hat{y}_l^{[1]}*, \hat{y}_p^{[1]}*$, які подаються на другий прихований шар, що утворені нейронами $N^{[2]}$, які є аналогічними до нейронів $N^{[1]}$.

З вихідних сигналів цього шару $\hat{y}_l^{[2]}$ блок селекції другого прихованого шару $SB^{[2]}$ відбирає лише ті, які за точністю, наприклад, за $\sigma^2_{y_l^{[2]}}$, перевершують найкращий сигнал першого прихованого шару $\hat{y}_1^{[1]}*$.

Третій прихований шар формує сигнали, що перевершують за точністю найкращий сигнал $\hat{y}_1^{[2]}*$ і т. д.

Процес еволюції системи відбувається доти, доки блок селекції $SB^{[s-1]}$ не сформує на своєму виході лише два сигнали $\hat{y}_1^{[s-1]}*$ і $\hat{y}_2^{[s-1]}*$. Саме ці два сигнали подаються на єдиний вихідний нейрон $N^{[s]}$, який розраховує вихідний сигнал системи $\hat{y}^{[s]}$.

В якості вузлів МГУА-систем можна використовувати різноманітні типи нейронів, наприклад, N-адаліни, активні [104-106], R- [107-109], Q- [110], спайк-, вейвлет-, нео-фаззі-нейрони та інші подібні блоки систем обчислювального інтелекту, які мають необхідні апроксимуючі можливості і здатність до навчання. При цьому, однак, може бути втрачена головна перевага оригінального МГУА – можливість роботи за умов дуже коротких навчальних вибірок.

1.5.2 Нейро-фаззі система з малою кількістю параметрів, що налаштовуються, в якості вузла МГУА-системи

В якості нейрону МГУА-системи, що розглядається, пропонується використовувати вузол, архітектура якого наведена на рис. 1.29.

Ця архітектура є по суті нейро-фаззі системою Ванга–Менделя з двома входами $x_i(k)$ і $x_j(k)$, п’ятьма послідовно з’єднаними шарами обробки інформації і одним виходом $\hat{y}_l(k)$.

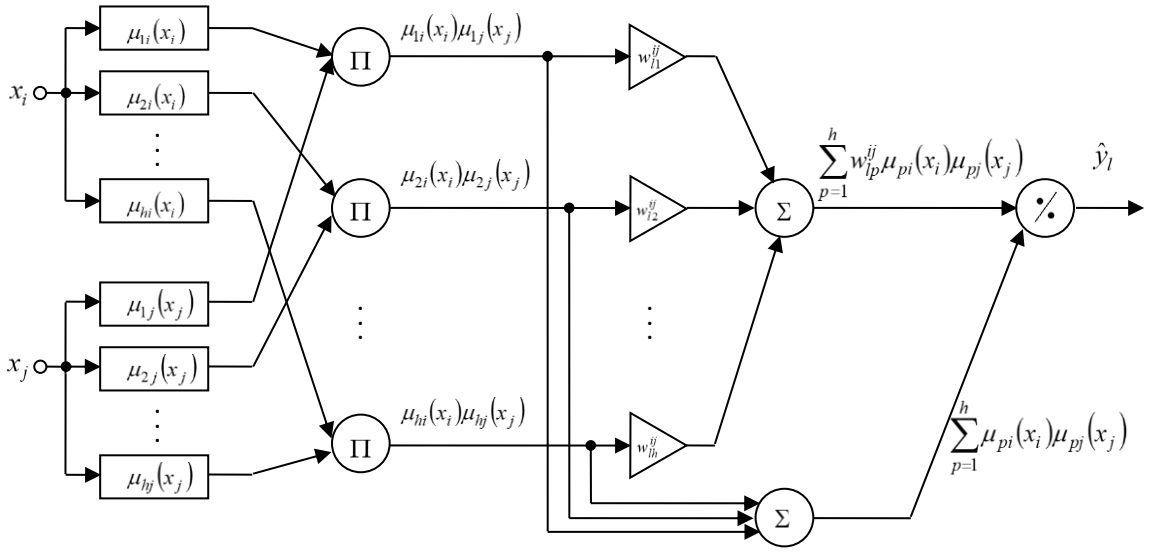


Рисунок 1.29 – Архітектура вузла МГУА-нейро-фаззі системи

До входу вузла подається двовимірний вектор вхідних сигналів $x(k)=\left(x_i(k), x_j(k)\right)^{\mathrm{T}}$, що будуть опрацьовуватись, де $k=1,2,...,N$ – номер спостереження в навчальній вибірці або поточний дискретний час.

Перший шар вузла містить $2h$ функцій належності $\mu_{pi}\left(x_i(k)\right)$, $\mu_{pj}\left(x_j(k)\right)$, $p=1,2,...,h$ і реалізує фаззіфікацію вхідних змінних.

Другий шар забезпечує агрегування рівнів належності, що розраховуються в першому шарі, містить h блоків множення і формує двовимірні радіально-базисні функції активації $\mu_{pi}\left(x_i(k)\right)\mu_{pj}\left(x_j(k)\right)$.

Третій шар – це шар синаптичних ваг, що будуть налаштовуватись в процесі навчання, при цьому виходами шару є значення $w_{lp}^{ij}\mu_{pi}\left(x_i(k)\right)\mu_{pj}\left(x_j(k)\right)$, а кількість ваг визначається кількістю функцій належності на кожному вході h.

Четвертий шар утворений двома суматорами і обчислює суми вихідних сигналів другого і третього шарів.

І, нарешті, у п’ятому (вихідному) шарі нейрона здійснюється нормалізація, в результаті якої обчислюється вихідний сигнал вузла $\hat{y}_l(k)$.

На виході вузла (п’ятого шару) формується сигнал

$$\hat{y}_l(k)=\frac{\sum\limits_{p=1}^{h} w_{lp}^{ij}\mu_{pi}\left(x_i(k)\right)\mu_{pj}\left(x_j(k)\right)}{\sum\limits_{p=1}^{h}\mu_{pi}\left(x_i(k)\right)\mu_{pj}\left(x_j(k)\right)}=\frac{\sum\limits_{p=1}^{h} w_{lp}^{ij}\tilde{x}_p(k)}{\sum\limits_{p=1}^{h}\tilde{x}_p(k)}=$$
$$=\sum_{p=1}^{h} w_{lp}^{ij}\frac{\tilde{x}_p(k)}{\sum\limits_{p=1}^{h}\tilde{x}_p(k)}=\sum_{p=1}^{h} w_{lp}^{ij}\varphi_p^{ij}\left(x(k)\right)=\left(w_l^{ij}\right)^{\mathrm{T}}\varphi^{ij}\left(x(k)\right), \tag{1.152}$$

де $\varphi^{ij}\left(x(k)\right)=\left(\varphi_1^{ij}\left(x(k)\right),...,\varphi_p^{ij}\left(x(k)\right),...,\varphi_p^{ij}\left(x(k)\right)\right)^{\mathrm{T}}$;

$$\varphi_p^{ij}\left(x(k)\right)=\mu_{pi}\left(x_i(k)\right)\mu_{pj}\left(x_j(k)\right)\left(\sum_{p=1}^{h}\mu_{pi}\left(x_i(k)\right)\mu_{pj}\left(x_j(k)\right)\right)^{-1};$$

$w_l^{ij}=\left(w_{l1}^{ij},...,w_{lp}^{ij},...,w_{lh}^{ij}\right)^{\mathrm{T}}$.

Неважко помітити, що вузол реалізує нелінійне відображення вхідних сигналів у вихідний подібно до нормалізованої радіально-базисної нейронної мережі, однак містить суттєво меншу кількість h параметрів, що налаштовуються, у порівнянні з нейронною мережею.

Якщо, користуючись введеними позначеннями, записати перетворення, що реалізуються у кожному вузлі стандартного МГУА, у вигляді

$$\hat{y}_l(k)=w_{l0}^{ij}+w_{l1}^{ij}x_i(k)+w_{l2}^{ij}x_j(k), \tag{1.153}$$

що містить три невідомих параметра, неважко помітити, що при трьох функціях належності на кожному вході запропонованого вузла, ми приходимо до тих самих трьох синаптичних ваг, що підлягають уточненню-навчанню.

Оцінювання цих синаптичних ваг може бути реалізовано у найпростішому випадку за допомогою звичайного методу найменших квадратів (МНК), що традиційно використовується у МГУА. При цьому, якщо навчальна вибірка задана у повному об’ємі, то можна використовувати пакетну форму МНК у вигляді

$$w_l^{ij}(N)=\left(\sum_{k=1}^{N}\varphi^{ij}\left(x(k)\right)\left(\varphi^{ij}\left(x(k)\right)\right)^{\mathrm{T}}\right)^{+}\sum_{k=1}^{N}\varphi^{ij}\left(x(k)\right)y(k) \qquad (1.154)$$

(тут $y(k)$ – зовнішній навчальний сигнал), якщо ж дані надходять на опрацювання послідовно в online режимі, то використовується рекурентна форма МНК

$$\begin{cases} w_l^{ij}(k)=w_l^{ij}(k-1)+\dfrac{\mathrm{P}^{ij}(k-1)\left(y(k)-\left(w_l^{ij}(k-1)\right)^{\mathrm{T}}\varphi^{ij}\left(x(k)\right)\right)\varphi^{ij}\left(x(k)\right)}{1+\left(\varphi^{ij}\left(x(k)\right)\right)^{\mathrm{T}}\mathrm{P}^{ij}(k-1)\varphi^{ij}\left(x(k)\right)}, \\ \mathrm{P}^{ij}(k)=\mathrm{P}^{ij}(k-1)-\dfrac{\mathrm{P}^{ij}(k-1)\varphi^{ij}\left(x(k)\right)\left(\varphi^{ij}\left(x(k)\right)\right)^{\mathrm{T}}\mathrm{P}^{ij}(k-1)}{1+\left(\varphi^{ij}\left(x(k)\right)\right)^{\mathrm{T}}\mathrm{P}^{ij}(k-1)\varphi^{ij}\left(x(k)\right)}. \end{cases} \qquad (1.155)$$

1.6 Адаптивне прогнозування нестаціонарних нелінійних послідовностей на основі еволюційної нейро(нео)-фаззі-WANARX-моделі

Задача математичного прогнозування послідовностей даних (часових рядів) на сьогодні досить добре вивчена, а кількість публікацій, що присвячені цій проблемі, дуже велика. Існує велика кількість методів розв’язку цієї задачі – від найпростіших регресійних, кореляційних, спектральних, експоненціального згладжування тощо до передових інтелектуальних, що потребують часом використання достатньо складних математичних методів і високої кваліфікації користувача. Проблема різко ускладнюється, якщо ряди, що досліджуються, є нестаціонарними і нелінійними, містять тренди невідомого характеру, квазіперіодичні компоненти, стохастичні і хаотичні складові. За цих умов найкраще проявили себе нелінійні прогнозувальні моделі, що засновані на математичному апараті обчислювального інтелекту, і, перш за все, нейро-фаззі системи, завдяки високим апроксимуючим і екстраполюючим властивостям,

зданості до навчання, прозорості і інтерпретованості отриманих результатів. Тут особливо варто відмітити, так звані, NARX-моделі [111], які мають вигляд

$$\hat{y}(k) = f\left(y(k-1), y(k-2), ..., y\left(k-n_y\right), x(k-1), ..., x\left(k-n_x\right)\right), \quad (1.156)$$

де $\hat{y}(k)$ – оцінка (прогноз) послідовності, що прогнозується, у момент дискретного часу $k = 1, 2, ...$;

$f(\bullet)$ – деяке нелінійне перетворення, що реалізується нейро-фаззі системою;

$x(k)$ – екзогенний фактор, що спостерігається, який визначає поведінку $y(k)$

.

Можна помітити, що в рамки опису (1.156) вкладаються як популярні AR-, ARX-, ARMAX-моделі Бокса–Дженкінса, так і нелінійні NARMA-моделі. Такі моделі добре вивчені, існує достатньо велика кількість архітектур і алгоритмів навчання, що їх реалізують, однак при цьому передбачається, що порядки моделі n_y, n_x певним чином задані заздалегідь.

У випадку структурної нестаціонарності досліджуваних рядів ці порядки апріорі невідомі і також повинні налаштовуватись у процесі навчання. У цій ситуації доцільно скористатись апаратом еволюційних конекціоністських систем, в яких передбачено не тільки налаштування синаптичних ваг і функцій активації-належності, але й власне архітектури.

На сьогодні відома ціла низка таких алгоритмів, що дозволяють реалізувати таке навчання як в пакетному, так і послідовному режимі. Ситуація суттєво ускладнюється, якщо інформація на опрацювання надходить з достатньо високою частотою у формі потоку даних.

У цьому випадку найбільш популярні еволюційні системи виявляються занадто громіздкими, щоб навчатися і опрацьовувати інформацію в online режимі.

1.6.1 Архітектура ANARX-моделі

В якості такої альтернативної достатньо простої і ефективної архітектури може бути розглянута, так звана ANARX-модель (Additive NARX), яка має вигляд [112], [113]

$$\begin{aligned}\hat{y}(k) = f_1\left(y(k-1), x(k-1)\right) + f_2\left(y(k-2), x(k-2)\right) + \ldots + \\ + f_n\left(y(k-n), x(k-n)\right) = \sum_{l=1}^{n} f_l\left(y(k-l), x(k-l)\right)\end{aligned} \tag{1.157}$$

(тут $n = \max\left\{n_y, n_x\right\}$), при цьому вихідна задача синтезу прогнозувальної системи декомпозується на множину локальних задач параметричної ідентифікації моделей-вузлів з двома входами $y(k-l)$, $x(k-l)$, $l = 1, 2, \ldots, n, \ldots$.

В якості таких вузлів автори використовували елементарні персептрони Розенблатта із сигмоїдальними функціями активації. ANARX-модель забезпечила високу якість прогнозування, але в загальному випадку зазвичай потребує включення до архітектури великої кількості вузлів.

Далі розглядаються деякі питання синтезу прогнозувальних нейро-фаззі- і нео-фаззі систем на основі ANARX-моделей, що не мають відміченних вище недоліків.

На рис. 1.30 наведена архітектура ANARX-системи, що утворена двома лініями елементів чистого запізнення z^{-1} ($z^{-1}y(k) = y(k-1)$) і n паралельно ввімкненими вузлами $N^{[l]}$, при цьому, як видно, навчання цих вузлів здійснюється незалежно один від одного, а додавання нових вузлів або виключення надлишкових жодним чином не впливає на роботу всіх інших нейронів, тобто еволюція такої системи реалізується за допомогою елементарної маніпуляції кількістю вузлів.

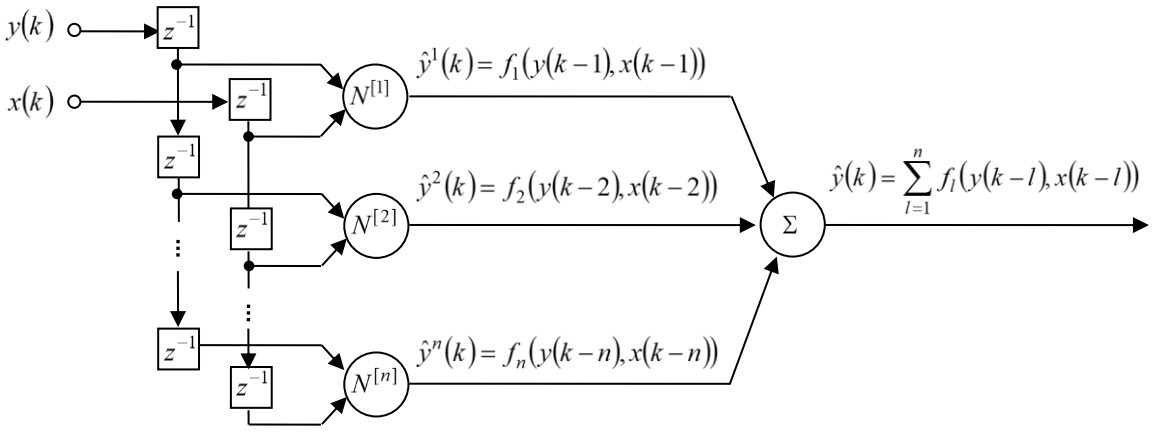


Рисунок 1.30 – Архітектура ANARX-системи

В якості вузла ANARX-системи, що розглядається, можуть використовуватись двовходові нейро-фаззі системи, що розглядались раніше, а також двовходові нео-фаззі вузли.

1.6.2 Нео-фаззі-ANARX-модель

У випадку необхідності опрацювання дуже великих об’ємів інформації в рамках концепції “Big Data”, коли на перший план виходять швидкість опрацювання даних і простота чисельної реалізації системи обчислювального інтелекту, замість нейро-фаззі-вузлів ANARX-моделі доцільно використовувати нео-фаззі-нейрони, які є нелінійними системами, що навчаються. Архітектура нео-фаззі-нейрона як вузла ANARX-системи наведена на рис. 1.31. До переваг нео-фаззі-нейрона можна віднести високу швидкість навчання, обчислювальну простоту, добрі апроксимуючі властивості, можливість знаходження глобального мінімуму критерію навчання в online режимі.

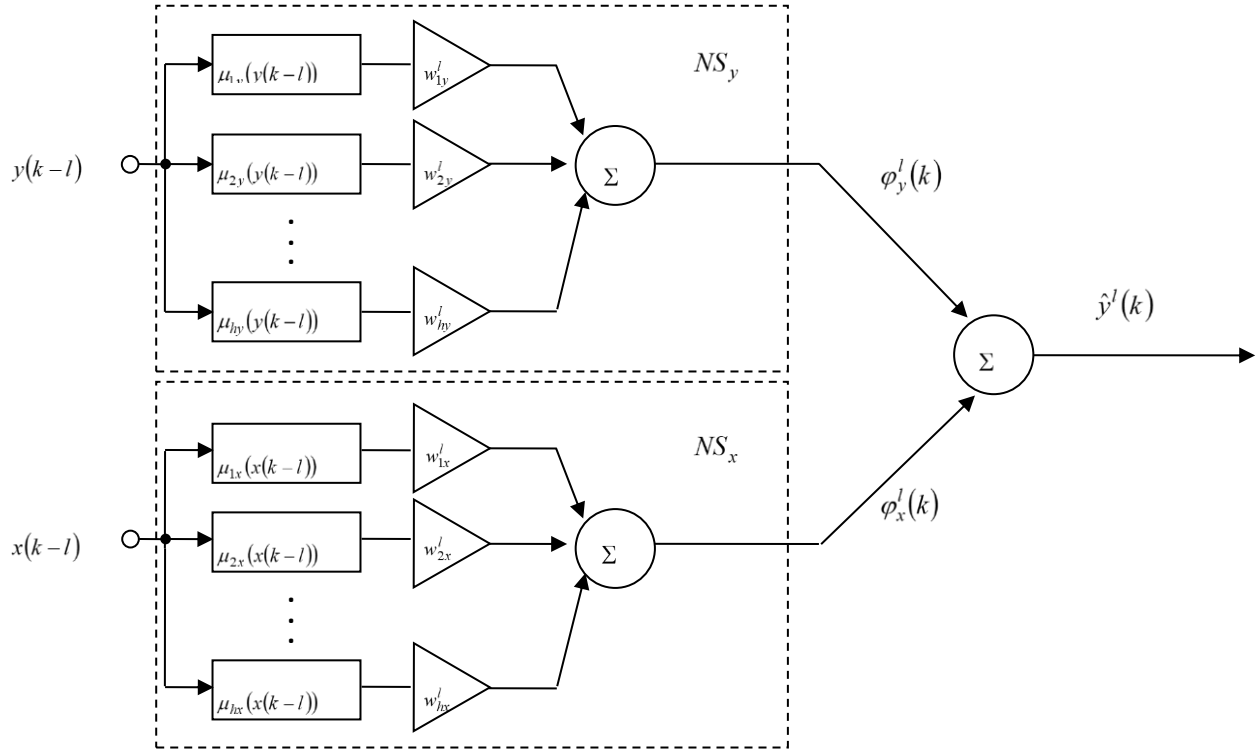


Рисунок 1.31 – Нео-фаззі вузол ANARX-системи

Складовими елементами нео-фаззі-нейрона є нелінійні синапси NS_y, NS_x, в яких реалізуються правила нечіткого висновування Такагі – Сугено нульового порядку, однак, як неважко помітити, конструктивно нео-фаззі-нейрон значно простіший за нейро-фаззі вузол, наведений на рис. 1.31.

При поданні на вхід такого вузла сигналів $y(k-l)$, $x(k-l)$, на його виході формується вихідне значення

$$\hat{y}^l(k) = \varphi^l_y(k) + \varphi^l_x(k) = \sum_{i=1}^{h} w^l_{iy}\mu_{iy}\left(y(k-l)\right) + \sum_{i=1}^{h} w^l_{ix}\mu_{ix}\left(x(k-l)\right), \qquad (1.158)$$

а на виході ANARX-моделі в цілому

$$\hat{y}(k) = \sum_{l=1}^{n}\left(\sum_{i=1}^{h} w^l_{iy}\mu_{iy}\left(y(k-l)\right) + \sum_{i=1}^{h} w^l_{ix}\mu_{ix}\left(x(k-l)\right)\right), \qquad (1.159)$$

тобто, оскільки нео-фаззі-нейрон також є адитивною моделлю, то можна сказати, що в такому випадку ANARX-модель на нео-фаззі-нейронах є двічі адитивною.

В якості функцій належності в нео-фаззі-нейроні зазвичай використовуються трикутні конструкції, які відповідають умовам одиничного розбиття

$$\sum_{i=1}^{h}\mu_{iy}\left(y\left(k-l\right)\right)=1;\ \sum_{i=1}^{h}\mu_{ix}\left(x\left(k-l\right)\right)=1, \tag{1.160}$$

що дозволяє спростити конструкцію вузла, виключивши з неї шар нормалізації.

Також в якості функцій належності нео-фаззі-нейрона можна використовувати B-сплайни, що забезпечують більш високу якість апроксимації і що також відповідають умовам одиничного розбиття. При цьому для B -сплайну q-го порядку можна записати

$$\mu_{iy}^{q}\left(y\left(k-l\right)\right)=\begin{cases}\left.\begin{array}{l}1,\text{ якщо } c_{iy}\le y\left(k-l\right)<c_{i+1,y},\\ 0\text{ в іншому випадку}\end{array}\right\}\text{для } q=1,\\ \dfrac{y\left(k-l\right)-c_{iy}}{c_{i+q-1,y}-c_{iy}}\mu_{iy}^{q-1}\left(y\left(k-l\right)\right)+\dfrac{c_{i+q,y}-y\left(k-l\right)}{c_{i+q,y}-c_{i+1,y}}\times\\ \times\mu_{i+1,y}^{q-1}\left(y\left(k-l\right)\right),\text{ для } q>1,\\ \\ i=1,\ldots,h-q,\end{cases} \tag{1.161}$$

$$\mu_{ix}^{q}\left(x\left(k-l\right)\right)=\begin{cases}\left.\begin{array}{l}1,\text{ якщо } c_{ix}\le x\left(k-l\right)<c_{i+1,x},\\ 0\text{ в іншому випадку}\end{array}\right\}\text{для } q=1,\\ \dfrac{x\left(k-l\right)-c_{ix}}{c_{i+q-1,x}-c_{ix}}\mu_{ix}^{q-1}\left(x\left(k-l\right)\right)+\dfrac{c_{i+q,x}-x\left(k-l\right)}{c_{i+q,x}-c_{i+1,x}}\times\\ \times\mu_{i+1,x}^{q-1}\left(x\left(k-l\right)\right),\text{ для } q>1,\\ i=1,...,h-q.\end{cases}\tag{1.162}$$

Варто відмітити, що B-сплайни є свого роду узагальненими функціями належності: так, при $q=2$ отримуємо традиційні трикутні функції належності, при $q=4$ – кубічні сплайни тощо.

Вводячи далі векторні змінні $w^{l}=\left(w_{1y}^{l},...,w_{hy}^{l},w_{1x}^{l},...,w_{hx}^{l}\right)^{\mathrm{T}}$, $\varphi^{l}\left(k\right)=\left(\mu_{1y}\left(y\left(k-l\right)\right),...,\mu_{hy}\left(y\left(k-l\right)\right),\mu_{1x}\left(x\left(k-l\right)\right),...,\mu_{hx}\left(x\left(k-l\right)\right)\right)^{\mathrm{T}}$, можна переписати (1.159) у вигляді

$$\hat{y}^{l}\left(k\right)=w^{l\mathrm{T}}\varphi^{l}\left(k\right)\tag{1.163}$$

і використовувати для навчання нео-фаззі-нейрона процедуру

$$\begin{cases}w^{l}\left(k\right)=w^{l}\left(k-1\right)+r_{l}^{-1}\left(k\right)\left(y\left(k\right)-w^{l\mathrm{T}}\left(k-1\right)\varphi^{l}\left(k\right)\right)\varphi^{l}\left(k\right)\\ r_{l}\left(k\right)=\alpha r_{l}\left(k-1\right)+\varphi^{l\mathrm{T}}\left(k\right)\varphi^{l}\left(k\right),\ 0\le\alpha\le1,\end{cases}\tag{1.164}$$

яка має як фільтрувальні, так і слідкувальні властивості. Можна відмітити також, що при $\alpha=1$ (1.164) повністю співпадає з оптимальним алгоритмом Качмажа–Уідроу–Хоффа.

Еволюційні системи на нео-фаззі-нейронах продемонстрували свою ефективність при розв’язку низки задач, включаючи і прогнозування, однак з точки зору простоти реалізації і швидкодії адитивна система найкращим чином пристосована для опрацювання потоків даних в задачах Data Stream Mining [114].

1.6.3 Зважена ANARX-модель (WANARX-модель)

Оскільки кожний вузол $N^{[l]}$ ANARX-моделі налаштовується незалежно від інших і представляє собою по суті окрему нейро(нео)-фаззі систему, для покращення якості прогнозів можна скористатися ідеєю комбінування ансамблю нейронних мереж. Цей підхід природньо веде до архітектури зваженої (weighted ANARX) нейро-нео-фаззі-WANARX системи, наведеної на рис. 1.32.

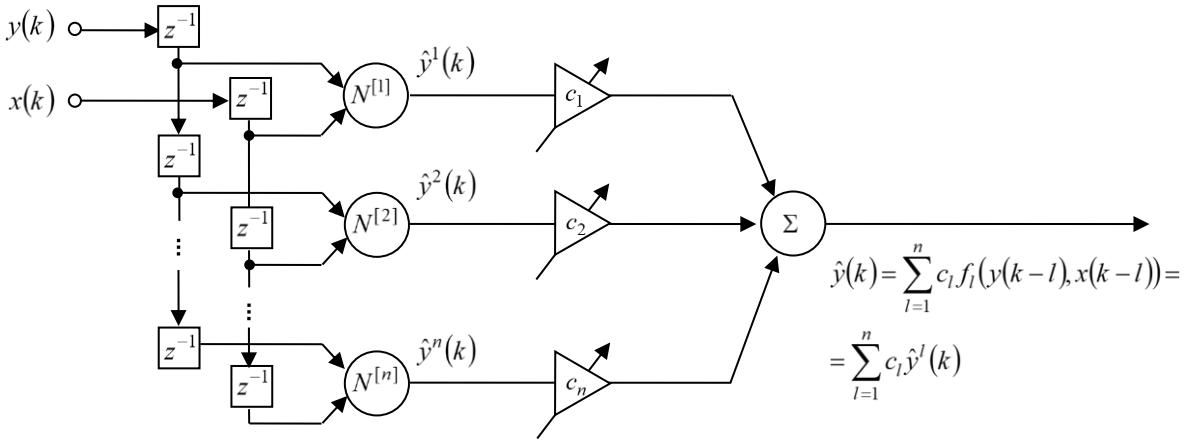


Рисунок 1.32 – Архітектура WANARX-системи

При цьому вихідний сигнал такої системи може бути записаний у вигляді

$$\hat{y}(k)=\sum_{l=1}^{n}c_l\hat{y}^l(k)=c^{\mathrm{T}}\hat{\bar{y}}(k), \tag{1.165}$$

де $\hat{\bar{y}}(k)=\left(\hat{y}^1(k),\hat{y}^2(k),...,\hat{y}^n(k)\right)^{\mathrm{T}}$;

$c=\left(c_1, c_2, ..., c_n\right)^{\mathrm{T}}$ – вектор вагових коефіцієнтів, що налаштовуються і що визначають близькість сигналів $\hat{y}^l\left(k\right)$ до процесу $y\left(k\right)$, що прогнозується, та які відповідають умовам незміщенності

$$\sum_{l=1}^{n} c_l = c^{\mathrm{T}} I_n = 1, \tag{1.166}$$

де I_n – $\left(n\times 1\right)$-вектор, утворений одиницями.

Для знаходження вектора c в пакетному режимі можна скористатися методом невизначених множників Лагранжа. Для цього у розгляд вводиться послідовність похибок прогнозування

$$\begin{aligned} v\left(k\right) &= y\left(k\right) - \hat{y}\left(k\right) = y\left(k\right) - c^{\mathrm{T}}\hat{\bar{y}}\left(k\right) = \\ &= c^{\mathrm{T}} I_n y\left(k\right) - c^{\mathrm{T}}\hat{\bar{y}}\left(k\right) = c^{\mathrm{T}}\left(I_n y\left(k\right) - \hat{\bar{y}}\left(k\right)\right) = c^{\mathrm{T}} V\left(k\right), \end{aligned} \tag{1.167}$$

функція Лагранжа

$$L\left(c,\lambda\right) = \sum_{k} c^{\mathrm{T}} V\left(k\right) V^{\mathrm{T}}\left(k\right) c + \lambda\left(c^{\mathrm{T}} I_n - 1\right) = c^{\mathrm{T}} R c + \lambda\left(c^{\mathrm{T}} I_n - 1\right), \tag{1.168}$$

де λ – невизначений множник Лагранжа;

$R=\sum_{k} V\left(k\right) V^{\mathrm{T}}\left(k\right)$ – кореляційна матриця похибок.

Також у розгляд вводиться система рівнянь Каруша–Куна–Таккера

$$\begin{cases} \nabla_c L\left(c,\lambda\right) = 2Rc + \lambda I_n = \vec{0}, \\ \dfrac{\partial L}{\partial \lambda} = c^{\mathrm{T}} I_n - 1 = 0. \end{cases} \tag{1.169}$$

Розв’язок системи (1.169) веде до відомого результату

$$\begin{cases} c = R^{-1} I_n \left(I_n^{\mathrm{T}} R^{-1} I_n \right)^{-1}, \\ \lambda = -2 I_n^{\mathrm{T}} R^{-1} I_n, \end{cases} \tag{1.170}$$

при цьому лагранжіан (1.168) у сідловій точці набуває значення

$$L^*(c, \lambda) = \left(I_n^{\mathrm{T}} R^{-1} I_n \right)^{-1}. \tag{1.171}$$

Реалізація алгоритму (1.170) може натрапляти на суттєві перешкоди при опрацюванні інформації в online режимі і високому рівні кореляції сигналів $\hat{y}^l(k)$, який призводить до поганої обумовленості матриці R, яку необхідно обертати на кожному такті реального часу k.

Запишемо функцію Лагранжа (1.168) у вигляді

$$L(c, \lambda) = \sum_k \left(y(k) - c^{\mathrm{T}} \hat{\vec{y}}(k) \right)^2 + \lambda \left(c^{\mathrm{T}} I_n - 1 \right) \tag{1.172}$$

і градієнтний алгоритм пошуку її сідлової точки на основі процедури Ерроу–Гурвіца:

$$\begin{cases} c(k) = c(k-1) - \eta_c(k) \nabla_c L(c, \lambda), \\ \lambda(k) = \lambda(k-1) + \eta_\lambda(k) \dfrac{\partial L(c, \lambda)}{\partial \lambda}, \end{cases} \tag{1.173}$$

або

$$\begin{cases} c(k)=c(k-1)+\eta_c(k)\Big(2\Big(y(k)-c^{\mathrm{T}}(k-1)\hat{\bar{y}}(k)\Big)\hat{\bar{y}}(k)-\lambda(k-1)I_n\Big)= \\ \qquad =c(k-1)+\eta_c(k)\Big(2v(k)\hat{\bar{y}}(k)-\lambda(k-1)I_n\Big) \\ \lambda(k)=\lambda(k-1)+\eta_\lambda(k)\Big(c^{\mathrm{T}}(k)I_n-1\Big), \end{cases} \tag{1.174}$$

де $\eta_c(k)$, $\eta_\lambda(k)$ – параметри кроку навчання.

Процедура Ерроу–Гурвіца збігається до сідлової точки за достатньо загальних припущеннях відносно значень $\eta_c(k)$, $\eta_\lambda(k)$, однак для прискорення процесу навчання, що особливо важливо в задачах Data Stream Mining, ці параметри можуть бути оптимізовані.

Для цього домножимо перше співвідношення (1.174) зліва на $\hat{\bar{y}}^{\mathrm{T}}(k)$

$$\hat{\bar{y}}^{\mathrm{T}}(k)c(k)=\hat{\bar{y}}^{\mathrm{T}}c(k-1)+\eta_c(k)\Big(2v(k)\big\|\hat{\bar{y}}(k)\big\|^2-\lambda(k-1)\hat{\bar{y}}^{\mathrm{T}}I_n\Big) \tag{1.175}$$

і введемо додаткову функцію, що характеризує критеріальну збіжність:

$$\begin{aligned}\Big(y(k)-\hat{\bar{y}}^{\mathrm{T}}(k)c(k)\Big)^2 &= v^2(k)-2\eta_c(k)v(k)\Big(2v(k)\big\|\hat{\bar{y}}(k)\big\|^2-\lambda(k-1)\hat{\bar{y}}^{\mathrm{T}}I_n\Big)+ \\ &+\eta_c^2(k)\Big(2v(k)\big\|\hat{\bar{y}}(k)\big\|^2-\lambda(k-1)\hat{\bar{y}}^{\mathrm{T}}I_n\Big)^2.\end{aligned} \tag{1.176}$$

Розв’язок диференційного рівняння

$$\begin{aligned}\frac{\partial\Big(y(k)-\hat{\bar{y}}^{\mathrm{T}}(k)c(k)\Big)^2}{\partial\eta_c(k)} &= -2v(k)\Big(2v(k)\big\|\hat{\bar{y}}(k)\big\|^2-\lambda(k-1)\hat{\bar{y}}^{\mathrm{T}}I_n\Big)+ \\ &+2\eta_c(k)\Big(2v(k)\big\|\hat{\bar{y}}(k)\big\|^2-\lambda(k-1)\hat{\bar{y}}^{\mathrm{T}}I_n\Big)^2=0\end{aligned} \tag{1.177}$$

дозволяє отримати оптимальне значення кроку навчання $\eta_c(k)$ у формі

$$\eta_c(k)=\frac{v(k)}{2v(k)\left\|\hat{\tilde{y}}(k)\right\|^2-\lambda(k-1)\hat{\tilde{y}}^{\mathrm{T}}I_n}, \tag{1.178}$$

підставляючи яку в (1.174), остаточно можна записати

$$\begin{cases} c(k)=c(k-1)+\dfrac{v(k)\left(2v(k)\hat{\tilde{y}}(k)-\lambda(k-1)I_n\right)}{2v(k)\left\|\hat{\tilde{y}}(k)\right\|^2-\lambda(k-1)\hat{\tilde{y}}^{\mathrm{T}}I_n}, \\ \lambda(k)=\lambda(k-1)+\eta_\lambda(k)\left(c^{\mathrm{T}}(k)I_n-1\right). \end{cases} \tag{1.179}$$

Неважко помітити, що при $\lambda(k-1)=0$ процедура (1.179) співпадає з алгоритмом Качмажа–Уідроу–Хоффа.

2 РОЗРОБКА МЕТОДІВ ОБРОБКИ НЕЧІТКОЇ ІНФОРМАЦІЇ, ЩО НАДХОДИТЬ У ФОРМІ ПОТОКІВ ДАНИХ

2.1 Багатошарові нечіткі кластерувальні системи для аналізу потоків даних

Завдання кластеризації великих масивів багатовимірних спостережень (векторів-образів) часто зустрічається в багатьох реальних практичних завданнях, а для її вирішення розроблено безліч методів, при цьому в останні роки в рамках концепції Big Data особлива увага приділяється обробці інформації, що зберігається або в надвеликих базах даних (VLDB), або надходить на обробку в online режимі в формі потоку даних (Data Stream). Для вирішення цих завдань з успіхом може бути використаний математичний апарат обчислювального інтелекту (Computational Intelligence) і, перш за все, штучні нейронні мережі і м'які обчислення (Soft Computing), засновані на нечіткій логіці.

У найбільш загальній постановці завдання кластеризації передбачається, що є масив з N багатовимірних спостережень, що описуються $n-$вимірними векторами ознак $x(k)=(x_1(k),x_2(k),...,x_n(k))^T \in R^n$, $k=1,2,...,N,...,$ який необхідно розбити на m кластерів, при цьому це число може бути заздалегідь не відомо, тобто $1<m<N$. Очевидно, що така велика кількість відомих методів рішення задачі кластеризації пов'язано з тим, що сьогодні не існує універсального алгоритму придатного для ефективного використання у всіх виникаючих реальних ситуаціях. Одна з таких можливих і досить складних ситуацій пов'язана з припущенням, що кожен вектор спостережень може одночасно ставитися з різними рівнями можливостей, ймовірностей або приладдя не до одного, а відразу до кількох чи до всіх формується кластерам. У цій ситуації доцільно використання, так званих, м'яких алгоритмів (soft algorithms) [115], серед яких найбільшу увагу отримали нечіткі методи кластеризації [116]–[118] і імовірнісні алгоритми (probabilistic model-based

algorithms) [119], серед яких для обробки великих масивів широке поширення отримав, так званий, ЕМ-алгоритм (Expectation-Maximization algorithm) [120]–[124], в основі якого лежать суто імовірнісні передумови. Кожен із зазначених підходів має свої достоїнства і недоліки, в зв’язку з чим представляється доцільною розробка чисельно простою м’якою процедури кластеризації, що об'єднує в собі переваги ймовірного і фаззи-підходів і призначеної для обробки потоків даних, що надходять в on-line режимі.

2.1.1 М’який імовірнісний нечіткий ЕМ-метод кластеризації багатовимірних даних

Завдання ймовірнісної кластеризації в загальній постановці зводиться до проблеми самонавчання при невідомому числі областей, при цьому передбачається, що щільність розподілу даних в кожному кластері підпорядковується багатовимірному нормальному (гаусівському) закону

$$p_j(x) = \frac{1}{(2\pi)^{\frac{n}{2}}\sqrt{\det \Sigma_j}} \exp(-\frac{1}{2}(x-c_j)^T \Sigma_j^{-1}(x-c_j)),\ j = 1,2,...m, \tag{2.1}$$

а спільна щільність розподілу всіх даних описується сумішшю

$$p(x) = \sum_{j=1}^{m} p_j p_j(x) = \sum_{j=1}^{m} \frac{p_j}{(2\pi)^{\frac{n}{2}}\sqrt{\det \Sigma_j}} \exp(-\frac{1}{2}(x-c_j)^T \Sigma_j^{=1}\left(x-c_j\right)),\ j = 1,2,...m, \tag{2.2}$$

де $c_j - (n \times 1) -$ мірний вектор-центр ваги $j -$ го кластера;

$\sum_j - (n \times n) -$ кореляційна матриця $j -$ го кластера така, що

$$\Sigma_j = \frac{1}{N}\sum_{k=1}^{N}(x(k)-c_j)(x(k)-c_j)^T; \tag{2.3}$$

p_j – апріорні ймовірності (ваги), що підкоряються природній умові

$$\sum_{j=1}^{m} p_j = 1, \tag{2.4}$$

при цьому передбачається, що c_j, Σ_j і $p_j \forall j = 1,2,...,m$ апріорі невідомі і підлягають оцінюванню в процесі кластеризації.

Тут можна відзначити, що показник ступеня експоненти в (2.1), (2.2) є відстань Махаланобіса між c_j і наглядом $x(k)$, Тобто

$$d_M^2(c_j, x(k)) = (x(k)-c_j)^T \Sigma_j^{-1}(x(k)-c_j), \tag{2.5}$$

а умова (2.4) збігається з умовою, що накладається на суму приладдя в популярному алгоритмі нечітких С-середніх (FCM) Дж. Бездека

$$\sum_{j=1}^{m} u_j(k) = 1, \tag{2.6}$$

де $u_j(k)$ – рівень нечіткої належності спостереження $x(k)$ j – му кластеру.

У зв'язку з цим методи кластеризації, пов'язані з обмеженням (2.6), називаються імовірнісними алгоритмами нечіткої кластеризації.

Робота ЕМ-алгоритму складається з повторюваної послідовності двох кроків, при цьому на кроці Е (expectation step) проводиться оцінювання параметрів спільного розподілу (2.2), а на кроці М (maximization step) максимізує критерій самонавчання у вигляді логарифмічною функції правдоподібності

$$E(c_j, \Sigma_j, p_j, x(k)) = \sum_{k=1}^{N} \log(\sum_{j=1}^{m} p_j p_j(x(k-1)),$$

для чого можуть бути використані як традиційні градієнтні, так і квазіньютонівські процедури оптимізації.

І, нарешті, для оцінки ймовірності належності спостереження $x(k)$ $j-$му кластеру використовується вираз

$$p_j(x(k)) = \frac{p_j \exp\left(-\frac{1}{2}\left(x(k)-c_j\right)^T \Sigma_j^{-1}\left(x(k)-c_j\right)\right)}{\sum_{l=1}^{m} p_l \exp\left(-\frac{1}{2}\left(x(k)-c_j\right)^T \Sigma_j^{-1}\left(x(k)-c_l\right)\right)}, \qquad (2.7)$$

який задовольняє умові (2.4).

Окремим випадком ЕМ-методу є відомий метод кластеризації K-середніх і збігається з ним при $p_j = m^{-1}$ і одиничних кореляційних матрицях Σ_j. При цьому метод K-середніх є чіткою процедурою, що означає, що кожне спостереження може бути віднесено до єдиного кластеру. При цьому метод K-середніх істотно простіше з обчислювальної точки зору, ніж ЕМ-метод, і пов'язаний з мінімізацією критерію самонавчання, заснованого на евклідових відстанях

$$E(c_j, x(k)) = \sum_{k=1}^{N} \sum_{j=1}^{m} u_j(k) \left\| x(k) - c_j \right\|^2, \qquad (2.8)$$

де

$$u_j(k) = \begin{cases} 1, \textit{ якщо } x(k) \textit{ належить } j-\textit{му кластеру}, \\ 0 \textit{ у протилежному випадку} \end{cases} \qquad (2.9)$$

ЕМ-метод також відноситься до процедур, заснованим на відстанях, і в цьому сенсі близький до, так званого, методу K-середніх Махаланобіса, що є чіткою процедурою, що мінімізує цільову функцію

$$E(c_j,\Sigma_j,x(k))=\sum_{k=1}^{N}\sum_{j=1}^{m}u_j(k)(x(k)-c_j)^T\Sigma_j^{-1}\left(x\left(k\right)-c_j\right), \qquad (2.10)$$

де Σ_j і $u_j\left(k\right)$ визначається виразами (2.3), (2.9).

Принципова різниця між ЕМ-методом і методом K-середніх Махаланобіса полягає в тому, що останній дає однозначне рішення про належність кожного спостереження єдиному кластеру, в той час як імовірнісна процедура враховує можливість перекриття класів з розрахунком ймовірностей відповідно до виразу (2.7).

Поряд з незаперечними перевагами ЕМ-метод має і ряд істотних обмежень. По-перше, вихідні дані повинні мати випадкову природу і підкорятися нормальному закону розподілу. По-друге, на М-етапі можливо «застрявання» процесу оптимізації в локальних екстремуму (ця проблема може бути подолана за допомогою використання процедур багатоекстремальної оптимізації). По-третє, з обчислювальної точки зору це все-таки досить громіздка процедура. І, нарешті, мається на увазі, що весь масив даних, що підлягають кластеризації, заданий заздалегідь і не змінюється в процесі обробки.

У зв’язку з цим представляється доцільною розробка чисельно простого алгоритму кластеризації, заснованого на метриці Махаланобіса, що враховує можливість взаємного перекриття формованих кластерів і дозволяє аналізувати потік даних, що послідовно надходять на обробку в online режимі.

2.1.2 Нечітка імовірнісна кластеризація на основі WTA-правила самонавчання

Результат мінімізації критерію самонавчання (2.8) методу K-середніх має вигляд середнього арифметичного даних кожного кластера

$$c_j = \frac{\sum_{k=1}^{N} u_j(k)x(k)}{\sum_{k=1}^{N} u_j(k)} = \frac{1}{N_j}\sum_{x(k)\in Cl_j} x(k), \qquad (2.11)$$

де N_j – кількість спостережень $x(k)$, віднесених до кластеру Cl_j.

Якщо ж дані надходять на обробку послідовно в online режимі, рішення (2.11) може бути отримано за допомогою кластеризуються нейронної мережі Т. Кохонена [18], настройка синаптичних ваг якої є по суті компонентами векторів-центроїдів та проводиться за допомогою тих або інших алгоритмів конкурентного самонавчання, найбільш популярним з яких є WTA-правило (“Winner Takes All”). При цьому сама процедура настройки подібно EM-алгоритму складається з послідовності двох етапів: конкуренції (відповідає E-кроку) і синаптичної адаптації (відповідає M-кроку).

Суть конкурентного навчання за Кохоненом полягає в тому, що якщо до моменту надходження спостереження $x(k)$ розраховані координати центроїдів $c_1(k-1),...,c_j(k-1),...,c_m(k-1)$, серед них обирається «переможець» найближчий в сенсі евклидової відстані

$$d_E^2\left(c_j(k-1), x(k)\right) = \left\|x(k) - c_j(k-1)\right\|^2 \qquad (2.12)$$

де $x(k)$ (E-крок), який і уточнюється на M-кроці за допомогою рекурентної процедури

$$c_j(k)=\begin{cases} c_j(k-1)+\eta(k)\big(x(k)-c_j(k-1)\big), \\ \textit{якщо } c_j(k-1)-\text{"}\textit{переможець}\text{"}, \\ c_j(k-1)\ \textit{у протилежному випадку}, \end{cases} \tag{2.13}$$

де $\eta(k)$ – параметр кроку навчання, який обирається зазвичай з емпіричних міркувань, хоча нескладно показати, що результат (2.11) може бути отриманий при $\eta(k)=k^{-1}$.

Нескладно помітити, що перше співвідношення (2.13) є не що інше, як градиентная мінімізація (2.8), тобто може бути переписано у вигляді

$$c_j(k)=\begin{cases} c_j(k-1)-\eta(k)\nabla_{c_j} d_E^2\big(c_j(k-1),x(k)\big), \\ \textit{якщо } c_j(k-1)-\text{"}\textit{переможець}\text{"}, \\ c_j(k-1)\ \textit{у протилежному випадку}. \end{cases} \tag{2.14}$$

Аналогічно (2.14) може бути введена градиентная процедура мінімізації критерію (2.10) на основі метрики Махаланобіса (2.5) у вигляді

$$c_j(k)=\begin{cases} c_j(k-1)-\eta(k)\nabla_{c_j} d_M^2\big(c_j(k-1),x(k)\big), \\ \textit{якщо } c_j(k-1)-\text{"}\textit{переможець}\text{"}, \\ c_j(k-1)\ \textit{у протилежному випадку}, \end{cases}$$

або

$$c_j(k)=\begin{cases} c_j(k-1)+\eta(k)\times \\ \times\Sigma_j^{-1}(k-1)\big(x(k)-c_j(k-1)\big), \\ \textit{якщо } c_j(k-1)-\text{"}\textit{переможець}\text{"}, \\ \Sigma_j(k-1)=\dfrac{1}{k-1}\times \\ \times\sum\limits_{l=1}^{k-1}\big(x(l)-c_j(k-1)\big)\big(x(l)-c_j(k-1)\big)^T, \\ c_j(k-1)\ \textit{у протилежному випадку}. \end{cases} \tag{2.15}$$

Нескладно помітити, що алгоритм самонавчання (2.15) є по суті послідовною реалізацією методу K-середніх Махаланобіса, тобто дозволяє отримати тільки чітке рішення.

Для оцінки рівня належності окремих спостережень в разі перекриття класів замість громіздкого виразу (2.7) доцільно скористатися оцінкою, прийнятою в стандартному FCM-алгоритмі Дж. Бездека, використовуючи замість відстані $d_E^2\big(c_j(k),x(k)\big)$ метрику Махаланобіса $d_M^2\big(c_j(k),x(k)\big)$ у вигляді

$$u_j(k)=\frac{d_M^{-2}\big(c_j(k),x(k)\big)}{\sum\limits_{l=1}^{m}\big(d_M^{-2}\big(c_l(k),x(k)\big)\big)}=\frac{\Big(\big(x(k)-c_j(k)\big)\Sigma_j^{-1}\big(x(k)-c_j(k)\big)\Big)^{-1}}{\sum\limits_{l=1}^{m}\Big(\big(x(k)-c_l(k)\big)^T\Sigma_l^{-1}\big(x(k)-c_l(k)\big)\Big)^{-1}}. \tag{2.16}$$

Таким чином, процедура нечіткої ймовірнісної кластеризації (2.15), (2.16), будучи своєрідним гібридом EM-алгоритму, методу K-середніх Махаланобіса, алгоритмів нечіткої кластеризації Бездека і Гата-Геви [125] і нечіткої кластерувальної нейронної мережі Кохонена в її адаптивному варіанті [126], [127], характеризується обчислювальної простотою і дозволяє аналізувати дані, послідовно надходять на обробку в online режимі.

2.1.3 Багатошарова ядерна кластерувальна нейро-фаззі система і алгоритми її самонавчання

Як було вище описано, в традиційній постановці завдання кластеризації передбачається, що в процесі обробки даних кластери лінійно роздільні і мають опуклу форму. На сьогоднішній день в реальних ситуаціях кластери, як правило, взаємно перетинаються, а для вирішення подібних завдань був розроблений нечіткий кластерний аналіз, що є узагальненням чітких процедур кластеризації. Для подолання труднощів, пов'язаних з лінійно нероздільними кластерами довільної форми, був розроблений ядерний кластерний аналіз [128]–[133] і, відповідно, ядерний нечіткий кластерний аналіз. В основі ядерних нечітких процедур кластеризації лежать ідеї, пов’язані з радіально-базисними нейронними мережами (RBFN) і машинами опорних векторів (SVM), а також традиційні процедури нечіткої кластеризації.

У зв’язку з цим є доцільною розробка online ядерної нейро-фаззі системи і адаптивних алгоритмів її самонавчання, що дозволяють в послідовному режимі обробляти потоки даних і формувати пересічні кластери довільної форми.

2.1.4 Архітектура ядерної нечіткої кластерувальної мережі Т. Кохонена

Архітектура пропонованої кластерувальної системи містить чотири шари обробки інформації і наведена на рис. 2.1.

Вхідна інформація в формі послідовності $n-$вимірних векторів $x(k)$ надходить на перший прихований шар PL (preprocessing layer), де вихідні дані центруються щодо поточного середнього і нормуються на гіперкуб за допомогою елементарних співвідношень (для online обробки)

$$\bar{x}(k)=\bar{x}(k-1)+\frac{1}{k}\left(x(k)-\bar{x}(k-1)\right) \qquad (2.17)$$

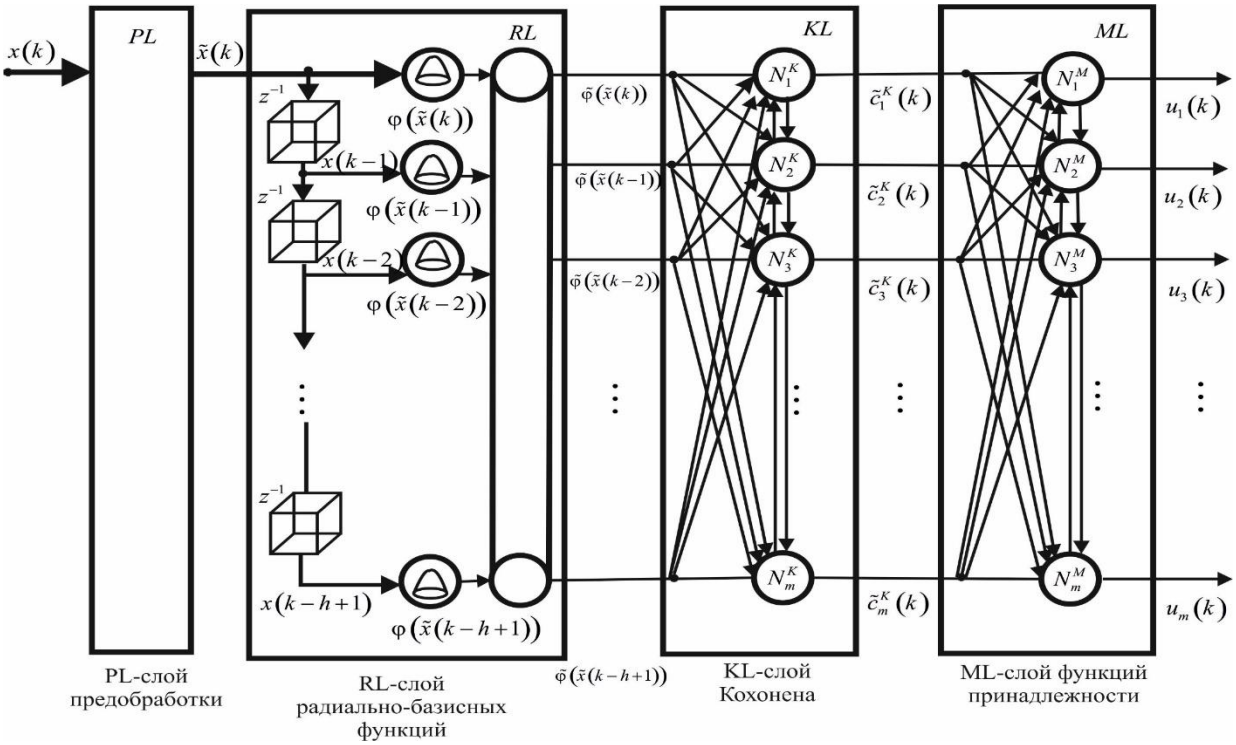


Рисунок 2.1 – Архітектура ядерної нечіткої кластерувальної мережі Т. Кохонена

або (в пакетному режимі)

$$\bar{x}(k)=\frac{1}{k}\sum_{p=1}^{k}x(p),$$

$$\bar{\bar{x}}(k)=x(k)-\bar{x}(k),$$

$$\tilde{x}_i(k)=2\frac{\bar{\bar{x}}_i(k)-\bar{\bar{x}}_{i\min}}{\bar{\bar{x}}_{i\max}-\bar{\bar{x}}_{i\min}}-1,\ \forall i=1,2,...,n, \qquad (2.18)$$

в результаті чого дані, що надходять на другий прихований шар, задовольняють умові

$$N^{-1}\sum_{k=1}^{N}\tilde{x}(k)=0,\ -1\le\tilde{x}_i(k)\le 1.$$

Послідовність векторів $\tilde{x}(k)$ подається на другий прихований шар радіально-базисних функцій RL (radial-basis-function layer), де проводиться

підвищення розмірності вхідного простору даних R^n в простір ознак більш високої розмірності R^h, $h > n$.

У загальному випадку для підвищення розмірності простору можуть бути використані стандартні Гауссіани

$$K\left(\tilde{x}(k), \tilde{x}(p)\right) = \exp\left(-\frac{\left\|\tilde{x}(k) - \tilde{x}(p)\right\|^2}{2\sigma^2}\right),$$

що застосовуються в RBFN і дозволяють забезпечити лінійну роздільність формованих кластерів відповідно до теореми Кавера [134]. При цьому для оцінки відстані між двома векторами $\tilde{x}(k)$ і $\tilde{x}(p)$ замість традиційної евклідової метрики

$$d_E^2\left(\tilde{x}(k), \tilde{x}(p)\right) = \left\|\tilde{x}(k) - \tilde{x}(p)\right\|^2, \tag{2.19}$$

що зазвичай використовується в задачах кластеризації, за допомогою «ядерного трюку» (distance kernel trick) може бути введена оцінка розходження (dissimilarity measure)

$$\begin{aligned} d_D^2\left(\tilde{x}(k), \tilde{x}(p)\right) &= K\left(\tilde{x}(k), \tilde{x}(k)\right) + K\left(\tilde{x}(p), \tilde{x}(p)\right) - \\ -2K\left(\tilde{x}(k), \tilde{x}(p)\right) &= 2\left(1 - K\left(\tilde{x}(k), \tilde{x}(p)\right)\right) \end{aligned} \tag{2.20}$$

або подібності (similarity measure)

$$d_S^2\left(\tilde{x}(k), \tilde{x}(p)\right) = K\left(\tilde{x}(k), \tilde{x}(p)\right)$$

в просторі підвищеної розмірності R^h.

Найбільш складним моментом при синтезі другого шару є вибір кількості і координат центрів радіально-базисних функцій. Найпростіше, використовуючи

ідею «нейрони в точках даних» [135], задати число радіально-базисних функцій, що дорівнює кількості спостережень N з центрами, що збігаються з координатами спостережень $\tilde{x}(k) \; \forall \; k = 1,2,...,N$ так, як це прийнято в нейронних мережах опорних векторів або узагальнених регресійних нейронних мережах, однак при обробці потоків даних, коли число спостережень стає дедалі більше, цей підхід є безперспективним.

Більш ефективним в даному випадку є використовувати ідею «ковзного вікна», коли замість всієї вибірки даних використовується тільки її частина, яка містить h останніх спостережень $\tilde{x}(k)$, $\tilde{x}(k-1)$, ..., $\tilde{x}(k-h+1)$. При цьому значення h може змінюватися в процесі обробки даних.

В результаті підвищення розмірності вхідного простору формується $(h \times 1)$ -вектор ознак на основі радіально-базисних функцій

$$\varphi(k,h) = \left(\varphi\left(\tilde{x}(k)\right), \varphi\left(\tilde{x}(k-1)\right),..., \varphi\left(\tilde{x}(k-q)\right),..., \varphi\left(\tilde{x}(k-h+1)\right)\right)^T,$$

де

$$\varphi\left(\tilde{x}(k-q)\right) = \exp\left(-\frac{\left\|\tilde{x} - \tilde{x}(k-q)\right\|^2}{2\sigma^2}\right).$$

Крім радіально-базисних функцій шар RL містить $(h-1)n$ елементів чистого запізнювання z^{-1}, які формують передісторію $\tilde{x}(k-1)$, $\tilde{x}(k-2)$, ..., $\tilde{x}(k-h+1)$, і блок нормування N, що перетворює вектор ознак так, щоб

$$\left\|\tilde{\varphi}(k,h)\right\|_2 = 1, \; \tilde{\varphi}(k,h) = \varphi(k,h)\left\|\varphi(k,h)\right\|^{-1}$$

або

$$\left\|\tilde{\varphi}(k,h)\right\|_{\infty}=1$$

за допомогою виразу (2.17).

Вектор ознак $\tilde{\varphi}(k,h)$ далі подається на третій прихований шар (KL), що представляє собою по суті шар Кохонена самоорганізовної карти, де в on-line режимі конкуренції і синаптичної адаптації проводиться настройка центроїдів-прототипів $\tilde{c}_j^k(k),\ j=1,2,...,m$ формованих кластерів.

І, нарешті, четвертий вихідний шар ML (membership layer) оцінює рівні належності кожного вектора ознак $\tilde{\varphi}(k,h)$ до кожного з m формованих кластерів.

Зауважимо також, що третій і четвертий шари є по суті адаптивною нейро-фаззі мережею Т. Кохонена, призначеною для вирішення завдань нечіткої кластеризації даних, що послідовно надходять на обробку.

2.1.5 Метод самонавчання кластерувальної нейро-фаззі системи

Процес самонавчання-настройки синаптичних ваг даної системи реалізується в третьому прихованому шарі (KL) на основі принципів конкурентного навчання і нечіткої кластеризації з використанням імовірнісного підходу Дж. Бездека.

Відповідно до цього підходу процес кластеризації реалізується в процесі мінімізації цільової функції

$$E\left(u_j(k),\tilde{c}_j^k\right)=\sum_{k=1}^{N}\sum_{j=1}^{m}u_j^{\beta}(k)\left\|\tilde{\varphi}(k,h)-\tilde{c}_j^k\right\|^2 \qquad (2.21)$$

при обмеженнях

$$\sum_{j=1}^{m} u_j(k) = 1, \tag{2.22}$$

$$0 \le \sum_{k=1}^{N} u_j(k) \le N, \tag{2.23}$$

де $u_j(k) \in [0,1]$;

$\beta -$ параметр фаззіфікації (фаззіфікатор), що визначає «розмитість» меж між кластерами.

Використовуючи далі стандартну техніку нелінійного програмування, засновану на невизначених множниках Лагранжа і вирішенні системи рівнянь Каруша–Куна–Таккера, нескладно отримати результат:

$$\begin{cases} \tilde{c}_j^k = \dfrac{\sum\limits_{k=1}^{N} u_j^{\beta}(k)\tilde{\varphi}(k,h)}{\sum\limits_{k=1}^{N} u_j^{\beta}(k)}, \\ u_j(k) = \dfrac{\left(\left\|\tilde{\varphi}(k,h) - \tilde{c}_j^k\right\|^2\right)^{\frac{1}{1-\beta}}}{\sum\limits_{l=1}^{m}\left(\left\|\tilde{\varphi}(k,h) - \tilde{c}_l^k\right\|^2\right)^{\frac{1}{1-\beta}}}, \end{cases} \tag{2.24}$$

при цьому в другому співвідношенні може бути використана як стандартна метрика (2.19), так і міра розходження (2.20).

Вважаючи в (2.24) значення фаззіфікатора $\beta = 2$, приходимо до популярного FCM-алгоритму

$$\begin{cases} \tilde{c}_j^k = \dfrac{\sum\limits_{k=1}^{N} u_j^2(k)\tilde{\varphi}(k,h)}{\sum\limits_{k=1}^{N} u_j^2(k)}, \\ u_j(k) = \dfrac{\left\|\tilde{\varphi}(k,h) - \tilde{c}_j^k\right\|^{-2}}{\sum\limits_{l=1}^{m}\left\|\tilde{\varphi}(k,h) - \tilde{c}_l^k\right\|^{-2}}. \end{cases} \quad (2.25)$$

Як можна бачити, процедури (2.24), (2.25) реалізують пакетний режим обробки інформації. Для обробки даних в online режимі можуть бути використані рекурентні алгоритми нечіткої кластеризації, в основі яких лежить процедура нелінійного програмування Ерроу–Гурвіца–Удзави:

$$\begin{cases} u_j(k+1) = \dfrac{\left(\left\|\tilde{\varphi}(k+1,h) - \tilde{c}_j^k(k)\right\|^2\right)^{\frac{2}{1-\beta}}}{\sum\limits_{l=1}^{m}\left(\left\|\tilde{\varphi}(k+1,h) - \tilde{c}_j^k(k)\right\|^2\right)^{\frac{2}{1-\beta}}}, \\ \tilde{c}_j^k(k+1) = \tilde{c}_j^k(k) + \eta(k)u_j^\beta(k+1)\left(\tilde{\varphi}(k+1,h) - \tilde{c}_j^k(k)\right), \end{cases} \quad (2.26)$$

де $\eta(k) > 0-$ параметр кроку, який визначає швидкість збіжності алгоритму (2.26).

Нескладно помітити, що співмножник $u_j^\beta(k+1)$ відповідає функції сусідства в WTM-правилі самонавчання [18]. При $\beta \to 1$ процедура (2.26) збігається з рекурентною формою алгоритму k-середніх чіткої кластеризації, а при $\beta = 0$ отримуємо стандартне WTA-правило самонавчання

$$\tilde{c}_j^k(k+1) = \tilde{c}_j^k(k) + \eta(k)\left(\tilde{\varphi}(k+1,h) - \tilde{c}_j^k(k)\right), \quad (2.27)$$

що збігається по суті з рекурентной процедурою оцінки середнього (2.17). Можна бачити, що (2.27) мінімізує цільову функцію

$$E\left(\tilde{c}_j^k\right)=\sum_k\left\|\tilde{\varphi}\left(k,h\right)-\tilde{c}_j^k\right\|^2$$

з параметром кроку

$$\eta\left(k\right)=\frac{1}{k+1},$$

тобто (2.27) в оптимальному варіанті набуває вигляду

$$\tilde{c}_j^k\left(k+1\right)=\tilde{c}_j^k\left(k\right)+\frac{1}{k+1}\left(\tilde{\varphi}\left(k+1,h\right)-\tilde{c}_j^k\left(k\right)\right). \tag{2.28}$$

Беручи до уваги (2.28), процедура кластеризації (2.26) остаточно приймає вигляд

$$\begin{cases} u_j\left(k+1\right)=\dfrac{\left\|\tilde{\varphi}\left(k+1,h\right)-\tilde{c}_j^k\left(k\right)\right\|^{\frac{2}{1-\beta}}}{\sum\limits_{l=1}^{m}\left\|\tilde{\varphi}\left(k+1,h\right)-\tilde{c}_l^k\left(k\right)\right\|^{\frac{2}{1-\beta}}}, \\ \tilde{c}_j^k\left(k+1\right)=\tilde{c}_j^k\left(k\right)+\dfrac{u_j^{\beta}\left(k+1\right)}{k+1}\left(\tilde{\varphi}\left(k+1,h\right)-\tilde{c}_j^k\left(k\right)\right) \end{cases} \tag{2.29}$$

і структурно близька до алгоритмів нечіткого конкурентного самонавчання, введеним в [136], [137].

І хоча алгоритми ймовірнісної нечіткої кластеризації (2.24), (2.25), (2.26), (2.29) досить ефективні при вирішенні багатьох практичних завдань, вони не позбавлені деяких істотних недоліків. По-перше, при високих розмінностях h оброблюваних сигналів вони страждають від ефекту «концентрації норм», а, по-друге, ці алгоритми не можна в повному розумінні називати нечіткими, оскільки

в них входить чітке значення фаззіфікатора $1<\beta<\infty$, що обирається зазвичай з емпіричних міркувань і істотно впливає на кінцеві результати.

У зв'язку з цим Ф. Клавонном і Ф. Хьоппнером було запропоновано замість цільової функції (2.21) з чітким фаззіфікатором β використовувати критерій виду

$$E\left(u_j(k),\tilde{c}_j^k\right)=\sum_{k=1}^{N}\sum_{j=1}^{m}\left(\alpha u_j^2(k)+(1-\alpha)u_j(k)\right)\left\|\tilde{\varphi}(k,h)-\tilde{c}_j^k\right\|^2 \tag{2.30}$$

з обмеженнями (2.22), (2.23), де $0<\alpha\le 1$ – параметр, що налаштовується, який визначає характер одержуваного рішення.

Вводячи функцію Лагранжа

$$\begin{aligned}L\left(u_j(k),\tilde{c}_j^k,\lambda(k)\right)=&\sum_{k=1}^{N}\sum_{j=1}^{m}\left(\alpha u_j^2(k)+(1-\alpha)u_j(k)\right)\left\|\tilde{\varphi}(k,h)-\tilde{c}_j\right\|^2+\\&+\sum_{k=1}^{N}\lambda(k)\left(\sum_{j=1}^{m}u_j-1\right)\end{aligned} \tag{2.31}$$

(тут $\lambda(k)-$ невизначені множники Лагранжа) і вирішуючи систему рівнянь Каруша–Куна–Таккера

$$\begin{cases}\dfrac{\partial L\left(u_j(k),\tilde{c}_j^k,\lambda(k)\right)}{\partial u_j(k)}=\left(2\alpha u_j(k)+1-\alpha\right)\left\|\tilde{\varphi}(k,h)-\tilde{c}_j^k\right\|^2+\lambda(k)=0,\\ \nabla\tilde{c}_j^k L\left(u_j(k),\tilde{c}_j^k,\lambda(k)\right)=-2\sum_{k=1}^{N}\left(\alpha u_j^2(k)+(1-\alpha)u_j(k)\right)\left(\tilde{\varphi}(k,h)-\tilde{c}_j^k\right)=\vec{0},\\ \dfrac{\partial L\left(u_j(k),\tilde{c}_j^k,\lambda(k)\right)}{\partial\lambda(k)}=\sum_{j=1}^{m}u_j(k)-1=0,\end{cases}$$

приходимо до вирішення

$$\begin{cases} u_j(k) = \dfrac{\alpha - 1}{2\alpha} + \left(1 + m\dfrac{1-\alpha}{2\alpha}\right) \dfrac{\left\|\tilde{\varphi}(k,h) - \tilde{c}_j^k\right\|^{-2}}{\sum\limits_{l=1}^{m} \left\|\tilde{\varphi}(k,h) - \tilde{c}_l^k\right\|^{-2}}, \\ \tilde{c}_j^k = \dfrac{\sum\limits_{k=1}^{N} \left(\alpha u_j(k) + (1-\alpha) u_j(k)\right) \tilde{\varphi}(k,h)}{\sum\limits_{k=1}^{N} \left(\alpha u_j(k) + (1-\alpha) u_j(k)\right)}, \\ \lambda(k) = \dfrac{1 + m\dfrac{1-\alpha}{2\alpha}}{\sum\limits_{l=1}^{m} \left(2\alpha \left\|\varphi(k,h) - \tilde{c}_l^k\right\|^2\right)^{-1}}. \end{cases} \tag{2.32}$$

Можна бачити, що при $\alpha = 1$ приходимо до FCM-алгоритму (2.25), а зменшення α надає отриманому рішенню більш чіткий характер.

Застосовуючи до (2.31) процедуру нелінійного програмування Ерроу–Гурвіца–Удзави, приходимо до рекурентного алгоритму нечіткої кластеризації за критерієм (2.30):

$$\begin{cases} u_j(k+1) = \dfrac{\alpha - 1}{2\alpha} + \left(1 + m\dfrac{1-\alpha}{2\alpha}\right) \dfrac{\left\|\tilde{\varphi}(k+1,h) - \tilde{c}_l^k(k)\right\|^{-2}}{\sum\limits_{l=1}^{m} \left\|\tilde{\varphi}(k+1,h) - \tilde{c}_l^k(k)\right\|^{-2}}, \\ \tilde{c}_j^k(k+1) = \tilde{c}_j^k(k) + \eta(k) \times \\ \qquad \times \left(\alpha u_j^2(k+1) + (1-\alpha) u_j(k+1) \left(\tilde{\varphi}(k+1,h) - \tilde{c}_j^k(k)\right)\right), \end{cases} \tag{2.33}$$

при цьому при $\alpha = 1$ алгоритм (2.33) збігається з (10) при $\beta = 2$. Нескладно також зауважити, що друге співвідношення (2.33) також є WTM-правилом самонавчання.

2.2 Послідовна кластеризація на основі ядерних нейронних мереж

В даний час штучні нейронні мережі набули широкого поширення для вирішення різного кола завдань, що виникають в рамках інтелектуального аналізу даних (Data Mining) таких, як прогнозування, класифікація, кластеризація і т. п. При цьому кластеризація в цьому ряду займає особливе місце, оскільки рішення цієї задачі ґрунтується на парадигмі самонавчання, що істотно ускладнює процес пошуку рішення. Тут найбільш популярними є BSB- і ART-штучні нейронні мережі, призначені для обробки інформації в пакетному режимі, і самоорганізовні карти Т. Кохонена (SOM), призначені для вирішення завдань кластеризації великих масивів інформації, завдяки простоті обчислювальної реалізації та можливості послідовної online обробки даних.

2.2.1 Ядерна кластерувальна нейронна мережа на основі радіально-базисної нейронної мережі

Серед самонавчанних систем обчислювального інтелекту особливо слід відзначити нейронні мережі Т. Кохонена (самоорганізовні карти, SOM), призначені для вирішення завдань кластеризації великих масивів інформації, завдяки своїй обчислювальної простоті, ефективності і можливості роботи в online режимі шляхом послідовної обробки даних у міру їх надходження.

Процес настройки цих мереж реалізується в режимі самонавчання на основі принципів «Переможець отримує все» (WTA) або «Переможець отримує більше» (WTM), при цьому апріорно передбачається, що вихідна структура даних така, що кластери взаємно не перетинаються і мають опуклу форму, тобто в процесі навчання нейронної мережі можуть бути побудовані розділяючі гіперплощини, що чітко розмежовують різні класи.

У випадку класів, що перетинаються, можуть бути використані методи нечіткого (fuzzy) кластерного аналізу, в тому числі нечіткі самоорганізовні карти, що реалізують в тій чи іншій формі метод нечітких С-середніх (FCM).

У випадку неопуклих класів ситуація видається більш складною. Тут слід зазначити, що мережі Кохонена фактично реалізують класичний метод К-середніх, а тому клітини Г. Вороного, що формуються в процесі самонавчання, обмежені гіперплощинами. Природно, що існують методи кластеризації для класів довільної форми, проте реалізують їх алгоритми досить складні з обчислювальної точки зору і жодним чином не призначені для вирішення завдань Dynamic Data Mining і Data Stream Mining, де інформація повинна оброблятися в реальному часі.

На сьогодні відомі, так звані, ядерні самоорганізовні карти (Kernel SOM), побудовані з використанням ядер Дж. Мерсера [138] (зазвичай типу потенційних функцій) і засновані на мінімізації критерію емпіричного ризику, що лежить в основі машин опорних векторів (SVM).

Безумовно, для вирішення різного кола завдань Data Mining можна використовувати нейронні мережі-машини опорних векторів, однак такі мережі можуть страждати від «прокляття розмірності», оскільки кількість нейронів даної мережі зростає зі збільшенням числа спостережень в вибірці даних. Саме тому слід використовувати ідеї, засновані на радіально-базисних нейронних мережах.

2.2.2 Архітектура ядерної самоорганізовної карти на основі радіально-базисної нейронної мережі.

На рис. 2.2 наведена архітектура даної ядерної самоорганізовної карти на основі радіально-базисної нейронної мережі.

Вихідною інформацією в даному випадку є центрована вибірка (можливо зростаюча) векторів спостережень $x(1),\ x(2),...,\ x(k),...,\ x(N),...;\ x(k) = \left(x_1(k),...,\ x_i(k),...,\ x_n(k)\right)^T \in R^n$ таких, що $-1 \le x_i(k) \le 1,\ \frac{1}{N}\sum_{k=1}^{N} x_1(k) = 0$, яка повинна бути розбита на m кластерів довільної форми, при цьому k тут може бути як номером спостережень, так і моментом поточного часу.

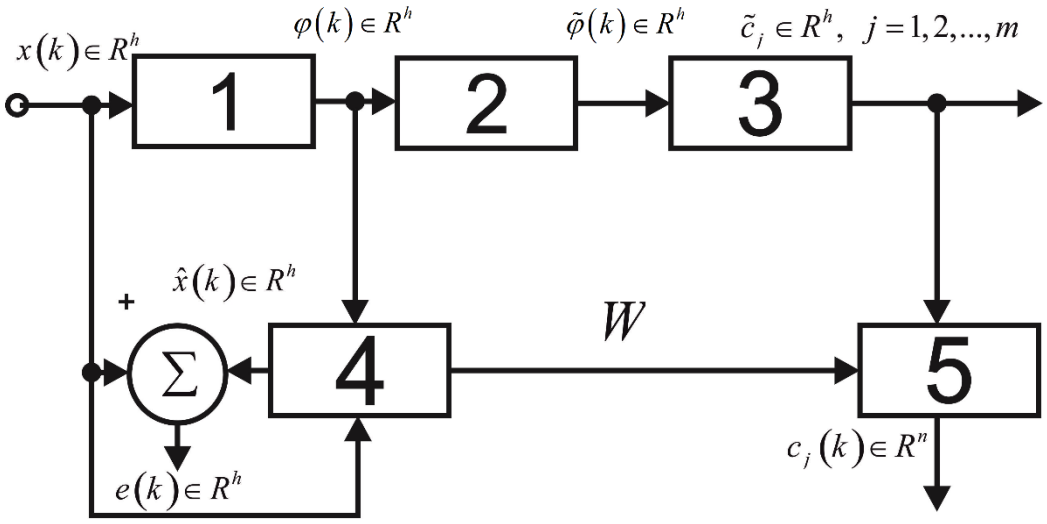


Рисунок 2.2 – Ядерна кластерувальна нейронна мережа на основі радіально-базисної нейронної мережі

Вектори спостережень $x(k)$ послідовно надходять на шар радіально-базисних функцій 1, що повністю співпадає за структурою з першим прихованим шаром стандартної радіально-базисної мережі і сформований ядерними (потенційними) функціями активації $\varphi_1,\ \varphi_2\ ,...,\ \varphi_l,...,\varphi_h\ \ (n+1 \le h \le N)$, за допомогою яких здійснюється підвищення розмірності вихідного простору входів. В якості таких функцій можуть бути використані традиційні Гауссіани:

$$\varphi_l = e^{-\frac{\|x-c_l\|^2}{2\sigma_l^2}}$$

хоча, звичайно можуть застосовуватися і інші дзвонуваті функції, а в якості їх центрів c_l в найпростішому випадку можуть бути взяті h довільно обраних векторів спостережень $c_l = x(l)$ (концепція «нейрони в точках даних»). Таким чином, при подачі на вхід системи вектора спостережень $x(k)$ на виході першого шару формується векторний сигнал

$$\varphi(k) = \left(\varphi_1(k),...,\ \varphi_l(k),...,\ \varphi_h(k)\right)^T \in R^h,$$

де

$$\varphi_l(k) = e^{-\frac{\|x(k)-c_l\|^2}{2\sigma_l^2}}.$$

Другий шар системи – шар нормалізації 2 – реалізує елементарне перетворення

$$\tilde{\varphi}_l(k) = \frac{\varphi_l(k)}{\|\varphi_l(k)\|},$$

необхідне для ефективної роботи третього шару – самоорганізовної карти 3.

Саме в цьому шарі і вирішується завдання кластеризації, тобто розбиття послідовності образів $\tilde{\varphi}(1),...,$ „ на m кластерів з уточненням в процесі самонавчання прототипів-центроїдів класів $\tilde{c}_j^K \in R^h$, $j = 1,2,...,m$.

Основна проблема полягає в тому, щоб ефективно сформувати базис, утворений радіально-базисними функціями, в якому можна було б зробити кластеризацію за допомогою карти Кохонена. Для цього, перш за все, необхідно оцінити, наскільки вдало було обрано кількість h і центри c_l радіально-базисних функцій першого шару.

Для цього призначений шар відновлення вхідного простору 4, що представляє собою по суті вихідний шар радіально-базисної мережі з h входами і n виходами і містить nh синаптичних ваг, що налаштовуються, і n суматорів.

Таким чином, перший і четвертий шари утворюють багатовиходову радіально-базисну нейронну мережу, відмінністю якої від стандартної є те, що в якості навчального тут використовується вхідний сигнал, тобто мережа працює в режимі автоасоціації. На виході четвертого шару формується сигнал $\hat{x}(k) \in \mathrm{R}^n$, що є оцінкою вхідного сигналу $x(k)$. Якість відновлення оцінюється на основі векторної помилки

$$e(k) = x(k) - \hat{x}(k)$$

за допомогою скалярного критерію

$$\overline{e} = \frac{1}{N}\sum_{k=1}^{N}\frac{\left\|x(k) - \hat{x}(k)\right\|}{\left\|x(k)\right\|}. \qquad (2.34)$$

Якщо виявиться, що значення $\overline{e}$ перевищує деякий заданий поріг e_T, приймається рішення про те, що кількість нейронів в першому шарі має бути збільшена. Цей процес триває до забезпечення необхідної якості відновлення вхідного простору. Результатом навчання четвертого шару є $(n \times h)$-матриця синаптичних ваг $W(N)$, отримана на підставі N спостережень.

Ця матриця є вихідною інформацією для п'ятого шару відновлення прототипів кластерів у вихідному просторі R^n. При цьому прототипи, сформовані самоорганізовною картою в h-вимірному просторі, проектуються в вихідний n-вимірний простір за допомогою елементарного перетворення

$$c_j^K(k) = W(N)\tilde{c}_j^K(k) \;\forall\; j = 1,2,...,m.$$

Таким чином, розглянута тут система є по суті об'єднанням двох нейромереж: еволюційної радіально-базисної нейронної мережі (ERBFN) і самоорганізовної карти Кохонена (SOM), які паралельно налаштовують свої синаптичні ваги в режимі самонавчання, одночасно з цим вирішуючи завдання кластеризації даних, що утворюють класи довільної форми.

2.2.3 Навчання ядерної самоорганізовної карти і радіально-базисної нейронної мережі

Навчання введеної системи може розглядатися як два відносно незалежні завдання: самонавчання радіально-базисної підсистеми і самонавчання власне самоорганізовної карти.

Завдання шару відновлення вхідного простору полягає в знаходженні $(n \times h)$-матриці синаптичних ваг $W = \{w_{il}\}$ за вибіркою, що містить N спостережень шляхом мінімізації критерію навчання

$$E^N = \sum_{k=1}^{N} E(k) = \frac{1}{2}\sum_{k=1}^{N} \|x(k) - W\varphi(k)\|^2 = \frac{1}{2}\sum_{k=1}^{N} \|e(k)\|^2. \qquad (2.35)$$

Мінімізація критерію (2.35) в пакетному варіанті веде до оцінки найменших квадратів виду

$$W(N) = (\sum_{k=1}^{N} x(k)\varphi^T(k))(\sum_{k=1}^{N} \varphi(k)\varphi^T(k))^{-1}$$

або в рекурентній формі –

$$\begin{cases} W(k) = W(k-1) + \dfrac{(x(k) - W(k-1)\varphi(k))\varphi^T(k)P(k-1)}{1 + \varphi^T(k)P(k-1)\varphi(k)}, \\ P(k) = P(k-1) - \dfrac{P(k-1)\varphi(k)\varphi^T(k)P(k-1)}{1 + \varphi^T(k)P(k-1)\varphi(k)}. \end{cases} \qquad (2.36)$$

Поліпшити якість відновлення вхідного простору можна, налаштовуючи не тільки синаптичні ваги четвертого шару, але і параметри центрів c_l і ширини σ_l активаційних функцій першого шару.

З урахуванням очевидних співвідношень

$$\begin{cases} \hat{x}(k)=\sum_{l=1}^{h} w_{il}(k-1)\varphi_l(k), \\ e_i^2(k)=(x_i(k)-\sum_{l=1}^{h} w_{il}(k-1)\varphi_l(k))^2, \\ E(k)=\frac{1}{2}\sum_{i=1}^{n} e_i^2(k), \\ \nabla_{c_l} E(k)=e_i(k)w_{il}(k-1)\varphi'\left(\frac{\|x(k)-c_l\|^2}{2\sigma_l^2}\right)\frac{x(k)-c_l}{\sigma_l^2}, \\ \frac{\partial E(k)}{\partial \sigma_l^{-2}}=-e_i(k)w_{il}(k-1)\varphi'\left(\frac{\|x(k)-c_l\|^2}{2\sigma_l^2}\right)\frac{\|x(k)-c_l\|^2}{2}, \end{cases}$$

можна ввести рекурентні градієнтні алгоритми навчання виду:

$$\begin{cases} c_l(k)=c_l(k-1)-\eta_c(k)e_i(k)w_{il}(k-1)\times e^{-\frac{\|x(k)-c_l(k-1)\|^2}{2\sigma_l^2(k-1)}}\frac{x(k)-c_l(k-1)}{\sigma_l^2(k-1)}, \\ \sigma_l^{-2}(k)=\sigma_l^{-2}(k-1)+\eta_\sigma(k)e_i(k)w_{il}(k-1)\times e^{-\frac{\|x(k)-c_l(k-1)\|^2}{2\sigma_l^2(k-1)}}\frac{\|x(k)-c_l(k-1)\|^2}{2}, \end{cases} \quad (2.37)$$

де $\eta_c(k),\eta_\sigma(k)$ – скалярні параметри кроку навчання.

Таким чином, співвідношення (2.36), (2.37) є процедурою навчання всіх параметрів радіально-базисної нейронної мережі, при цьому процес навчання (включаючи зміну числа радіально-базисних функцій) триває до досягнення необхідного значення критерію.

Навчання шару 3 – власне карти Кохонена – полягає в розбитті послідовності векторів-образів $\tilde{\varphi}(k)$ на m кластерів, кожен з яких характеризується власним прототипом-центроїдом $\tilde{c}_j^K(k)\in R^h$, $j=1,2,...,m$, що безперервно уточнюються при подачі чергового спостереження $\tilde{\varphi}(k)$. Кількість кластерів m покладається апріорно заданою.

Як і будь-яка інша процедура самонавчання, процес налаштування починається з ініціалізації синаптичних ваг мережі, в якості яких і виступають початкові значення прототипів $\tilde{c}_j^K(0)$. При цьому ці значення в процесі обробки нормуються подібно вхідним образам, тобто

$$\left\|\tilde{c}_j^K(k)\right\| = 1.$$

При подачі на вхід третього шару сигналу $\tilde{\varphi}(k)$ спочатку обчислюються m відстаней

$$D(\tilde{\varphi}(k), \tilde{c}_j^K(k-1)) = \left\|\tilde{\varphi}(k) - \tilde{c}_j^K(k-1)\right\| \;\forall\; j = 1, 2, ..., m, \tag{2.38}$$

при цьому якщо в якості відстаней використовується евклідова метрика, то замість (2.38) набагато зручніше використовувати міру подібності

$$SM\left(\tilde{\varphi}(k), \tilde{c}_j^K(k-1)\right) = \tilde{\varphi}^T(k)\tilde{c}_j^K(k-1) = \cos(\tilde{\varphi}(k), \tilde{c}_j^K(k-1)) = \cos\theta_j(k). \tag{2.39}$$

На підставі (2.38) або (2.39) визначається нейрон-переможець, «найближчий» до вхідного образу такий, що

$$D\left(\tilde{\varphi}(k), \tilde{c}_*^K(k-1)\right) = \min_j D\left(\tilde{\varphi}(k), \tilde{c}_j^K(k-1)\right)$$

або

$$SM(\tilde{\varphi}(k), \tilde{c}_*^K(k-1)) = \max_j SM(\tilde{\varphi}(k), \tilde{c}_j^K(k-1)).$$

Далі налаштовуються ваги цього нейрона за допомогою правила самонавчання Кохонена в формі

$$\tilde{c}_j^K(k)=\begin{cases}\dfrac{\tilde{c}_j^K(k-1)+\eta(k)\left(\tilde{\varphi}(k)-\tilde{c}_j^K(k-1)\right)}{\left\|\tilde{c}_j^K(k-1)+\eta(k)\left(\tilde{\varphi}(k)-\tilde{c}_j^K(k-1)\right)\right\|}, \\ \textit{якщо } j-\textit{й нейрон переміг}, \\ \\ \tilde{c}_j^K(k-1) \textit{ у протилежному випадку}.\end{cases} \tag{2.40}$$

Наявність знаменника в першому співвідношенні (2.40) автоматично забезпечує протікання процесу навчання на одиничній гіперкулі.

Процедура (2.40) реалізує принцип «Переможець отримує все» (WTA), при цьому вектор синаптичних ваг нейрона-переможця $\tilde{c}_*^K(k-1)$ «підтягується» до вхідного образу $\tilde{\varphi}(k)$ на відстань, що визначається кроком пошуку $0<\eta(k)<1$.

Регулювання кроку пошуку зазвичай проводиться, виходячи з емпіричних міркувань, а загальна рекомендація зводиться до того, що крок повинен монотонно зменшуватися в процесі самонавчання. Ця умова може бути забезпечена при виборі кроку згідно зі співвідношенням

$$\eta(k)=r^{-1}(k),\, r(k)=\alpha r(k-1)+\left\|\tilde{\varphi}(k)\right\|^2=\alpha r(k-1)+1, 0<\alpha\le 1,$$

при цьому при $\alpha=1$ параметр кроку $\eta(k)=\dfrac{1}{k}$ задовольняє умовам А. Дворецького, а одержувані результати збігаються з оцінками K-середніх.

Варіюючи фактором забування α, можна забезпечити широкий інтервал оновлення кроку

$$\frac{1}{k}\le\eta(k)\le 1.$$

В результаті пред'явлення самоорганізовній мапі N образів $\tilde{\varphi}(k)$ буде отримано m прототипів-центроїдів $\tilde{c}_j^K(N)$, які далі з простору підвищеної розмірності R^h можуть бути спроектовані в початковий простір R^n за допомогою простого співвідношення

$$c_j^K(N) = W(N)\tilde{c}_j^K(N) \;\; \forall \; j = 1,2,...,m.$$

2.2.4 Ядерна кластеризація на основі узагальненої регресійної нейронної мережі та самоорганізовної карти Кохонена

Як вже зазначалося вище, для вирішення завдань кластеризації в ситуаціях, коли класи мають довільну форму, можуть бути використані, так звані, ядерні самоорганізовні карти (KSOM), побудовані з використанням ядер Дж. Мерсера і засновані на мінімізації критерію емпіричного ризику, що лежить в основі, так званих, машин опорних векторів (SVM), введених В. Н. Вапніком. Недоліком нейронної мережі опорних векторів є те, що кількість нейронів такої мережі визначається обсягом вибірки даних, тому дана мережа не підходить для online обробки.

Нижче на рис. 2.3 наведено архітектуру ядерної самоорганізовної карти на основі узагальненої регресійної нейронної мережі.

Вихідною інформацією для даної мережі є вибірка (можливо зростаюча) векторів спостережень $x(1), x(2), ..., x(k), ..., x(N), ...;$ $x(k) = (x_1(k), x_2(k)..., x_i(k), ..., x_N(k))^T \in R^n$, яка повинна бути розбита на m кластерів довільної форми, при цьому k тут може бути як номером спостереження, так і моментом поточного часу.

Вектори спостережень $x(k)$ послідовно надходять на перший шар радіально-базисних функцій (R-нейронів), що повністю співпадає за структурою з першим шаром (шаром образів) стандартної узагальненої регресійної мережі

Д. Шпехта і сформований ядерними дзвонуватими функціями активації $\varphi_1,...,\varphi_k,...,\varphi_N$, за допомогою яких здійснюється підвищення розмірності вхідного простору. В якості таких функцій зазвичай використовуються традиційні гаусіани, а налаштування цього шару забезпечується за допомогою «лінивого» навчання на основі концепції «нейрони в точках даних». При цьому в якості центрів активаційних функцій приймаються самі оброблювані вектори-образи. Таким чином, при подачі на вхід нейронної мережі деякого некласифікованого образу x, на виходах R-нейронів першого шару з'являються значення

$$\varphi_k(x) = e^{-\frac{\|x-x(k)\|^2}{2\sigma^2}}, k = 1,2,...,N$$

(тут σ^2 – параметр рецепторного поля дзвонуватої функції), а на виході GRNN в цілому – сигнал

$$\hat{y}(x) = \frac{\sum_{k=1}^{N} y(k)\varphi_k(x)}{\sum_{k=1}^{N} \varphi_k(x)}, \qquad (2.41)$$

де $y(k)$ – зовнішній навчальний сигнал, відповідний образу $x(k)$. Зрозуміло, що в задачі кластеризації навчальний сигнал відсутній як такий, а сама GRNN у загальному випадку орієнтована на вирішення завдань інтерполяції, а не кластеризації.

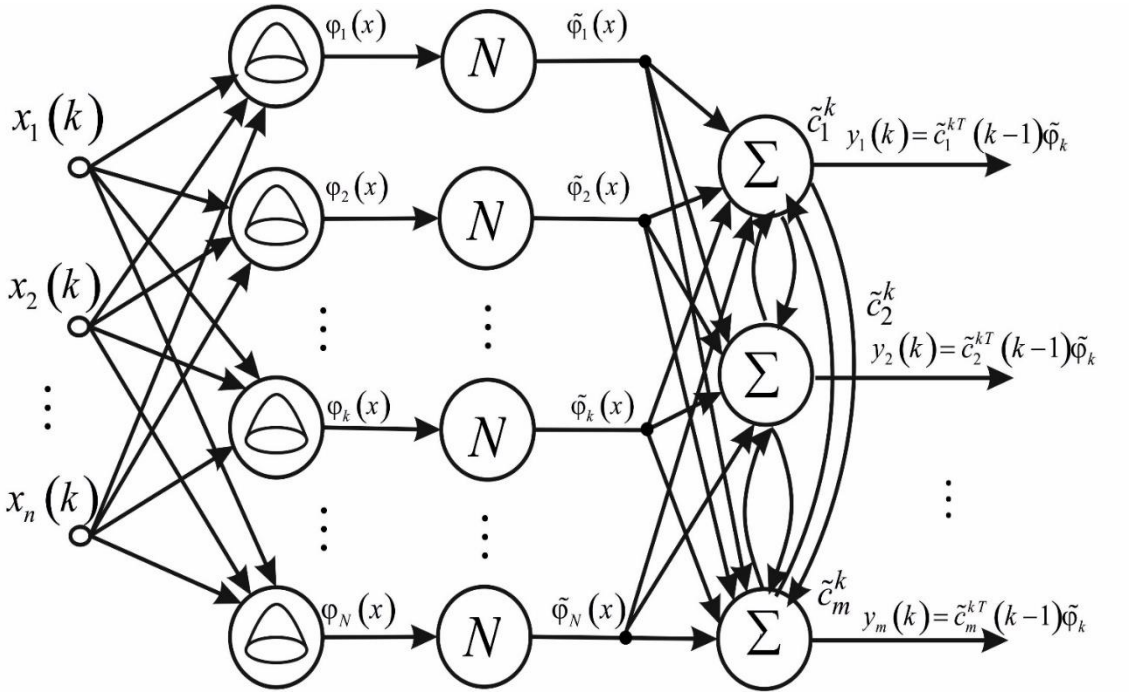


Рисунок 2.3 – Архітектура ядерної самоорганізовної карти Т. Кохонена, що заснована на узагальненій регресійній нейронній мережі

Другий прихований шар розглянутої мережі – шар нормалізації реалізує елементарне перетворення

$$\tilde{\varphi}(x)=\frac{\varphi(x)}{\|\varphi(x)\|}$$

(тут $(\varphi(x)=(\varphi_1(x),\varphi_2(x),...,\varphi_N(x))^T$), необхідне для обробки інформації вихідним шаром, що є по суті кластерувальною нейронною мережею Т. Кохонена, настройка параметрів якої проводиться на основі конкурентного самонавчання. У цьому вихідному шарі вирішується завдання розбиття послідовності образів підвищеної розмірності $\tilde{\varphi}_1,...,\tilde{\varphi}_2,...,\tilde{\varphi}_k,...,\tilde{\varphi}_N$ на m кластерів з прототипами-центроїдами $\tilde{c}_1^{\,K},\tilde{c}_2^{\,K},...,\tilde{c}_m^{\,K}\in R^N$.

Незважаючи на простоту, при реалізації цього підходу можуть виникати суттєві обчислювальні проблеми при великому обсязі N оброблюваної вибірки,

оскільки мережа, яка містить N нейронів, стає занадто громіздкою. У зв'язку з цим є доцільним замість традиційної процедури навчання GRNN ввести метод, що дозволяє не тільки налаштувати параметри мережі, а й істотно скоротити число її R-нейронів.

2.2.5 Навчання ядерної самоорганізовної карти на основі узагальненої регресійної нейронної мережі

Для формування першого шару даної гібридної нейронної мережі можна скористатися ідеями, що лежать в основі еволюційних систем обчислювального інтелекту, адаптованими до online обробки інформації, що послідовно надходить в систему [132]. Реалізація цього підходу проводиться в формі наступної послідовності кроків:

Крок 0: задати поріг нерозрізненості векторів центрів активаційних функцій Δ, максимально допустиму кількість нейронів в першому шарі $H \le N$ і параметр ширини рецепторного поля σ^2.

Крок 1: при надходженні спостереження $x(1)$ формується перший центр $c_1 = x(1)$ і сама активаційна функція

$$\varphi_1(x) = e^{-\frac{\|x-c_1\|^2}{2\sigma^2}} = e^{-\frac{\|x-x(1)\|^2}{2\sigma^2}}.$$

Крок 2: при надходженні спостереження $x(2)$ перевіряється умова

$$\|x(2) - c_1\|^2 \le \Delta. \qquad (2.42)$$

Якщо вона виконується, то спостереження $x(2)$ не формує новий центр і автоматично вважається належним до того ж кластера, що і $x(1)$. Якщо виконується умова

$$\Delta < \left\| x(2) - c_1 \right\| \le 2\Delta ,$$

то відбувається корекція c_1 згідно з WTA-правилом самонавчання Т. Кохонена у формі

$$c_1(2) = c_1(1) + \eta(2)(x(2) - c_1(1))$$

(тут $0 < \eta(2) < 1$ – параметр кроку налаштування). Якщо ж виконується умова

$$2\Delta \le \left\| x(2) - c_1 \right\|,$$

то формується друга активаційна функція

$$\varphi_2(x) = e^{-\frac{\|x - c_2\|^2}{2\sigma^2}} = e^{-\frac{\|x - x(2)\|^2}{2\sigma^2}} .$$

Крок N: Якщо до k моменту надходження N-го вектора-образу $x(N)$ сформовано $h \le H$ активаційних функцій, процес нарощування числа R-нейронів першого шару закінчується, і надалі структура цього шару залишається незмінною.

Оцінити якість функціонування першого шару можна було б, скориставшись виразом (2.41), яке для $h = H = N$ набуває вигляду

$$\hat{x}(k) = \frac{\sum_{k=1}^{N} x(k)\varphi_k(x(k))}{\sum_{k=1}^{N} \varphi_k(x(k))}, \qquad (2.43)$$

після чого оцінити похибку відновлення вхідних образів:

$$\varepsilon = \frac{1}{N}\sum_{k=1}^{N}\frac{\left\|x(k)-\hat{x}(k)\right\|}{\left\|x(k)\right\|}. \qquad (2.44)$$

Однак, оскільки в процесі формування першого шару за допомогою описаного вище підходу деякі з центрів активаційних функцій не збігаються з векторами спостережень, використання виразу (2.43), коректного для «класичної» GRNN, в нашому випадку може виявитися неправомірним.

У цій ситуації можна скористатися модифікованою GRNN, архітектура якої наведена на рис. 2.4.

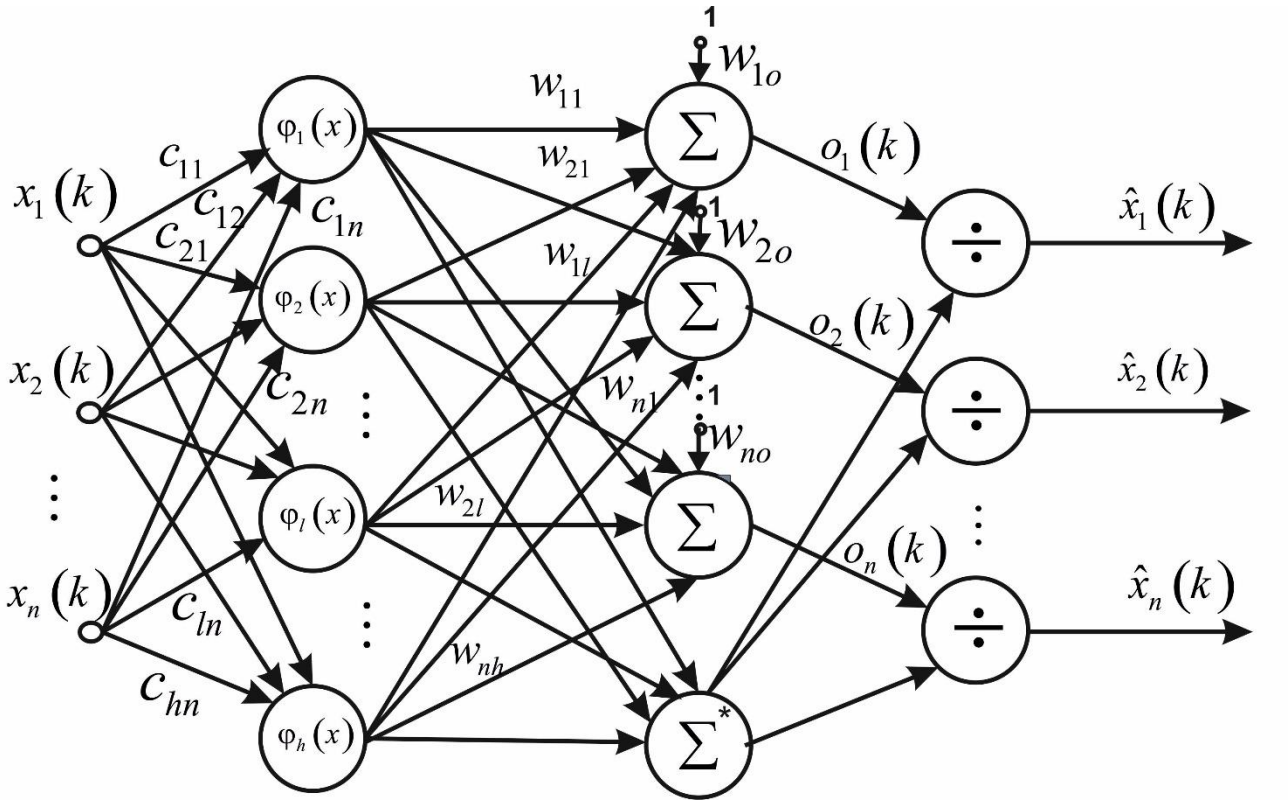


Рисунок 2.4 – Модифікована узагальнена регресійна нейронна мережа

Дана мережа, подібно тришаровому персептрону, містить три шари обробки інформації, проте, як активаційні функції використовує радіально-базисні конструкції в першому прихованому шарі. Другий прихований шар містить $n+1$ вузлів, n з яких є адаптивними лінійними асоціаторами, а $(n+1)$-й

– стандартним блоком підсумовування $\sum^*$. Вихідний сигнал мережі утворено n блоками поділу $\div$.

Навчання такої мережі є комбінованим процесом установки центрів радіально-базисних функцій на основі методів еволюційних систем і навчання з вчителем синаптичних ваг лінійних асоціаторів. При цьому в якості навчального тут використовується сам вхідний сигнал, тобто мережа налаштовується в автоасоціативному режимі.

Другий прихований шар мережі налаштовується аналогічно процесу навчання радіально-базисних нейронних мереж. При цьому на виходах n адаптивних лінійних асоціаторів формуються сигнали

$$o_i(k) = \sum_{l=0}^{h} w_{il}(k)\varphi_l(x(k)),\ \ i = 1,2...,n,$$

а на виході сумматоров $\sum^*$ з'являється сума $\sum_{l=0}^{h}\varphi_l(x(k))$.

Вихідний шар забезпечує нормування вихідного сигналу по типу нормованої радіально-базисної мережі (NRBFN) так, що

$$\hat{x}_i(k) = \frac{\sum_{l=0}^{h} w_{il}(k)\varphi_l(x(k))}{\sum_{l=0}^{h}\varphi_l(x(k))} = \sum_{l=0}^{h} w_{il}(k)\frac{\varphi_l(x(k))}{\sum_{l=0}^{h}\varphi_l(x(k))} = \sum_{l=0}^{h} w_{il}(k)\varphi_l^*(x(k)), \qquad (2.45)$$

$$\varphi_0(x(k)) = 1,\ \varphi_l^*(x(k)) = \varphi_l(x(k))(\sum_{l=0}^{h}\varphi_l(x(k)))^{-1},$$

або

$$\hat{x}(k) = W(k)\varphi^*(k), \qquad (2.46)$$

де

$$\hat{x}_i(k) = (\hat{x}_1(k), ...\hat{x}_i(k), ..., \hat{x}_n(k))^T,$$

$$\varphi^*(k) = (\varphi_0^*(x(k)), \varphi_1^*(x(k)), ..., \varphi_h^*(x(k)))^T,$$

$\mathrm{W}(k)$ – $(n \times (h+1))$-матриця синаптичних ваг, що підлягає визначенню.

Для налаштування матриці синаптичних ваг $\mathrm{W}(k)$ може бути використаний рекурентний метод найменших квадратів, який є по суті оптимальною за швидкодією гаусівсько-ньютонівською процедурою оптимізації виду:

$$\begin{cases} W(k) = W(k-1) + \dfrac{(x(k) - W(k-1)\varphi^*(k))\varphi^{*T}(k)P(k-1)}{1 + \varphi^{*T}(k)P(k-1)\varphi^*(k)}, \\ P(k) = P(k-1) - \dfrac{P(k-1)\varphi^*(k)\varphi^{*T}(k)P(k-1)}{1 + \varphi^{*T}(k)P(k-1)\varphi^*(k)}. \end{cases}$$

Таким чином можна оцінити якість роботи першого шару, використовуючи у виразі (2.44) замість стандартного співвідношення (2.43) введені вище оцінки (2.45), (2.46).

Вихідний сигнал синтезованого першого шару в формі $(h \times 1)$-вектора $\varphi(x) = (\varphi_1(x)..., \varphi_l(x), ..., \varphi_h(x))^T$ у другому прихованому шарі нормалізується до виду

$$\tilde{\varphi}(x) = \varphi(x) \left\| \varphi(x) \right\|^{-1},$$

тобто проектується на h-вимірну гіперсферу одиничного радіусу, після чого у вигляді послідовності $\tilde{\varphi}_1, \tilde{\varphi}_2, ..., \tilde{\varphi}_k, ..., \tilde{\varphi}_N$ надходить на вхід самоорганізовної карти Кохонена.

Налаштування карти Кохонена, утвореної m адаптивними лінійними асоціаторами, проводиться на основі WTM-правила самонавчання («Переможець отримує більше») і полягає в розбитті послідовності нормованих векторів-образів $\tilde{\varphi}_1, \tilde{\varphi}_2, ..., \tilde{\varphi}_N$ на m кластерів, кожен з яких характеризується власним прототипом-центроїдом $\tilde{c}_j^K \in R^h$, $j = 1, 2, ..., m$, що безперервно уточнюється при надходженні чергового образу підвищеної розмірності $\tilde{\varphi}_k$.

Процес самонавчання починається з ініціалізації синаптичних ваг вихідного шару, в якості яких виступають досить довільно обрані початкові значення прототипів $\tilde{c}_j^K(0)$ такі, що

$$\left\| \tilde{c}_j^K(0) \right\| = 1.$$

При подачі на вхід третього шару сигналу $\tilde{\varphi}_1$ обчислюється m відстаней

$$D(\tilde{\varphi}_1, \tilde{c}_j^K(0)) = \left\| \tilde{\varphi}_1 - \tilde{c}_j^K(0) \right\| \; \forall j = 1, 2, ..., m,$$

на підставі яких оцінюється нейрон-переможець, для якого

$$D(\tilde{\varphi}_1, \tilde{c}_*^K(0)) = \min_j D(\tilde{\varphi}_1, \tilde{c}_j^K(0)).$$

Після цього проводиться перший крок налаштування ваг-центроїдів

$$\tilde{c}_l^K(1) = \frac{\tilde{c}_l^K(0) + \eta(1)\psi(\tilde{c}_*^K(0), \tilde{c}_l^K(0))(\tilde{\varphi}_1 - \tilde{c}_l^K(0))}{\left\| \tilde{c}_l^K(0) + \eta(1)\psi(\tilde{c}_*^K(0), \tilde{c}_l^K(0))(\tilde{\varphi}_1 - \tilde{c}_l^K(0)) \right\|}, \; \forall l = 1, 2, ..., m.$$

Аналогічним чином можна записати правило самонавчання для k-го вектора-образа

$$\tilde{c}_l^K(k)=\frac{\tilde{c}_l^K(k-1)+\eta(k)\psi(\tilde{c}_*^K(k-1),\tilde{c}_l^K(k-1)(\tilde{\varphi}_k-\tilde{c}_l^K(k-1))}{\left\|\tilde{c}_l^K(k-1)+\eta(k)\psi(\tilde{c}_*^K(k-1),\tilde{c}_l^K(k-1))(\tilde{\varphi}_k-\tilde{c}_l^K(k-1))\right\|},\quad \forall l=1,2,...,m, \tag{2.47}$$

де $\psi(\tilde{c}_*^K(k-1),\tilde{c}_l^K(k-1))$ – так звана, функція сусідства, яка визначає локальну область топологічного сусідства, в якій налаштовуються не тільки нейрон-переможець $\tilde{c}_*^K$, але і його найближче оточення, при цьому більш близькі до переможця нейрони підтягуються до вхідного вектору $\tilde{\varphi}_k$ більше, ніж віддалені від $\tilde{c}_*^K$ центроїди $\tilde{c}_l^K$.

В якості функції сусідства зазвичай використовується все той же Гаусіан, який отримує в даному випадку вид

$$\psi(\tilde{c}_*^K(k),\tilde{c}_l^K(k))=e^{-\frac{\left\|\tilde{c}_k^*(k)-\tilde{c}_l^K(k)\right\|^2}{2\sigma_c^2}},$$

де σ_C^2 визначає розміри області сусідства, при цьому в процесі навчання цей параметр повинен монотонно зменшуватися.

В принципі, для навчання самоорганізовної карти можна взагалі не визначати переможця, а в якості функції сусідства використовувати вихідний сигнал кожного нейрона

$$y_l(k)=\tilde{\varphi}_k^T\tilde{c}_l^K, \tag{2.48}$$

при цьому правило (2.47) може бути переписано у вигляді

$$\tilde{c}_l^K(k)=\frac{\tilde{c}_l^K(k-1)+\eta(k)y_l(k)(\tilde{\varphi}_k-\tilde{c}_l^K(k-1))}{\left\|\tilde{c}_l^K(k-1)+\eta(k)y_l(k)(\tilde{\varphi}_k-\tilde{c}_l^K(k-1))\right\|},\ \forall l=1,2,...,m. \tag{2.49}$$

Помітивши, що

$$\|\tilde{\varphi}_k\| = \|\tilde{c}_l^K(k-1)\| = 1,$$

нескладно встановити, що вираз (2.48) є не що інше, як косинус кута між вхідним образом $\tilde{\varphi}_k$ і вектором-центроїдом $\tilde{c}_l^K(k-1)$, тобто $\cos(\tilde{\varphi}_k, \tilde{c}_l^K(k-1))$. Тоді з урахуванням невід'ємності функції сусідства можна остаточно переписати правило самонавчання у вигляді

$$\tilde{c}_l^K(k) = \frac{\tilde{c}_l^K(k-1) + \eta(k)[\cos(\tilde{\varphi}_k, \tilde{c}_l^K(k-1))]_+(\tilde{\varphi}_k - \tilde{c}_l^K(k-1))}{\left\|\tilde{c}_l^K(k-1) + \eta(k)[\cos(\tilde{\varphi}_k, \tilde{c}_l^K(k-1))]_+(\tilde{\varphi}_k - \tilde{c}_l^K(k-1))\right\|}, \forall l = 1,2,...,m \qquad (2.50)$$

де $[\bullet]_+$ – проектор на позитивний ортант.

Процес кластеризації закінчується або після вичерпання вибірки, що містить N спостережень, або відбувається безперервно, якщо дані надходять у формі потоку в online режимі.

2.3 Адаптивний метод комбінованого навчання-самонавчання нейро-фаззі системи

В даному підрозділі розглядаються архітектури та алгоритми самонавчання нейро-фаззі гібридних систем обчислювального інтелекту, які призначені для кластеризації даних в умовах, коли кластери можуть мати довільну форму і взаємно перетинатися. Введені процедури самонавчання досить прості в чисельної реалізації і призначені для обробки даних, що послідовно в online режимі надходять в систему.

2.3.1 Послідовна ядерна кластеризація на основі нейро-фаззі підходу

На сьогоднішній день розроблено безліч різноманітних методів кластеризації, при цьому переважна їх більшість обробляє інформацію в припущенні, що весь масив вихідних даних сформований заздалегідь, його обсяг не змінюється в процесі аналізу, а кожне спостереження-вектор ознак може оброблятися багаторазово. Зрозуміло, що саме це вихідне припущення унеможливлює застосування традиційного підходу до кластеризації в ситуаціях, коли дані надходять на обробку послідовно у вигляді потоку інформації або їх обсяг настільки великий (Big Data), що багаторазовий аналіз окремого спостереження просто неможливий.

У зв'язку з цим в задачах ядерної кластеризації кращим з позицій Big Data представляється використання підходу, заснованого на оцінках Е. Парзена, узагальнених регресійних нейронних мережах (GRNN) Д. Шпехта, заснованих на «лінивому навчанні» за принципом «нейрони в точках даних», і теоремі Т. Кавера про можливість лінійної роздільності класів в просторах підвищеної розмірності. І хоча подібний підхід також схильний до прокляття розмірності і орієнтований на пакетний режим роботи, він може бути адаптований до online обробки великих масивів інформації на основі гібридизації нейро-фаззі підходу і еволюційних систем обчислювального інтелекту, що дозволяє в процесі самонавчання «вирощувати» архітектуру, найкращим чином орієнтовану на вирішення конкретного завдання нечіткої кластеризації даних.

У зв'язку з цим представляється доцільною розробка online нейро-фаззі систем для вирішення задачі послідовної нечіткої кластеризації даних, що дозволяють обробляти вектори спостережень будь-якої розмірності в умовах обмеженого числа спостережень в оброблюваної вибірці.

На рис. 2.5 наведена архітектура пропонованої ядерної кластерувальної online нейро-фаззі системи. Вихідною інформацією для цієї системи є центрована щодо середнього вибірка векторів спостережень, можливо зростаюча $x(1),\ x(2),...,\ x(k),...,\ x(N),...;\ x(k) = \left(x_1(k),...,\ x_i(k),...,\ x_n(k)\right)^T \in R^n$, таких, що

$-1 \le x_i(k) \le 1$, $\frac{1}{N}\sum_{k=1}^{N} x_i(k) = 0$, яка повинна бути розбита на m кластерів довільної форми, при цьому k тут може бути як номером поточного спостереження, так і моментом поточного часу.

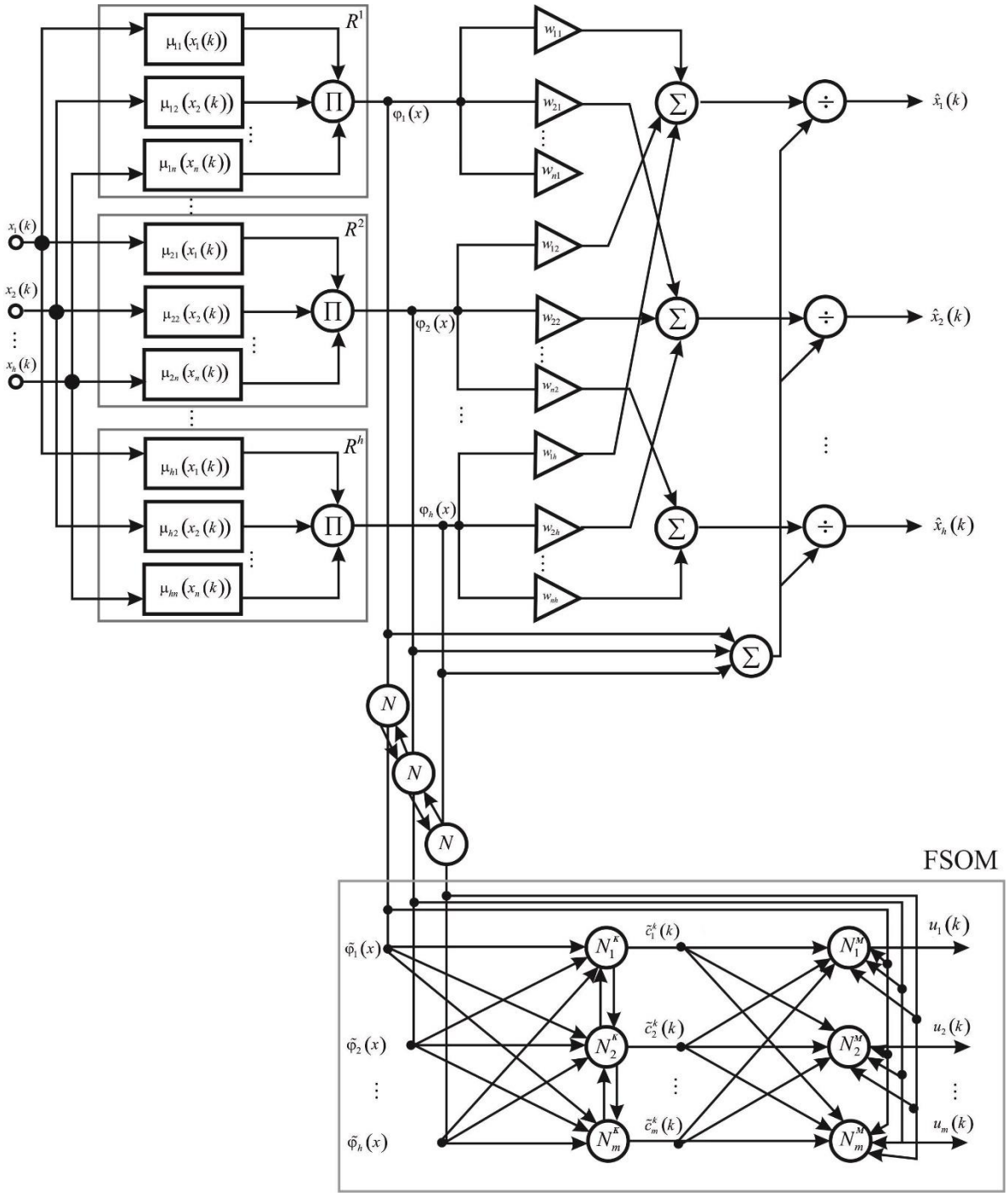


Рисунок 2.5 – Архітектура ядерної кластерувальної нейро-фаззі системи.

Вектори спостережень $x(k)$ послідовно надходять на нульовий (рецепторний) шар системи, звідки передаються на перший прихований шар, утворений nh (по h на кожен вхід) функціями $\mu_{li}(x_i), l=1,2,...,h; i=1,2,...,n$ і виконує фаззіфікацію вхідного простору R^n.

Другий прихований шар забезпечує агрегування рівнів належності, розрахованих в першому шарі, і містить h блоків множення.

Таким чином, перші два шари даної системи повністю збігаються з шарами найбільш популярних нейро-фаззі систем ANFIS і TSK, основною перевагою яких крім універсальних аппроксимувальних властивостей є те, що їх вихідний сигнал лінійно залежить від параметрів, що налаштовуються, – синаптичних ваг, що дозволяє використовувати для їх налаштування як безліч відомих лінійних алгоритмів навчання, так і відомі методи адаптивної кластеризації, оптимальні за швидкодією.

Отже, якщо на вхід системи поданий векторний сигнал $x(k)$, елементи першого прихованого шару проводять його фаззіфікацію, обчислюючи рівні належності $0 < \mu_{li}(x_i(k)) \le 1$, при цьому зазвичай в якості функцій належності використовуються традиційні гауссіани

$$\mu_{li}(x_i(k)) = \exp(-\frac{(x_i(k)-c_{li})^2}{2\sigma_i^2}),$$

де c_{li}, σ_i – параметри центрів і ширини відповідно.

Зауважимо також, що попереднє кодування даних на деякий інтервал, наприклад, $-1 \le x_i(k) \le 1$ дозволяє спростити розрахунки, оскільки параметри ширини σ_i в цьому випадку можуть бути прийняті однаковими для всіх входів, тобто $\sigma_i = \sigma$.

У другому прихованому шарі обчислюються агреговані значення

$$\varphi_l(k) = \prod_{i=1}^{n} \mu_{li}(x_i(k)),$$

при цьому для гауссіанів з однаковими параметрами ширини справедливо співвідношення

$$\varphi_l(k) = \prod_{i=1}^{n} \mu_{li}(x_i(k)) = \prod_{i=1}^{n} \exp(-\frac{(x_i(k) - c_{li})^2}{2\sigma^2}) = \exp(-\frac{\|x(k) - c_l\|^2}{2\sigma^2})$$

(тут $c_l = (c_{l1}, c_{l2}, ..., c_{\ln})^T$), тобто сигнали на виходах блоків множення другого прихованого шару аналогічні сигналам на виходах нейронів R_l першого прихованого шару звичайних радіально-базисних нейронних мереж (RBFN).

Третій прихований шар даної системи – шар нормалізації (N) реалізує елементарне перетворення

$$\tilde{\varphi}_l(k) = \frac{\varphi_l(k)}{\|\varphi_l(k)\|}$$

(тут $\varphi(x) = (\varphi_1(x), ..., \varphi_l(x), ..., \varphi_h(x)^T)$), необхідне для ефективної роботи четвертого вихідного шару, що є по суті кластерувальною нейро-фаззі мережею Кохонена (FSOM), налаштування параметрів якої проводиться за допомогою спеціалізованої процедури конкурентного самонавчання. У цьому шарі вирішується завдання розбиття послідовностей образів підвищеної розмірності $\tilde{\varphi}(x(1)), ..., \tilde{\varphi}(x(k)), ..., \tilde{\varphi}(x(N)), ...$ на m кластерів з прототипами-центроїдами $\tilde{c}_1^k, ..., \tilde{c}_j^k, ..., \tilde{c}_m^k$ (нейрони N_j^k) і рівнями належності кожного $\tilde{\varphi}(k)$ до кожного j-го кластеру $u_j(k)$ (нейрони N_j^u), $\tilde{c}_1^k \in R^h$, $j = 1, 2, ..., m$.

Важлива проблема, що виникає в процесі синтезу системи, полягає в тому, як ефективно організувати базис, утворений ядерними функціями, в якому

можна було б точно провести кластеризацію. Для цього необхідно організувати процес адаптації кількості $h > n$ і центрів c_{li} функцій належності першого шару.

Для вирішення цього завдання призначені п’ятий, шостий і сьомий додаткові шари, утворені nh налаштованими синаптичними вагами, $n+1$ суматорами і n блоками поділу.

Таким чином, перший, другий, п’ятий, шостий і сьомий шари даної системи утворюють по суті багатовиходову нейро-фаззі систему Ванга–Менделя (TSK-система нульового порядку), основною відмінністю якої є те, що в якості навчального тут використовується вхідний сигнал $x(k)$, тобто система працює в режимі автоасоціації. Виходом сьомого шару є векторний сигнал $\hat{x}(k) \in R^n$, що є оцінкою вхідного сигналу $x(k)$.

Якість відновлення оцінюється на основі векторної помилки

$$e(k) = x(k) - \hat{x}(k)$$

за допомогою того чи іншого скалярного критерію, наприклад,

$$\overline{e} = \frac{1}{N}\sum_{k=1}^{N}\frac{\left\|x(k) - \hat{x}(k)\right\|}{\left\|x\right\|^{\bullet}}$$

(тут $\left\|\bullet\right\|$ – будь-яка норма в метриці Мінковського) або MAPE-оцінки

$$\overline{e} = \frac{1}{N}\sum_{k=1}^{N}\frac{\sum_{i=1}^{n}\left|x_i(k) - \hat{x}_i(k)\right|}{\sum_{i=1}^{n}\left|x_i(k)\right|} \times 100\%.$$

Якщо виявиться, що значення $\overline{e}$ перевищує деякий апріорно заданий поріг e_T, приймається рішення про те, що процес налаштування цих шарів повинен

тривати, тобто h, c_{li}, w_{il} підлягають подальшому уточненню. Кінцевим результатом навчання цих шарів є значення h, c_{li} і $(n \times h)$-матриця синаптичних ваг $w = \{w_{il}\}$.

Необхідно також зауважити, що на виходах п'ятого прихованого шару формується nh сигналів

$$w_{il} \prod_{i=1}^{n} \mu_{li}(x_i(k)) = w_{il} \varphi_l(k),$$

шостого – $n+1$ сигналів

$$\sum_{l=1}^{h} w_{il} \prod_{i=1}^{n} \mu_{li}(x_i(k)) = \sum_{l=1}^{h} w_{il} \varphi_l(k),$$

$$\sum_{l=1}^{h} \prod_{i=1}^{n} \mu_{li}(x_i(k)) = \sum_{l=1}^{h} \varphi_l(k),$$

де

$$\hat{x}_i(k) = \frac{\sum_{l=1}^{h} w_{il} \prod_{i=1}^{n} \mu_{li}(x_i(k))}{\sum_{l=1}^{h} \prod_{i=1}^{n} \mu_{li}(x_i(k))} = \sum_{l=1}^{h} w_{il} \frac{\prod_{i=1}^{n} \mu_{li}(x_i(k))}{\sum_{l=1}^{h} \prod_{i=1}^{n} \mu_{li}(x_i(k))} = \sum_{l=1}^{h} w_{il} \hat{\varphi}_l(x(k)) = w_i^T \hat{\varphi}^h(x(k)),$$

$$\hat{\varphi}_l(x(k)) = \frac{\prod_{i=1}^{n} \mu_{li}(x_i(k))}{\sum_{l=1}^{h} \prod_{i=1}^{n} \mu_{li}(x_i(k))},$$

$$w_i = (w_{i1}, w_{i2}, ..., w_{ih})^T,$$

$$\hat{\varphi}^h(x(k)) = (\hat{\varphi}_1(x(k)), ..., \hat{\varphi}_l(x(k)), ..., \hat{\varphi}_h(x(k)))^T.$$

Вводячи далі в розгляд $(n \times 1)$ вектор $\hat{x}(k) = (\hat{x}_1(k), ..., \hat{x}_i(k), ..., \hat{x}_n(k))^T$ і $(n \times h)$-матрицю $W = (w_1, w_i, ..., w_n)^T$, остаточно можна записати

$$\hat{x}(k) = W\hat{\varphi}^h(x(k)),$$

$$e(k) = x(k) - W\hat{\varphi}^h(x(k)).$$

2.3.2 Навчання гібридної кластерувальної нейро-фаззі системи

Процес навчання нейро-фаззі системи зводиться до самонавчання-еволюції першого прихованого шару, навчання з учителем матриці синаптичних ваг W п'ятого прихованого шару і конкурентного самонавчання нейро-фаззі мережі Кохонена четвертого вихідного шару.

В основу налаштування першого прихованого шару покладені ідеї еволюційних фаззі-систем і, перш за все, введений адаптивний метод навчання еволюційної нейро-фаззі системи.

Нехай на вхід системи, у якій в початковому стані в першому прихованому шарі відсутні функції належності, надходить перше спостереження навчальної вибірки $x(1) = (x_1(1),...,x_i(1),...,x_n(1))^T$. Це спостереження формує перший набір функцій належності $\mu_1 = (\mu_{11},...,\mu_{1i},...,\mu_{1n})^T$ так що $c_{1i} = x_i(1)$.

Далі для векторної функції належності μ_1 з центром $c_1(1)$ задається радіус сусідства r, який визначається максимально можливим числом h функцій належності в системі. Так, якщо функції належності за компонентами розподілені рівномірно, то

$$r = \frac{2}{h-1}.$$

Далі, у випадку надходження другого спостереження $x(2)$ проводиться перевірка умови

$$\max_i \left| c_{1i} - x_i(2) \right| \le r. \qquad (2.51)$$

Якщо ця умова виконується, проводиться корекція центрів функцій належності μ_1 згідно з правилом

$$c_{1i}(2) = c_{1i}(1) + \eta(2)(x_i(2) - c_{1i}(1)),$$

де $\eta(2)$ – параметр кроку навчання, наприклад, при $\eta(2) = 0.5$

$$c_{1i}(2) = \frac{c_{1i}(1) + x_i(2)}{2}.$$

У тому випадку, якщо умова (2.51) не виконується, формується друга векторна функція належності з центрами

$$c_{2i}(2) = x_i(2).$$

Таким чином, формується вузол системи R_2, утворений елементами першого і другого шарів.

Нехай до моменту надходження на вхід системи спостереження $x(k)$ сформовано p вузлів R_l функцій належності $\mu_1, \mu_2, ..., \mu_p, p < h$ з центрами $c_{li}(k-1), l = 1, 2, ..., p; i = 1, 2, ..., n$. З приходом $x(k)$ проводиться перевірка умови

$$\max_i \left| c_{1i} - x_i(k) \right| \le r \ \forall\, l = 1, 2, ..., p. \qquad (2.52)$$

Якщо ця умова виконується, проводиться корекція центрів функцій належності, найближчих до відповідних компонентів $x_i(k)$, згідно з правилом

$$c_{li}(k) = c_{1i}(k-1) + \eta(k)(x_i(k) - c_{1i}(k-1)). \qquad (2.53)$$

Нескладно помітити, що (2.53) є не що інше, як відоме правило самонавчання Т. Кохонена «Переможець отримує все» з тією лише різницею, що самонавчання карти Кохонена реалізується на гіперсфері

$$\|x(k)\|_2 = 1,$$

а правило (2.53) – на гіперкубі

$$\|x(k)\|_\infty = 1.$$

У тому випадку, якщо умова (2.52) не виконується, в системі формується $(p+1)$-й $(p+1 \le h)$ вузол R_{p+1} з центрами функцій належності

$$c_{p+1,i}(k) = x_i(k).$$

Як видно, дана процедура є гібридом еволюційного алгоритму Н. Касабова і самоорганізовної карти Т. Кохонена, при цьому процес еволюції архітектури-самонавчання функцій належності може протікати як безперервно, так і до досягнення числом функцій належності граничного значення nh.

Для налаштування матриці синаптичних ваг може бути використаний або рекурентний метод найменших квадратів, який є по суті оптимальною за швидкодією гаусівсько-ньютонівською процедурою оптимізації виду

$$\begin{cases} W(k) = W(k-1) + \dfrac{(x(k) - W(k-1)\hat{\varphi}^h(x(k))\hat{\varphi}^{hT}(x(k))P(k-1)}{1 + \varphi^{hT}(x(k))P(k-1)\hat{\varphi}^h(x(k))}, \\ P(k) = P(k-1) - \dfrac{P(k-1)\hat{\varphi}^h(x(k))\hat{\varphi}^{hT}(x(k))P(k-1)}{1 + \hat{\varphi}^{hT}(x(k))P(k-1)\hat{\varphi}^h(x(k))}, \end{cases}$$

або багатовимірна версія алгоритму Качмажа–Уідроу–Хоффа

$$W(k) = W(k-1) + \frac{(x(k) - W(k-1)\hat{\varphi}^h(x(k))}{\left\|\hat{\varphi}^h(x(k))\right\|^2}\hat{\varphi}^{hT}(x(k)) =$$
$$= W(k-1) + (x(k) - W(k-1)\hat{\varphi}^h(x(k))\hat{\varphi}^{h+}(x(k)),$$

де $(\bullet)^+$ – операція псевдообернення.

В основі конкурентного самонавчання четвертого прихованого шару лежить імовірнісний алгоритм нечіткої кластеризації, заснований на оптимізації цільової функції виду

$$E(u_i, \tilde{c}_j^K) = \sum_{k=1}^{N}\sum_{j=1}^{m} u_j^B(k)\left\|\tilde{\varphi}(x(k)) - \tilde{c}_j^K\right\|^2$$

при обмеженнях

$$\sum_{j=1}^{m} u_j(k) = 1, k = 1,2,...,N,$$

$$0 \le \sum_{k=1}^{N} u_j(k) \le N,\ j = 1,2,...,m,$$

де $u_j(k) \in [0,1]$;

β – параметр фаззіфікації, що визначає нечітку межу між різними класами і впливає на рівень нечіткості в остаточному розбитті даних по кластерам.

Застосування стандартного апарату нелінійного програмування, заснованого на невизначених множниках Лагранжа і вирішенні системи рівнянь Каруша–Куна–Таккера, веде до результату

$$\begin{cases} \tilde{c}_j^K = \dfrac{\sum\limits_{k=1}^{N} u_j^{\beta}(k)\tilde{y}(x(k))}{\sum\limits_{k=1}^{N} u_j^{\beta}(k)}, \\ u_j(k) = \dfrac{(\left\|\tilde{\varphi}(x(k)) - \tilde{c}_j^K\right\|^2)^{\frac{1}{1-\beta}}}{\sum\limits_{l=1}^{m}(\left\|\tilde{\varphi}(x(k)) - \tilde{c}_l^K\right\|^2)^{\frac{1}{1-\beta}}}, \end{cases} \tag{2.54}$$

який при $\beta = 2$ збігається з алгоритмом нечітких С-середніх (FCM) Дж.Бездека:

$$\begin{cases} \tilde{c}_j^K = \dfrac{\sum\limits_{k=1}^{N} u_j^{2}(k)\tilde{\varphi}(x(k))}{\sum\limits_{k=1}^{N} u_j^{2}(k)}, \\ u_j(k) = \dfrac{(\left\|\tilde{\varphi}(x(k)) - \tilde{c}_j^K\right\|^2)^{-2}}{\sum\limits_{l=1}^{m}(\left\|\tilde{\varphi}(x(k)) - \tilde{c}_l^K\right\|^2)^{-2}}. \end{cases} \tag{2.55}$$

Алгоритми (2.54), (2.55) набули широкого поширення при вирішенні задач нечіткої кластеризації, проте їх використання можливе тільки в пакетному режимі при фіксованому обсязі N оброблюваної вибірки даних.

Якщо ж дані надходять на обробку послідовно в online режимі, доцільно використання рекурентного варіанту (2.54), заснованого на процедурі нелінійного програмування Ерроу–Гурвіца–Удзави виду

$$\begin{cases} \tilde{c}_j^K(k) = \tilde{c}_j^K(k-1) + \eta(k)u_j^{\beta}(k-1)(\tilde{\varphi}(x(k)) - \tilde{c}_j^K(k-1)),\ j = 1,2,...,m, \\ u_j(k) = \dfrac{(\left\|\tilde{\varphi}(x(k)) - \tilde{c}_j^K\right\|^2)^{\frac{1}{1-\beta}}}{\sum\limits_{l=1}^{m}(\left\|\tilde{\varphi}(x(k)) - \tilde{c}_l^K(k)\right\|^2)^{\frac{1}{1-\beta}}}. \end{cases} \tag{2.56}$$

Аналізуючи (2.56), можна помітити, що розглядаючи співмножник $u_j^{\beta}(k-1)$ в якості функції сусідства, приходимо до правила самонавчання Кохонена на основі принципу «Переможець отримує більше» (WTM), при $\beta \to 1$ отримуємо алгоритм типу К-середніх, а $\beta \to 0$ відповідає стандартному правилу Кохонена типу «Переможець отримує все» (WTA):

$$\tilde{c}_j^K(k) = \tilde{c}_j^K(k-1) + \eta(k)(\tilde{\varphi}(x(k)) - \tilde{c}_j^K(k-1)) \tag{2.57}$$

аналогічне (2.53). Нескладно помітити також, що рекурентная процедура (2.57) мінімізує цільову функцію виду

$$\mathrm{E}(\tilde{c}_j^K) = \sum_k \left\| \tilde{\varphi}(x(k)) - \tilde{c}_j^K \right\|^2,$$

оптимізація якої веде до звичайної оцінки середнього арифметичного

$$\tilde{c}_j^K(k) = \frac{1}{k} \sum_{\varphi(\tilde{x}(k)) \in c_{lj}} \tilde{\varphi}(x(k)),$$

або в рекуррентной формі:

$$\tilde{c}_j^K(k) = \tilde{c}_j^K(k-1) + \frac{1}{k}(\tilde{\varphi}(x(k)) - \tilde{c}_j^K(k-1)).$$

Такий вибір параметра кроку $\eta(k)$ узгоджується з правилами стохастичної апроксимації і надає результатам ясний фізичний зміст.

Таким чином, алгоритм самонавчання (2.56) четвертого прихованого шару остаточно може бути записаний у вигляді

$$\begin{cases} \tilde{c}_j^K(k) = \tilde{c}_j^K(k-1) + \dfrac{u_j^\beta(k-1)}{k}(\tilde{\varphi}(x(k)) - \tilde{c}_j^K(k-1)),\ j = 1,2,...,m, \\ u_j(k) = \dfrac{(\left\|\tilde{\varphi}(x(k)) - \tilde{c}_j^K(k)\right\|^2)^{\frac{1}{1-\beta}}}{\sum_{l=1}^{m}(\left\|\tilde{\varphi}(x(k)) - \tilde{c}_l^K(k)\right\|^2)^{\frac{1}{1-\beta}}}, \end{cases}$$

який поєднує в собі обчислювальну простоту і послідовну обробку кохоненівського навчання з можливостями нечіткої кластеризації.

Вектори-центроїди кластерів $\tilde{c}_j^K(k)$, обчислювані в просторі підвищеної розмірності R^h, далі можуть бути спроектовані в початковий простір R^n за допомогою матриці синаптичних ваг на основі перетворення

$$c_j^k(k) = W(k)c_j^k(k) \ \forall\ j = 1,2,...,m,$$

$$c_j^k(k) \in R^n,\ \tilde{c}_j^k(k) \in R^h.$$

Таким чином, розглянута тут кластерувальна нейро-фаззі система є по суті об’єднанням двох нейро-фаззі систем: еволюційної TSK-системи і нечіткої кластерувальної мережі Кохонена, що дозволяє в online режимі відновлювати класи довільної форми (лінійно нероздільні), що формуються даними, які надходять на обробку в формі потоку інформації.

2.3.3 Послідовна ядерна нечітка кластеризація великих масивів даних на основі гібридної системи обчислювального інтелекту

У задачах ядерної кластеризації кращим з позицій Big Data представляється використання підходу, заснованого на оцінках Е. Парзена, узагальнених регресійних нейронних мережах (GRNN) Д. Шпехта, заснованих на «лінивому навчанні» за принципом «нейрони в точках даних», і теоремі Т. Кавера. Подібний підхід може бути адаптований до online обробки великих

масивів інформації на основі гібридизації нейро-фаззі підходу і еволюційних систем обчислювального інтелекту, що дозволяє в процесі самонавчання «вирощувати» архітектуру, найкращим чином орієнтовану на вирішення конкретного завдання нечіткої кластеризації даних, що утворюють класи довільного виду.

Пропонована ядерна нечітка кластерувальна система містить чотири шари обробки інформації: перший прихований шар фаззіфікації вхідного простору, другий прихований шар агрегування, третій прихований шар обчислення центроїдів кластерів в просторі підвищеної розмірності і четвертий вихідний шар для розрахунку рівнів належності.

На вхідний нульовий шар системи подаються $(n\times 1)$-вимірні вектори вхідних сигналів-образів $x(k)=(x_1(k),...,x_i(k),...,x_n(k))^T$ (тут $k=1,2,...,N,...$ - поточний дискретний час), перетворені, так що $-1\le x_i(k)\le 1$, $\frac{1}{N}\sum_{k=1}^{N}x_i(k)=0$, при цьому потік даних повинен бути розбитий на m можливо пересічних кластерів довільної форми. Перший прихований шар містить $n\,h$ (по h на кожен вхід) функцій належності $\mu_{li}(x_i)$, $i=1,2,...,n;\ l=1,2,...,h$ і виконує фаззіфікацію вхідного простору, при цьому число функцій належності на кожному вході $h>n$ задається або апріорно, або визначається в процесі самонавчання системи. Другий прихований шар виконує агрегування рівнів належності, розрахованих в першому шарі, і складається з h блоків множення. Таким чином, перші два прихованих шари даної системи по архітектурі збігаються з відповідними шарами відомих нейро-фаззі систем Такагі–Сугено–Канга, Ванга–Менделя, ANFIS. Основна відмінність даної системи від зазначених полягає в тому, що навчання останніх пов’язане з налаштуванням синаптичних ваг за допомогою тих чи інших алгоритмів оптимізації, а настройка даної системи заснована на лінивому навчанні і зводиться до установки центрів функцій належності. Можна помітити, що відомі нейро-фаззі системи близькі за своєю природою до радіально-базисних нейронних мереж , а пропонована система є нечітким аналогом GRNN, тобто узагальненої регресійної нейро-фаззі мережі.

Таким чином, якщо на вхід нейро-фаззі системи поданий векторний сигнал $x(k)$, елементи першого прихованого шару обчислюють рівні належності $0 \le \mu_{li}(x_i(k)) \le 1$. При цьому в якості функцій належності найчастіше використовуються одновимірні гаусіани виду

$$\mu_{li}(x_i(k)) = \exp\left(-\frac{(x_i(k) - c_{li}(k))^2}{2\sigma_i^2} \right),$$

де $c_{li}(k)$ – параметри центру l-тої функції належності на i-му вході, що налаштовуються в процесі самонавчання;

σ_i^2 – параметр ширини.

У другому прихованому шарі обчислюються агреговані значення $\prod_{i=1}^{n} \mu_{li}(x_i(k)) = \varphi_l(k),$ при цьому для гаусіанів з однаковими параметрами ширини σ можна записати

$$\varphi_l(k) = \prod_{i=1}^{n} \mu_{li}(x_i(k)) = \prod_{i=1}^{n} \exp\left(-\frac{(x_i(k) - c_{li}(k))^2}{2\sigma_i^2} \right) = \exp\left(-\frac{\left\| (x(k) - c_l(k) \right\|^2}{2\sigma_i^2} \right),$$

де $c_l(k) = (c_{l1}(k), ..., c_{li}(k), ..., c_{ln}(k))^T$ – вектор центрів багатовимірної активаційної функції, сформованої на основі одновимірних функцій належності на кожному вході.

Таким чином, якщо на вхід системи поданий $(n \times 1)$-вимірний вектор $x(k)$, на виході другого прихованого шару з'явиться $(h \times 1)$-вимірний образ підвищеної розмірності $\varphi(k) = (\varphi_1(k), ..., \varphi_l(k), ..., \varphi_h(k))^T$. Надалі нечіткій кластеризації піддається послідовність $\varphi(1), \varphi(2), ..., \varphi(N), ...$

Третій прихований і четвертий вихідний шари даної системи утворюють двошарову нечітку кластерувальну мережу Т. Кохонена, в першому шарі якої

відновлюються центроїди кластерів $c_1^K,...,c_j^K,...,c_m^K \in R^h$, $j=1,2,...,m$, а в вихідному – рівні належності $u_j(k)$ кожного вектора $\varphi(k)$ до кожного кластеру з центроїдом c_j^K.

2.3.4 Самонавчання ядерної нечіткої кластерувальної системи

Процес самонавчання даної системи складається з налаштування функцій належності першого прихованого шару на основі лінивого навчання і методів еволюційних систем і настройки центроїдів формованих кластерів в третьому прихованому шарі.

Процес настройки функцій належності першого прихованого шару для наочності проілюструємо на прикладі двовимірного вектора входів $x(k)=(x_1(k),x_2(k))^T$, $k=1,2,...,$ при цьому кількість функцій належності на кожному вході $h=5$. Таким чином, число параметрів-центрів $c_{li}(k)$ визначається значенням $nh=10$.

У найпростішому випадку навчання покладається $c_1(0)=x(1), c_2(0)=x(2),...,c_l(0)=x(l),...,c_5(0)=x(5)$, а подальша корекція центрів $c_l(k)$ у міру надходження даних в систему проводиться відповідно до техніки, прийнятої в GRNN.

При послідовній обробці даних зручнішим представляється підхід, коли початкові положення центрів функцій належності $c_{li}(0)$ рівномірно розподілені по координатних осях x_1, x_2, при цьому відстань між ними визначається співвідношенням

$$\Delta(0)=\frac{x_{i\max}-x_{i\min}}{h-1}=\frac{2}{h-1}=0.5$$

для $-1\le x_i(k)\le 1$.

Дана ситуація ілюструється на рис. 2.6.

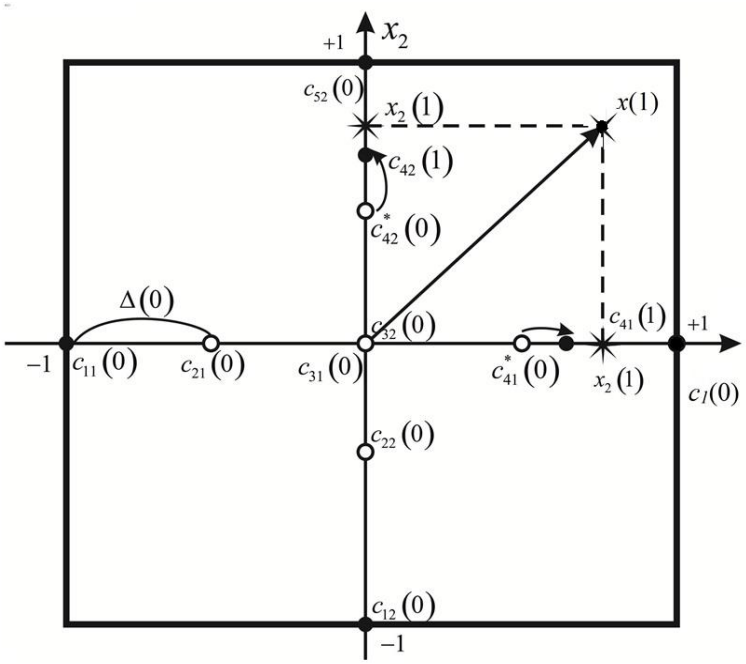


Рисунок 2.6 – Налаштування функцій належності на першому кроці самонавчання

Таким чином, в разі багатовимірного вектора входів $x(k) \in R^n$, центри $c_{li}(0)$ рівномірно розподіляються по осях гіперкуба $[-1,\ 1]^n$.

Нехай на вхід системи подано перше спостереження $x(1) = (x_1(1), x_2(1))^T$, позначене на рис. 2.6 символом $*$. Далі на кожній з осей знаходяться центри-«переможці» (аналоги нейронів-«переможців» у SOM) $c^*_{li}(0)$, найближчі до $x_i(1)$ в сенсі відстані

$$d_{li} = |x_i(1) - c_{li}(0)|,$$

тобто

$$c^*_{li}(0) = \arg\min\{d_{1i}, d_{2i}, ..., d_{hi}\}.$$

Пошук центру-«переможця» є по суті реалізацією процесу конкуренції за Т. Кохоненом з тією лише різницею, що «переможці» по різним осях можуть належати функціям належності з різними індексами l. На рис. 2.6 такими «переможцями» є центри $c_{41}^{*}(0)$ і $c_{42}^{*}(0)$.

Далі ці «переможці» підтягуються до компонентів вхідного сигналу $x_i(1)$ згідно з WTA-правилом самонавчання, яке для ситуації, наведеної на рис. 2.6, може бути записано у вигляді

$$c_{li}(1)=\begin{cases} c_{li}^{*}+\eta_{li}(k)(x_i(1)-c_{li}^{*}(0)) - \textit{для переможця } (l=4), \\ c_{li}(0) - \textit{для всіх інших } l=1,2,3,5. \end{cases} \tag{2.58}$$

для довільних n, h і k WTA-правило самонавчання першого прихованого шару (2.58) має форму

$$c_{li}(k)=\begin{cases} c_{li}^{*}(k-1)+\eta_{li}(k)(x_i(k)-c_{li}^{*}(k-1)) - \textit{для переможця} \\ \qquad \textit{при } l=1,2,...,h;\ i=1,2,...,n; \\ c_{li}(k-1) - \textit{для всіх інших}, \end{cases} \tag{2.59}$$

де $\eta_{li}(k)$ – параметр кроку навчання, що монотонно зменшується в процесі настройки.

В разі

$$\eta_{li}(k)=k_{li}^{-1}$$

(тут k_{li} – число «перемог» центру $c_{li}(k)$), правило (2.59) описує послідовний online процес кластеризації, відповідний класичному методу К-середніх.

Як вже зазначалося вище, послідовність $(h\times 1)$-вимірних векторів $\varphi(1),\varphi(2),...,\varphi(N),...$ піддається нечіткій кластеризації в третьому і четвертому шарах даної системи.

У класі процедур нечіткої кластеризації найбільш суворими з математичної точки зору є алгоритми, засновані на цільових функціях і вирішальні завдання їх оптимізації при тих чи інших апріорних припущеннях. Найбільш поширеним тут є імовірнісний підхід, заснований на мінімізації критерію (цільової функції)

$$E(u_j(k), c_j^K) = \sum_{k=1}^{N}\sum_{j=1}^{m} u_j^{\beta}(k)\left\|\varphi(k) - c_j^K\right\|^2 \tag{2.60}$$

при обмеженнях

$$\sum_{j=1}^{m} u_j(k) = 1, \tag{2.61}$$

$$0 \le \sum_{k=1}^{N} u_j(k) \le N, \tag{2.62}$$

де β – невід'ємний параметр фаззіфікації, що визначає «розмитість» меж між кластерами.

Вводячи функцію Лагранжа

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

(тут $\lambda(k)$ – невизначені множники Лагранжа) і вирішуючи систему рівнянь Каруша–Куна–Таккера, нескладно отримати остаточний результат у вигляді

$$\begin{cases} u_j(k) = \dfrac{(\left\|\varphi(k) - c_j^K\right\|^2)^{\frac{1}{1-\beta}}}{\sum\limits_{l=1}^{m}(\left\|\varphi(k) - c_l^K\right\|^2)^{\frac{1}{1-\beta}}}, \\ c_j^K = \dfrac{\sum\limits_{k=1}^{N} u_j^\beta(k)\varphi(k)}{\sum\limits_{k=1}^{N} u_j^\beta(k)}, \\ \lambda(k) = -((\sum\limits_{l=1}^{m}\beta\left\|\varphi(k) - c_l^K\right\|^2)^{\frac{1}{1-\beta}})^{1-\beta}, \end{cases} \tag{2.63}$$

що збігається при $\beta = 1$ з FCM Дж. Бездека, а при $\beta \to 1$ близький до чіткого методу К-середніх.

У ситуаціях, коли дані надходять на обробку послідовно в online режимі, може бути використана рекурентна версія

$$\begin{cases} u_j(k) = \dfrac{(\left\|\varphi(k) - c_j^K(k-1)\right\|^2)^{\frac{1}{1-\beta}}}{\sum\limits_{l=1}^{m}(\left\|\varphi(k) - c_l^K\right\|^2)^{\frac{1}{1-\beta}}}, \\ c_j^K(k) = c_j^K(k-1) + \eta(k)u_j^\beta(k)(\varphi(k) - c_j^K(k-1)), \end{cases} \tag{2.64}$$

при цьому нескладно помітити, що друге співвідношення (2.64) є по суті WTM-правилом самонавчання Т. Кохонена, де співмножник $u_j^\beta(k)$ відіграє роль функції сусідства.

Основний недолік імовірнісного підходу пов’язаний з обмеженням (2.62), що вимагає рівності одиниці суми належності кожного спостереження всіх векторів. Цього недоліку позбавлений метод можливісної нечіткої кластеризації.

У можливісному методі цільова функція має вигляд

$$E(u_j(k), c_j^K) = \sum_{k=1}^{N}\sum_{j=1}^{m} u_j^{\beta}(k)\left\|\varphi(k) - c_j^K\right\|^2 + \sum_{j=1}^{m}\mu_j\sum_{k=1}^{N}(1-u_j(k))^{\beta}, \qquad (2.65)$$

де скалярний параметр $\mu_j > 0$ визначає відстань, на якому рівень належності приймає значення 0,5, тобто якщо

$$\left\|\varphi(k) - c_j^K\right\|^2 = \mu_j,$$

то

$$u_j(k) = 0.5.$$

Пряма мінімізація (2.65) по $u_j(k)$ і c_j^K веде до рішення

$$\begin{cases} u_j(k) = \left(1 + \left(\dfrac{\left\|\varphi(k) - c_j^K\right\|^2}{\mu_j}\right)^{\frac{1}{1-\beta}}\right)^{-1}, \\ c_j^K = \dfrac{\sum\limits_{k=1}^{N} u_j^{\beta}(k)\varphi(k)}{\sum\limits_{k=1}^{N} u_j^{\beta}(k)}, \\ \mu_j = \dfrac{\sum\limits_{k=1}^{N} u_j^{\beta}(k)\left\|\varphi(k) - c_j^K\right\|^2}{\sum\limits_{k=1}^{N} u_j^{\beta}(k)}, \end{cases} \qquad (2.66)$$

яке у рекурентному варіанті має вигляд:

$$\begin{cases} u_j(k) = \dfrac{1}{1 + \left(\dfrac{\left\| \varphi(k) - c_j^K(k-1) \right\|^2}{\mu_j(k)} \right)^{\frac{1}{\beta-1}}}, \\ c_j^K(k) = c_j^K(k-1) + \eta(k) u_j^\beta(k)(\varphi(k) - c_j^K(k-1)), \\ \mu_j(k) = \left(\sum\limits_{p=1}^{k} u_j^\beta(p) \right)^{-1} \left(\sum\limits_{p=1}^{K} u_j^\beta(p) \left\| \varphi(p) - c_j^K(k) \right\|^2 \right), \end{cases} \tag{2.67}$$

де друге співвідношення є WTM-правило самонавчання з функцією сусідства, яка визначається першим співвідношенням процедури.

Необхідно відзначити, що використання цих процедур в задачах кластеризації даних високої розмірності (а саме такі дані обробляються розглянутою системою) може не привести до бажаного результату через, так званий, ефект «концентрації норми» (CoN) [139], [140]. Для подолання цього негативного ефекту Ф. Клавонном і Ф. Хьоппнером в задачах імовірнісної нечіткої кластеризації було запропоновано замість цільової функції (2.60) вжити співвідношення

$$E(u_j(k), c_j^K) = \sum_{k=1}^{N} \sum_{j=1}^{m} (\alpha u_j^2(k) + (1-\alpha) u_j(k)) \left\| \varphi(k) - c_j^K \right\|^2 \tag{2.68}$$

з обмеженнями (2.61), (2.62), де $0 < \alpha \le 1$ – параметр, що налаштовується, який визначає характер одержуваного рішення.

Вводячи функцію Лагранжа

$$L(u_j(k), c_j^K, \lambda(k)) = \sum_{k=1}^{N} \sum_{j=1}^{m} (\alpha u_j^2(k) + (1-\alpha) u_j(k)) \left\| \varphi(k) - c_j^K \right\|^2 + \\ + \sum_{k=1}^{N} \lambda(k) \left(\sum_{j=1}^{m} u_j(k) - 1 \right) \tag{2.69}$$

і вирішуючи систему рівнянь Каруша–Куна–Таккера, приходимо до вирішення

$$\begin{cases} u_j(k) = \dfrac{\alpha - 1}{2\alpha} + \dfrac{1 + m\dfrac{1-\alpha}{2\alpha}}{\sum\limits_{l=1}^{m} \dfrac{\left\| \varphi(k) - c_j^K \right\|^2}{\left\| \varphi(k) - c_l^K \right\|^2}}, \\ c_j^K = \dfrac{\sum\limits_{k=1}^{N} (\alpha u_j^2(k) + (1-\alpha) u_j(k)) \varphi(k)}{\sum\limits_{k=1}^{N} (\alpha u_j^2(k) + (1-\alpha) u_j(k))}, \\ \lambda(k) = -\dfrac{1 + m\dfrac{1-\alpha}{2\alpha}}{\sum\limits_{l=1}^{m} \left(2\alpha \left\| \varphi(k) - c_l^K \right\| \right)^{-1}}. \end{cases}$$

Нескладно помітити, що при $\alpha = 1$ приходимо до стандартної FCM-процедурі.

Застосовуючи далі до лагранжіану (2.69) процедуру нелінійного програмування Ерроу–Гурвіца–Удзави, приходимо до рекурентного алгоритму нечіткої кластеризації за критерієм (2.68):

$$\begin{cases} u_j(k) = \dfrac{\alpha - 1}{2\alpha} + \dfrac{1 + m\dfrac{1-\alpha}{2\alpha}}{\sum\limits_{l=1}^{m} \dfrac{\left\| \varphi(k) - c_j^K \right\|^2}{\left\| \varphi(k) - c_l^K \right\|^2}}, \\ c_j^K(k) = c_j^K(k-1) + \eta(k)(\alpha u_j^2(k) + (1-\alpha) u_j(k))(\varphi(k) - c_j^K(k-1)), \end{cases} \tag{2.70}$$

при цьому нескладно помітити, що (2.64) і (2.70) збігаються при $\beta = 2$, $\alpha = 1$ і являють собою рекурентну версію FCM.

Підхід до нечіткої кластеризації, заснований на критерії (2.69), може бути поширений на можливісні процедури шляхом використання цільової функції виду

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

мінімізація якої веде до результату:

$$\begin{cases} u_j(k)=\dfrac{\left(\alpha\mu_j+(1-\alpha)\left\|\varphi(k)-c_j^K\right\|^2\right)+\mu_j}{2\alpha\left(\left\|\varphi(k)-c_j^K\right\|^2+\mu_j\right)}, \\ c_j^K=\dfrac{\sum\limits_{k=1}^{N}\left(\alpha u_j^2(k)+(1-\alpha)u_j(k)\right)\varphi(k)}{\sum\limits_{k=1}^{N}\left(\alpha u_j^2(k)+(1-\alpha)u_j(k)\right)}, \\ \mu_j=\dfrac{\sum\limits_{k=1}^{N}\left(\alpha u_j^2(k)+(1-\alpha)u_j(k)\right)\left\|\varphi(k)-c_j^K\right\|^2}{\sum\limits_{k=1}^{N}\left(\alpha u_j^2(k)+(1-\alpha)u_j(k)\right)}, \end{cases}$$

а в рекурентному варіанті:

$$\begin{cases} u_j(k)=\dfrac{\left(\alpha\mu_j(k)+(1-\alpha)\left\|\varphi(k)-c_j^K(k-1)\right\|^2\right)+\mu_j(k)}{2\alpha\left(\left\|\varphi(k)-c_j^K(k-1)\right\|^2+\mu_j(k)\right)}, \\ c_j^K(k)=c_j^K(k-1)+\eta(k)\left(\alpha u_j^2(k)+(1-\alpha)u_j(k)\right)\left(\varphi(k)-c_j^K(k-1)\right), \\ \mu_j=\dfrac{\sum\limits_{p=1}^{N}\left(\alpha u_j^2(p)+(1-\alpha)u_j(p)\right)\left\|\varphi(p)-c_j^K(k-1)\right\|^2}{\sum\limits_{p=1}^{k}\left(\alpha u_j^2(p)+(1-\alpha)u_j(p)\right)}. \end{cases} \qquad (2.71)$$

Зауважимо, що у (2.69) і (2.70) алгоритми налаштування центроїдів кластерів збігаються, будучи по суті WTM-правилом самонавчання, а відмінність полягає у визначенні рівнів належності до конкретних кластерів.

Таким чином, пропонована ядерна гібридна нейро-фаззі система реалізує по суті двоетапну процедуру нечіткої кластеризації, де в перших двох шарах проводиться грубе налаштування центрів функцій належності на основі звичайної SOM, навченою за допомогою WTA-правила, а в вихідних шарах – точне налаштування на основі нечіткої online кластеризації даних підвищеної розмірності з використанням модифікованого WTM-правила.

3 РОЗРОБКА МЕТОДІВ ОБРОБКИ ПОТОКІВ НЕЧІТКОЇ ІНФОРМАЦІЇ ЗА УМОВ НЕСТАЦІОНАРНОСТІ ТА НЕВИЗНАЧЕНОСТІ ЩОДО КІЛЬКОСТІ ТА ФОРМИ КЛАСТЕРІВ

3.1 Каскадна нейронна мережа, що еволюціонує, для послідовного нечіткого кластерування потоків даних

У цьому розділі описані архітектура та методи навчання каскадної нейро-мережі для нечіткого кластерування, зокрема потоків даних; проведено аналіз існуючих систем, що еволюціонують, для кластерування даних, зокрема нечіткого, і розглянуті особливості та труднощі послідовного кластерування, та описані два підходи, переваги яких поєднує у собі пропонована система: нечітке та ієрархічне кластерування.

Завдання кластерування (класифікування без вчителя) досить часто зустрічається в багатьох додатках, пов’язаних з видобутком знань, де у режимі самонавчання необхідно розбити деякий вхідний нерозмічений масив даних на однорідні в прийнятому сенсі групи. Розглянемо деякі ієрархічні та розподільні методи кластерування, адже, як буде показано далі, пропонована у цьому розділі самонавчанна система поєднує у собі переваги обох підходів.

Розподільні методи кластерування (чи то жорсткі, чи нечіткі) можна назвати динамічними у тому сенсі, що належність певного образу до певного кластеру (кластерів для нечіткої модифікації) не є постійною. Нездатність методів розподільного кластерування самостійно визначити кількість кластерів у певному сенсі компенсується тим, що знання форми чи розміру кластерів може стати у нагоді на етапі вибору відповідних прототипів та насамперед типу відстані (міри схожості) і суттєво поліпшити кінцеве розбиття вибірки. Але, варто зазначити чутливість таких методів до початкової ініціалізації, шуму і викидів, їх сприйнятливість до локальних мінімумів, адже вони ґрунтуються на оптимізації певної цільової функції. Типові методи розподільного кластерування мають обчислювальну складність $\mathcal{O}(N)$ для тренувальної вибірки розміру N.

Серед методів ієрархічного кластерування виділяють два основних типи: висхідні та спадні методи. Спадні методи працюють за принципом «зверху-вниз»: на початку припускається, що всі образи належать до одного кластеру, який потім розбивається на все більш дрібні кластери. Більш поширеними є висхідні алгоритми, які на початку роботи поміщають кожен об'єкт до окремого кластеру, а потім об'єднують кластери у все більш крупні, доки усі образи не матимуть свій власний кластер. Таким чином будується система вкладених розбиттів. Результати таких алгоритмів зазвичай представляють у вигляді дерева – дендрограми (тут можна провести аналогію між висхідними та спадними методами ієрархічного кластерування та конструктивними і деструктивними системами, що еволюціонують. У цій роботі здебільшого розглядається конструктивний підхід, тому пропонована самонавчанна система є у певному сенсі альтернативою системам висхідного ієрархічного кластерування, що здатна працювати в онлайн режимі).

Для обчислення відстаней між кластерами використовуються такі відстані:

– одинарний зв’язок (відстань найближчого сусіда): відстань між двома кластерами визначається відстанню між двома найбільш близькими об'єктами (найближчими сусідами) у різних кластерах. Результуючі кластери мають тенденцію об’єднуватися в ланцюжки;

– повний зв’язок (відстань найбільш віддалених сусідів): відстані між кластерами визначаються найбільшою відстанню між будь-якими двома об'єктами різних кластерів (тобто найбільш віддаленими сусідами). Цей метод зазвичай працює дуже добре, коли об'єкти походять з окремих груп. Якщо ж кластери мають видовжену форму або їх природний тип є «ланцюжковим» цей метод непридатний;

– незважене попарне середнє: відстань між двома різними кластерами обчислюється як середня відстань між усіма парами об’єктів у них. Метод ефективний, коли об’єкти формують різні групи, проте він працює однаково добре і у випадках протяжних («ланцюжкового» типу) кластерів;

– зважене попарне середнє: метод ідентичний методу незваженого попарного середнього, за винятком того, що при обчисленнях розмір відповідних кластерів (тобто число об'єктів, що містяться в них) використовується у якості вагового коефіцієнту. Тому доцільно використовувати даний метод у випадку нерівних за розміром кластерів;

– незважений центроїдний метод: у цьому методі відстань між двома кластерами визначається як відстань між їх центрами тяжкості;

– зважений центроїдний метод (медіана): цей метод ідентичний попередньому, за винятком того, що при обчисленнях використовуються ваги для обліку різниці між розмірами кластерів. Тому, якщо є або підозрюються значні відмінності в розмірах кластерів, цей метод має перевагу над попереднім.

Порівняно з розподільним кластеруванням, методи ієрархічного кластерування легко ідентифікують викиди, не потребують визначеної кількості кластерів та нечутливі до початкової ініціалізації чи локальних мінімумів. До недоліків варто віднести нездатність методів визначати кластери, що перекривають один одного. Крім того, ієрархічне кластерування є статичним, тобто образи віднесені до певного кластеру на ранніх стадіях не можуть бути пізніше належати іншому, що унеможливлює створення модифікацій методів для послідовного кластерування, на відміну від розподільного кластерування. Методи ієрархічного кластерування здебільшого мають обчислювальну складність принаймні $\mathcal{O}(N^2)$, що робить їх використання недоцільним для великих масивів даних.

Традиційний підхід до завдання кластерування припускає, що кожне спостереження належить лише одному кластерові, в той час як більш природною видається ситуація, коли кожен вектор-спостереження оброблюваної вибірки можна віднести відразу декільком класам з різними рівнями належності. Така ситуація є предметом розгляду нечіткого кластерного аналізу, а для його вирішення широко використовується апарат обчислювального інтелекту і, насамперед, нейро-фаззі підхід. При цьому більшість алгоритмів нечіткої кластеризації призначені для роботи в пакетному режимі, коли усі дані, що

підлягають обробці, задані апріорно. Вихідною інформацією для такої задачі є вибірка спостережень, сформована з m-вимірних векторів ознак $x(1), x(2), \dots, x(N)$, при цьому для зручності чисельної реалізації вихідні дані попередньо деяким чином перетворюються, наприклад, так, щоб всі спостереження належали до гіперкубу $[-1,1]^n$ або одиничній гіперсфері $\|x(k)\|^2$.

Результатом такого кластерування є розбиття масиву вихідних даних на M кластерів з певним рівнем належності $u_J(k)$ k-ого вхідного образу $x(k)$ до J-ого кластеру $(J = 1,2, \dots, M)$. Передбачається, що N та M, а також параметри кластерування (в першу чергу, фаззіфікатор) задані апріорі і не змінюються під час обробки даних. Варто зауважити, що існує широкий клас задач динамічного інтелектуального аналізу даних і потоків даних (Dynamic Data Mining, Data Stream Mining) у випадку, коли дані надходять у вигляді послідовного потоку в онлайн режимі. Отже, кількість вхідних образів N у цьому випадку не обмежується, а k набуває значення поточного дискретного часу.

Самоорганізовні мапи Кохонена добре пристосовані для вирішення завдання кластерування в онлайн режимі. Ці нейронні мережі мають один шар латеральних з'єднань та навчаються за принципами «переможець отримує все» або «переможець отримує більше».

Самоорганізовні мапи також відомі своєю ефективністю вирішення задачі кластерування класів, що перетинаються. Тому, у зв'язку з дедалі більшою кількістю завдань кластерування потоків даних, з'явилися самонавчанні нейро-фазі гібридні системи, що у деякому сенсі поєднують у собі самоорганізовні мапи Кохонена (SOM) та метод нечітких c-середніх Бездека. Такі гібридні системи володіють обширною функціональністю завдяки використанню спеціальних методів налаштування, що ґрунтуються на процедурах оптимізації прийнятої цільової функції, але потребують попередньо заданої кількості кластерів та фіксованого значення фаззіфікатора.

3.2 Критерії дійсності нечіткого кластерування

Оскільки коефіцієнт розбиття залежить лише від значень функції належності, йому властиві деякі недоліки. Коли фаззіфікатор наближається до 1, індекс дійсності буде однаковим для усіх c, коли фаззіфікатор наближається до ∞.

Індекс розбиття ентропії PE – важливий критерій дійсності нечіткого кластерування, що залежить лише від значень функції належності

$$PE = -\frac{1}{N}\sum_{l=1}^{M}\sum_{i=1}^{N} u_{li}\, log_a(u_{li}). \tag{3.1}$$

Індекс ентропії розбиття набуває значень у інтервалі $[0, log_a M]$. Що ближче значення PE до 0, то жорсткіше розбиття вхідних даних. Значення PE близькі до верхньої межі вказують на відсутність будь-якої структури, притаманної набору вхідних даних, або на нездатність методу її виявити. Індекс ентропії розбиття має ті самі недоліки, що і коефіцієнт розбиття. Оптимальній кількості кластерів M^* відповідає мінімальне значення (3.1).

Фукуяма та Сугено запропонували індекс дійсності нечіткого кластерування, залежний як від рівнів належності так і від самих вхідних даних:

$$FS = \sum_{i=1}^{N}\sum_{l=1}^{M} u_{li}^{\beta}(\|x_i - z_l\|^2 - \|z_l - z\|^2) \tag{3.2}$$

де z та z_l – середнє арифметичне усієї виборки та образів віднесених до кластеру M_l відповідно.

З визначення (3.2**Помилка! Джерело посилання не знайдено.**) видно, що малі значення FS говорять про компактні добре визначені кластери.

Нечітка множина i-ого образу визначається як

$$\tilde{A}_l = \sum_{i=1}^{N} \frac{u_{li}}{x_i}, l = 1,2, \dots, M. \tag{3.3}$$

Ступінь, в якій $\widetilde{A}_l$ є підмножиною $\widetilde{A}_p$ визначається таким чином

$$\begin{cases} S(\tilde{A}_l, \tilde{A}_p) = \dfrac{S(\tilde{A}_l, \tilde{A}_p) + S(\tilde{A}_p, \tilde{A}_l)}{2}, \\ U(\tilde{A}_j) = \sum_{i=1}^{N} U_{ji}. \end{cases} \tag{3.4}$$

Зважаючи на (3.4**Помилка! Джерело посилання не знайдено.**), можна запропонувати такі варіанти обчислення міри подібності:

$$N_1(\tilde{A}_l, \tilde{A}_p) = \frac{S(\tilde{A}_l, \tilde{A}_p) + S(\tilde{A}_p, \tilde{A}_l)}{2}, \tag{3.5a}$$

$$N_2(\tilde{A}_l, \tilde{A}_p) = min\left(S(\tilde{A}_l, \tilde{A}_p), S(\tilde{A}_p, \tilde{A}_l)\right), \tag{3.5b}$$

$$N_3(\tilde{A}_l, \tilde{A}_p) = S(\tilde{A}_l \cup \tilde{A}_p, \tilde{A}_p \cap \tilde{A}_l). \tag{3.5c}$$

Тоді індекс дійсності кластерування, що ґрунтується на нечіткій подібності, можна визначити як

$$FSim = \max_{1 \le l \le M} \min_{\substack{1 \le p \le M, \\ p \ne l}} N(\tilde{A}_l, \tilde{A}_p), \tag{3.6}$$

де міру нечіткої подібності $N(\tilde{A}_l, \tilde{A}_p)$ можна знайти за будь-яким виразом (3.5**Помилка! Джерело посилання не знайдено.**).

3.3 Архітектура самонавчанної каскадної мережі, що еволюціонує, для нечіткого кластерування

До нульового шару пропонованої системи послідовно передаються дані у формі векторного сигналу $x(k) = (x_1(k), x_2(k), \dots)^T$. Вхідні сигнали надходять до всіх вузлів системи $N_j^{[m]}$, де $j = 1,2, \dots, q$ – кількість вузлів у пулі-ансамблі, $m = 1,2, \dots$ – номер каскаду. Вузол кожного каскаду призначений для онлайн кластерування потоку даних і відрізняється від вузлів-сусідів методом навчання або, у випадку спільного методу кластерування, параметрами алгоритму. Кількість кластерів для кожного каскаду є відомою і дорівнює $m + 1$. Елемент $PC_j^{[m]}$ дає оцінку якості кластерування кожного вузла у пулі, а елемент $PC^{*[m]}$ визначає найкращий елемент у пулі кожного каскаду; елемент системи $XB^{[m]}$ оцінює загальну якість кластеризації пулу, враховуючи прийняту кількість кластерів $m + 1$.

Таким чином, система розв’язує задачу кластерування нестаціонарного потоку даних в умовах невизначеності щодо кількості кластерів, а також їх вигляду і рівню взаємного перекриття.

І, нарешті, вихідний вузол системи XB^*, порівнюючи якість кластерування кожного з каскадів, виділяє найкращий результат – кількість кластерів, їх центроїди-прототипи та рівні належності кожного спостереження до кожного з сформованих центроїдів.

Незважаючи на удавану громіздкість, чисельна реалізація запропонованої архітектури не викликає принципових труднощів завдяки тому, що потік даних, який надходить до системи, може оброблятися у паралельному режимі вузлами системи $N_j^{[m]}$.

3.4 Адаптивне навчання вузлів каскадної нейро-фаззі системи, що еволюціонує

В основі методів навчання вузлів системи лежать алгоритми нечіткого кластерування, засновані на цільових функціях, такі, що вирішують задачу їх оптимізації за деяких апріорних припущень.

Найбільш поширеним є ймовірнісний підхід, заснований на мінімізації цільової функції

$$E\left(u_{jl}^{[m]}(k), c_{jl}^{[m]}\right) = \sum_{k+1}^{N} \sum_{l+1}^{m+1} \left(u_{jl}^{[m]}(k)\right)^{\beta} \left\| x(k) - c_{jl}^{[m]} \right\|^2 \qquad (3.7)$$

при обмеженнях

$$\sum_{l=1}^{m+1} u_{jl}^{[m]}(k) = 1, \quad 0 \le \sum_{k=1}^{N} u_{jl}^{[m]}(k) \le N, \qquad (3.8)$$

де $u_{ij}^{[m]}(k) \in [0,1]$ – рівень належності спостереження $x(k)$ до l-ого кластеру у j-ому вузлі каскаду m;

$c_{jl}^{[m]}$ – $(n \times 1)$-вимірній вектор-центроїд l-ого кластеру у j-ому вузлі каскаду m;

$\beta > 1$ – параметр фаззіфікації (фаззіфікатор), що визначає розмитість границь між кластерами;

N – кількість образів у вхідній виборці, що, у рамках класичного підходу Бездека, вважається незмінною та такою, що задана апріорі.

Вводячи функцію Лагранжа

$$\Lambda\left(u_{jl}^{[m]}(k), c_{jl}^{[m]}, \lambda^{[m]}(k)\right) \qquad (3.9)$$

$$= \sum_{k=1}^{N} \sum_{l=1}^{m+1} \left(u_{jl}^{[m]}(k)\right)^{\beta} \left\| x(k) - c_{jl}^{[m]} \right\|^2 + \sum_{k=1}^{N} \lambda_j^{[m]}(k) \left(\sum_{l=1}^{m+1} u_{jl}^{[m]}(k) - 1 \right),$$

де $\lambda_j^{[m]}(k)$ – невизначений Лагранжів множник, та розв'язавши систему рівнянь Каруша–Куна–Таккера, нескладно отримати шукане рішення у вигляді

$$\begin{cases} u_{jl}^{[m]}(k) = \dfrac{\left(\left\| x(k) - c_{jl}^{[m]} \right\|^2 \right)^{\frac{1}{1-\beta}}}{\sum_{l=1}^{m+1} \left(\left\| x(k) - c_{jl}^{[m]} \right\|^2 \right)^{\frac{1}{1-\beta}}}, \\ c_{jl}^{[m]} = \dfrac{\sum_{k=1}^{N} \left(u_{jl}^{[m]}(k) \right)^{\beta} x(k)}{\sum_{k=1}^{N} \left(u_{jl}^{[m]}(k) \right)^{\beta}}, \\ \lambda_j^{[m]}(k) = -\left(\left(\sum_{l=1}^{m+1} \beta \left\| x(k) - c_{jl}^{[m]} \right\|^2 \right)^{\frac{1}{1-\beta}} \right)^{\frac{1}{1-\beta}}, \end{cases} \tag{3.10}$$

що при $\beta = 2$ збігається з алгоритмом нечітких c-середніх Бездека (FCM) і приймає форму

$$\begin{cases} u_{jl}^{[m]}(k) = \dfrac{\left(\left\|x(k) - c_{jl}^{[m]}\right\|^2\right)^{-2}}{\sum_{l=1}^{m+1}\left(\left\|x(k) - c_{jl}^{[m]}\right\|^2\right)^{-2}}, \\ c_{jl}^{[m]} = \dfrac{\sum_{k=1}^{N}\left(u_{jl}^{[m]}(k)\right)^2 x(k)}{\sum_{k=1}^{N}\left(u_{jl}^{[m]}(k)\right)^2}. \end{cases} \quad (3.11)$$

Тут варто відзначити, що вибір фаззіфікатору $\beta = 2$ в (3.11) не дає жодних переваг порівняно з довільним значенням β у (3.10), у зв'язку з чим пропонується використовувати різні значення параметра фаззіфікації для кожного вузла пулу каскаду, після чого вибирати найкращий результат залежно від прийнятого критерію якості нечіткого кластерування.

Для послідовної обробки потоку даних, що надходять в онлайн режимі, нами були запропоновані рекурентні алгоритми, в основі яких лежить процедура нелінійного програмування Ерроу–Гурвіца–Удзави. Так, пакетному алгоритму (3.10**Помилка! Джерело посилання не знайдено.**) відповідає вираз

$$\begin{cases} u_{jl}^{[m]}(k+1) = \dfrac{\left\|x(k+1) - c_{jl}^{[m]}(k)\right\|^{\frac{1}{1-\beta}}}{\sum_{l=1}^{m+1}\left\|x(k+1) - c_{jl}^{[m]}(k)\right\|^{\frac{1}{1-\beta}}} \\ c_{jl}^{[m]}(k+1) = c_{jl}^{[m]}(k) + \eta(k+1)\left(u_{jl}^{[m]}(k+1)\right)^{\beta_j}\left(x(k+1) - c_{jl}^{[m]}(k)\right), \end{cases} \quad (3.12)$$

(тут $\eta(k+1)$ – параметр кроку навчання), що є узагальненням алгоритму навчання Чанга–Лі і при $\beta = 2$ близьке до ґрадієнтної процедури Парка-Деґера:

$$\begin{cases} u_{jl}^{[m]}(k+1) = \dfrac{\left\| x(k+1) - c_{jl}^{[m]}(k) \right\|^{-2}}{\sum_{l=1}^{m+1} \left\| x(k+1) - c_{jl}^{[m]}(k) \right\|^{-2}}, \\ c_{jl}^{[m]}(k+1) = c_{jl}^{[m]}(k) + \eta(k+1)\left(u_{jl}^{[m]}(k+1)\right)^{-2}\left(x(k+1) - c_{jl}^{[m]}(k)\right). \end{cases} \quad (3.13)$$

Варто зауважити, що, розглянувши співвідношення (**Помилка! Джерело посилання не знайдено.**) з позицій навчання Кохоненової самоорганізовної мапи (SOM), можна помітити, що множник $\left(u_{jl}^{[m]}\right)^{\beta_i}$ відповідає функції сусідства в правилі навчання на основі принципу «переможцю дістається більше», маючи при цьому дзвонуватий вигляд.

Вочевидь, у випадку, коли $\beta = 1$ та $u_{jl}^{[m]}(k) \in [0,1]$, процедура (3.12**Помилка! Джерело посилання не знайдено.**) збігається з чітким алгоритмом c-середніх (HCM), коли ж $\beta = 0$, маємо стандартне правило навчання Кохонена «переможцю дістається все»

$$c_{jl}^{[m]}(k+1) = c_{jl}^{[m]}(k) + \eta(k+1)\left(x(k+1) - c_{jl}^{[m]}(k)\right). \quad (3.14)$$

Легко побачити, що процедура (3.14**Помилка! Джерело посилання не знайдено.**) оптимізує цільову функцію

$$E\left(c_{jl}^{[m]}\right) = \sum_{k=1}^{N} \left\| x(k) - c_{jl}^{[m]} \right\|^2, \qquad \sum_{l=1}^{m+1} N_l = N, \quad (3.15)$$

мінімум якої збігається із середнім арифметичним

$$c_{jl}^{[m]} = \frac{1}{N}\sum_{k=1}^{N_l} x(k), \quad (3.16)$$

де N_l – кількість векторів, віднесених до l-го кластеру у процесі конкуренції.

Якщо записати (3.16) у рекурентній формі, отримаємо оптимальний алгоритм самонавчання Ципкіна

$$c_{jl}^{[m]}(k+1) = c_{jl}^{[m]}(k) + \frac{1}{N_l(k+1)}\Big(x(k+1) - c_{jl}^{[m]}(k)\Big), \qquad (3.17)$$

де $N_l(k+1)$ – число векторів, віднесених до l-го кластеру в $(k+1)$-й момент реального часу, що є стандартною процедурою стохастичної апроксимації.

У загальному випадку метод навчання (3.12) вузла можна розглядати як правило самонавчання нечіткої модифікації самоорганізовної мапи Кохонена. Тут $N_{jl}^{[m]K}$ – стандартні нейрони Кохонена, пов’язані між собою латеральними зв'язками, що налаштовуються згідно «переможцю дістається більше» правилу навчання на основі другого співвідношення (3.12**Помилка! Джерело посилання не знайдено.**). Вузли $N_{jl}^{[m]u}$ обчислюють рівні належності згідно першому співвідношенню (3.12**Помилка! Джерело посилання не знайдено.**). Вузли $N_{jl}^{[m]}$ кожного з каскадів відрізняються тільки фаззіфікатором алгоритму самонавчання, а вузол кожного наступного каскаду містить додатково один нейрон Кохонена і один елемент для розрахунку рівнів належності.

3.5 Керування каскадами самонавчанної нейро-фаззі системи, що еволюціонує

Якість кластерування кожного вузла системи може бути оцінена за допомогою будь-якого з індексів, що використовується у задачах нечіткого кластерування. Одним за найпростіших та разом з тим найефективніших індексів є так званий «коефіцієнт розбиття», який, власне, є середнім квадратів рівнів належності всіх спостережень до кожного кластеру і має вигляд

$$PC_j^{[m]} = \frac{1}{N}\sum_{k=1}^{N}\sum_{l=1}^{m+1}\left(u_{jl}^{[m]}(k)\right)^2 \tag{3.18}$$

Цей коефіцієнт має ясний фізичний зміст: що краще виражені кластери, то більше значення $PC_j^{[m]}$ (верхня межа – $PC_j^{[m]} = 1$), а його мінімум $PC_j^{[m]} = (m+1)^{-1}$ досягається, якщо дані належать усім кластерам рівномірно, що, вочевидь, є тривіальним рішенням. Для розглянутої нами системи цей коефіцієнт зручний тим, що його легко розрахувати в онлайн режимі

$$PC_j^{[m]}(k+1) = PC_j^{[m]}(k) + \frac{1}{k+1}\left(\sum_{l=1}^{m+1}\left(u_{jl}^{[m]}(k+1)\right)^2 - PC_j^{[m]}(k)\right). \tag{3.19}$$

Розрахунок коефіцієнту розбиття проводиться для кожного вузла системи разом з налаштуванням їх параметрів, тобто співвідношення (3.12) та (3.19) реалізуються одночасно. На кожному такті навчання вузол $PC^{*[m]}$ визначає найкращий елемент каскаду, що забезпечує максимальне значення коефіцієнта розбиття у кожний поточний момент k, при цьому не виключається ситуація, коли в різні моменти обробки інформації «переможцями» виявляться різні вузли.

Кожен з каскадів розглянутої системи відрізняється від інших числом кластерів, на які розбивається оброблюваний потік даних. Тому, якщо вузли $PC_j^{[m]}$ і $PC^{*[m]}$ оцінюють якість кластеризації без урахування кількості сформованих класів, то вузли системи, позначені $XB^{[m]}$ та XB^*, оцінюють результати з урахуванням числа кластерів у кожному каскаді. Одним з таких показників є індекс Ксі-Бені, який для фіксованої вибірки з N спостережень може бути записаний у вигляді

$$XB_j^{[m]} = \frac{\left(\sum_{k=1}^{N}\sum_{l=1}^{m+1}\left(u_{jl}^{[m]}(k)\right)^2 \left\|x(k) - c_{jl}^{[m]}\right\|^2\right)/N}{\min\limits_{l\neq q}\left\|c_{jl}^{[m]} - c_{jq}^{[m]}\right\|^2} = \frac{NXB_j^{[m]}}{DXB_j^{[m]}}. \quad (3.20)$$

Вираз (**Помилка! Джерело посилання не знайдено.**) також можна записати у рекурентній формі

$$XB_j^{[m]}(k+1) = \frac{NXB_j^{[m]}(k+1)}{DXB_j^{[m]}(k+1)} = \frac{NXB_j^{[m]}(k)}{\min\limits_{l\neq q}\left\|c_{jl}^{[m]}(k+1) - c_{jq}^{[m]}(k+1)\right\|^2} + \\ + \frac{\sum_{l=1}^{m+1}\left(u_{jl}^{[m]}(k+1)\right)^2 \left\|\mathrm{x(k+1)} - \mathrm{c_{jl}^{[m]}(k+1)}\right\|^2 - \mathrm{NXB_j^{[m]}(k)}}{\mathrm{(k+1)}\min\limits_{\mathrm{l\neq q}}\left\|\mathrm{c_{jl}^{[m]}(k+1) - c_{jq}^{[m]}(k+1)}\right\|^2}, \quad (3.21)$$

при цьому рекурентні вирази (3.19) та (3.21) обчислюються одночасно.

Індекс Ксі-Бені є по суті співвідношенням відхилення всередині кластерів $\mathrm{NXB_j^{[m]}}$ до величини поділу кластерів $\mathrm{DXB_j^{[m]}}$. Оптимальному числу кластерів у каскаді відповідає мінімальне значення (3.20) та (3.21). Тому процес нарощування каскадів у системі продовжується доки значення індексу не почне збільшуватися. Цей процес контролює вузол архітектури XB*.

Варто зауважити, що оскільки вузли кожного каскаду відрізняються тільки значенням фаззіфікатору, ефективність роботи кожного каскаду доцільно оцінювати за допомогою розширеного індексу Ксі-Бені EXB

$$\mathrm{EXB_j^{[m]}} = \frac{\left(\sum_{\mathrm{k=1}}^{\mathrm{N}}\sum_{\mathrm{l=1}}^{\mathrm{m+1}}\left(\mathrm{u_{jl}^{[m]}(k)}\right)^{\beta_{[m]}} \left\|\mathrm{x(k) - c_{jl}^{[m]}}\right\|^2\right)/\mathrm{N}}{\min\limits_{\mathrm{l\neq q}}\left\|\mathrm{c_{jl}^{[m]} - c_{jq}^{[m]}}\right\|^2} = \frac{\mathrm{NEXB_j^{[m]}}}{\mathrm{DEXB_j^{[m]}}}. \quad (3.22)$$

або його рекурентної форми

$$\mathrm{EXB}_{\mathrm{j}}^{[\mathrm{m}]}(\mathrm{k}+1)=\frac{\mathrm{NEXB}_{\mathrm{j}}^{[\mathrm{m}]}(\mathrm{k}+1)}{\mathrm{DEXB}_{\mathrm{j}}^{[\mathrm{m}]}(\mathrm{k}+1)}=\frac{\mathrm{NEXB}_{\mathrm{j}}^{[\mathrm{m}]}(\mathrm{k})}{\min\limits_{\mathrm{l}\neq\mathrm{q}}\left\|\mathrm{c}_{\mathrm{jl}}^{[\mathrm{m}]}(\mathrm{k}+1)-\mathrm{c}_{\mathrm{jq}}^{[\mathrm{m}]}(\mathrm{k}+1)\right\|^{2}}+$$

$$+\frac{\sum_{\mathrm{l}=1}^{\mathrm{m}+1}\left(\mathrm{u}_{\mathrm{jl}}^{[\mathrm{m}]}(\mathrm{k}+1)\right)^{\beta_{[\mathrm{m}]}}\left\|\mathrm{x}(\mathrm{k}+1)-\mathrm{c}_{\mathrm{jl}}^{[\mathrm{m}]}(\mathrm{k}+1)\right\|^{2}-\mathrm{NEXB}_{\mathrm{j}}^{[\mathrm{m}]}(\mathrm{k})}{(\mathrm{k}+1)\min\limits_{\mathrm{l}\neq\mathrm{q}}\left\|\mathrm{c}_{\mathrm{jl}}^{[\mathrm{m}]}(\mathrm{k}+1)-\mathrm{c}_{\mathrm{jq}}^{[\mathrm{m}]}(\mathrm{k}+1)\right\|^{2}},\quad(3.23)$$

де $\beta_{[\mathrm{m}]}$ – фаззіфікатор найкращого з вузлів m-го каскаду.

Таким чином, процес еволюції пропонованої системи зумовлений максимізуванням поточного значення показника якості кластерування потоку даних, що надходять на опрацювання в онлайн режимі.

3.6 Онлайн модифікація методу Х-середніх на основі ансамблю самоорганізовних мап Т. Кохонена

Для формального знаходження числа кластерів m був розроблений метод Х-середніх [142], заснований на статистичному аналізі розподілу даних у вихідному масиві X. Якщо при роботі з К-середніми число кластерів m було вибрано правильно, то отримані результати повністю збігаються з результатами Х-середніх.

Останні роки в зв'язку з інтенсивним розвитком Data Stream Mining природно виникла необхідність вирішення завдань кластерування в online режимі, коли дані на обробку послідовно надходять спостереження за спостереженням, обсяг масиву N не обмежений і зростає з часом, а k набуває сенсу поточного дискретного часу. У подібній ситуації стандартні К-середні неефективні, проте з успіхом можуть бути використані кластерувальні нейронні мережі Т. Кохонена (SOM), що вирішують завдання в online режимі, а одержаний результат повністю збігається з К-середніми в силу використання загального критерію кластеризації-самонавчання, заснованого на евклідовій метриці. При

цьому проблема вибору m тут залишається відкритою, включення додаткових «мертвих» нейронів в мережу, як правило, її не вирішує, а використання X-середніх в online режимі в їх традиційній формі принципово неможливо.

Альтернативою стандартним X-середнім може бути використання ідеї кластерувальних ансамблів, при цьому нами пропонується формувати ансамбль на основі паралельно з'єднаних входами SOM^m, кожна з яких апріорно орієнтована на різну кількість можливих кластерів $m = 1, 2, 3, ..., M$. Таким чином, перша кластерувальна мережу ансамблю працює в припущенні $m = 2$, тобто в шарі Кохонена містить всього два нейрона з синаптичними вагами-центроїдами w_1^2 і w_2^2. Другий елемент ансамблю містить три нейрона з векторами синаптичних ваг w_1^3, w_2^3, w_3^3 і, нарешті, остання SOM^M ансамблю працює в припущенні, що число можливих кластерів дорівнює M, тобто містить M нейронів-адаптивних лінійних асоціаторів.

3.7 Алгоритм налаштування нейронних мереж ансамблю

Для навчання кожної з окремих SOM^m можуть бути використані як стандартні кохоненівські WTA- і WTM-правила самонавчання, так і їх модифікації.

Розглянемо процес самонавчання m-ої мережі Кохонена SOM^m, що містить m нейронів з синаптичними вагами $\left\{w_1^m, w_2^m, ...w_m^m\right\} \subset R^n$. В основі алгоритму налаштування синаптичних вагів полягає принцип конкурентного самонавчання, який реалізується в три основних етапи (конкуренція, кооперація, синаптична адаптація) і починається з аналізу вхідного вектора-образу $x(k)$, що надходить з рецепторного (нульового) шару на всі нейрони шару Кохонена. Для кожного з нейронів обчислюється відстань

$$D\left(x(k), w_j^m(k-1)\right) = \left\| x(k) - w_j^m(k-1) \right\|, \; j = 1, 2, ...m,$$

при цьому, якщо вхідні сигнали попередньо пронормовані за допомогою перетворення

$$x(k)=\frac{x(k)}{\|x(k)\|} \tag{3.24}$$

так, що $\|x(k)\|=1$, а в якості відстані використовується евклідова метрика, то мірою схожості (подібності) векторів $x(k), w_j^m(k-1)$ може служити скалярний добуток

$$sim\left(x(k), w_j^m(k-1)\right)=x^T(k)w_j^m(k-1)=\cos\left(x(k), w_j^m(k-1)\right). \tag{3.25}$$

Далі визначається нейрон-переможець «найближчий» до вхідного образу такий, що

$$sim\left(x(k), w^{m*}(k-1)\right)=\max_j sim\left(x(k), w_j^m(k-1)\right),$$

після чого, опускаючи тимчасово процес кооперації, можна уточнити синаптичні ваги переможця за допомогою рекурентного співвідношення

$$w_j^m(k)=\begin{cases} w_j^m(k-1)+\eta(k)\times \\ \times\left(x(k)-w_j^m(k-1)\right), \text{ якщо } w_j^m(k-1)=w^{m*}(k-1), \\ w_j^m(k-1) \text{ у протилежному випадку.} \end{cases} \tag{3.26}$$

Таким чином, процедура реалізує правило «переможець отримує все» (WTA), при цьому вектор синаптичних ваг переможця $w^{m^*}(k-1)$ «підтягується» до вхідного образу на відстань, що визначається величиною кроку

$$0<\eta(k)<1.$$

Регулювання кроку $\eta(k)$ зазвичай проводиться, виходячи з емпіричних міркувань, а загальна рекомендація полягає в тому, що він повинен монотонно зменшуватися в процесі самонавчання. У найпростішому випадку для регулювання кроку можуть бути використані співвідношення

$$\eta(k)=r^{-1}(k),\ r(k)=\alpha r(k-1)+\left\|x(k)\right\|^2,\ 0\le\alpha\le 1,$$

або

$$r(k)=\alpha r(k-1)+1,\ 0\le\alpha\le 1 \qquad (3.27)$$

для входів, нормованих відповідно до (3.24).

Зрозуміло, що при $\alpha=1$, $\eta(k)=k^{-1}$, тобто задовольняє умовам стохастичної апроксимації.

Важливою особливістю нейронної мережі Кохонена є наявність етапу кооперації, коли нейрон-переможець $w^{m^*}(k-1)$ визначає локальну область топологічного сусідства, в якому збуджується не тільки він сам, але і його оточення, при цьому більш «схожі» на переможця нейрони збуджуються сильніше ніж більш віддалені «сусіди». Ця область описується функцією сусідства $\varphi(j,l)$, $l=1,2,...,m$, що залежить від відстані $D\left(w^{m^*}(k-1),w_l^m(k-1)\right)=D\left(w_j^m(k-1),w_l^m(k-1)\right)$, між переможцем і будь-яким з

нейронів $w_l^m(k-1)$ шару Кохонена. Як правило $\varphi(j,l)$ - це ядерна функція симетрична щодо максимуму в точці з $D\left(w_j^m(k-1), w_j^m(k-1)\right)$ і приймаюча в ній одиничне значення $\varphi(j,l)=1$. Зі збільшенням відстані $D\left(w_j^m(k-1), w_l^m(k-1)\right)$ ця функція монотонно зменшується. У переважній більшості випадків в якості функції сусідства використовується гауссіан

$$\varphi(j,l)=\exp\left(-\frac{\left\|w_l^m(k-1)-w^{m*}(k-1)\right\|^2}{2\sigma^2}\right).$$

Використання функції сусідства призводить до алгоритму самонавчання

$$w_l^m(k)=w_l^m(k-1)+\eta(k)\varphi(j,l)\left(x(k)-w_l^m(k-1)\right)\forall l=1,2,...,m, \qquad (3.28)$$

що реалізує правило «переможець отримує більше» (WTM), при цьому при $l=j$ цей алгоритм збігається зі співвідношенням (3.26).

В принципі, можна взагалі відмовитися від етапу конкуренції та визначення переможця як такого. При цьому в ролі переможця в даному випадку виступає сам вхідний вектор-образ, а в якості функції сусідства використовується міра схожості (2).

При цьому алгоритм самонавчання m-го елемента ансамблю набуває вигляду

$$\begin{aligned} w_l^m(k) &= w_l^m(k-1)+\eta(k)\left[\cos\left(x(k), w_l^m(k-1)\right)\right]_+\left(x(k)-w_l^m(k-1)\right)= \\ &= w_l^m(k-1)+\eta(k)\left[x^T(k)w_l^m(k-1)\right]_+\left(x(k)-w_l^m(k-1)\right)= \\ &= w_l^m(k-1)+\eta(k)\left[y_l^m(k)\right]_+\left(x(k)-w_l^m(k-1)\right), \end{aligned} \qquad (3.29)$$

де $\left[y_l^m(k) \right]_+ = \max\left\{ y_l^m(k), 0 \right\}$ – невід'ємне значення l-го вихідного сигналу m-ої мапи Кохонена ансамблю.

Зрозуміло, що процедура (3.29) з обчислювальної точки зору набагато простіша стандартних алгоритмів (3.26), (3.28), завдяки виключенню етапу конкуренції і має ясний фізичний зміст.

3.8 Визначення кількості кластерів

В процесі роботи ансамблю постійно проводиться оцінка якості кластерування за допомогою критерію Цаліньського–Харабаша або в його стандартній формі, або за допомогою його online модифікації. При цьому критерій в загальному вигляді має форму

$$CH(m) = \frac{1}{m-1} TrS_B^m \left(\frac{1}{N-m} TrS_w^m \right)^{-1}, \tag{3.30}$$

де $S_B^m = \frac{1}{N} \sum_{j=1}^{m} N_j^m \left(w_j^m - \overline{w}^m \right) \left(w_j - \overline{w}^m \right)^T$ – матриця міжкластерної відстані для m кластерів;

$\overline{w}^m = \frac{1}{N} \sum_{j=1}^{m} N_j^m w_j^m$ – центр ваги масиву даних X;

N_j^m – кількість спостережень, що відносяться до j-го кластеру, $j = 1, 2, ...m$;

$S_w^m = \frac{1}{N} \sum_{j=1}^{m} \sum_{k=1}^{N} u_j(k) \left(x(k) - w_j^m \right) \left(x(k) - w_j^m \right)^T$ – матриця розсіяння m-го кластеру;

$u_j = \begin{cases} 1, \text{ якщо } x(k) \text{ належить j-му кластеру,} \\ 0 - \text{у протилежному випадку} \end{cases}$ – чітка функція належності k-го спостереження j-му кластеру.

Переписавши вираз для TrS_B^m у формі

$$TrS_B^m = \frac{1}{N}\sum_{j=1}^{m} N_j^m \left\| w_j^m - \overline{w}^m \right\|^2 ,$$

а TrS_w^m –

$$TrS_w^m = \frac{1}{N}\sum_{j=1}^{m} u_j(k) \left\| x(k) - w_j^m \right\|^2 ,$$

критерій (3.30) можна представити у вигляді

$$CH(m) = \frac{\frac{1}{m-1}\sum_{j=1}^{m}\sum_{\tau=k-s+1}^{k} N_j^m \left\| w_j^m - \overline{w}^m \right\|^2}{\frac{1}{N-m}\sum_{j=1}^{m}\sum_{\tau=k-s+1}^{k} u_j(k) \left\| x(k) - w_j^m \right\|^2} \qquad (3.31)$$

більш зручному з точки зору обчислювальної реалізації.

При аналізі даних, що надходять на обробку в online режимі, розрахунок критерію (3.31) доцільно організувати на ковзному вікні розмірності s $(s = 1,2,...N)$, при цьому в поточний момент часу k $CH(m)$ можна записати як

$$CH(m,k) = \frac{\frac{1}{m-1}\sum_{j=1}^{m} N_j^m(\tau) \left\| w_j^m(\tau) - \overline{w}^m(\tau) \right\|^2}{\frac{1}{N-m}\sum_{j=1}^{m}\sum_{k=1}^{N} u_j(\tau) \left\| x(\tau) - w_j^m(\tau) \right\|^2},$$

де

$$\overline{w}^m(\tau) = \frac{1}{s}\sum_{\tau=k-s+1}^{k} x(\tau).$$

Як оптимальної кількості кластерів у вибірці m^* приймається m, що забезпечує максимум значенню $CH(m)$, тобто

$$CH(m^*) = \max_m \{CH(2), CH(3), ..., CH(M)\}.$$

Пропонована процедура ансамблевого online кластерування на основі системи нейронних мереж Т. Кохонена є за суттю адаптивною модифікацією методу X-середніх, орієнтованою на обробку потоків даних, досить проста в чисельній реалізації і дозволяє вирішити задачу чіткого кластерування в умовах апріорно невідомого або змінного числа кластерів.

3.9 Імітаційне моделювання

Для підтвердження працездатності розробленого ансамблю самоорганізовних мап Т. Кохонена була вирішена задача кластерування на основі штучно згенерованої вибірки і тестових вибірок з UCI-репозіторія. Було взято набори даних:

1. Штучна вибірка RandomMatrix, яка наочно відображає три лінійно розділимих кластери. Вибірка RandomMatrix, як представлено на рис. 3.1, містить три лінійно розділимих класів, де кожен елемент вибірки має три випадкові параметри.

2. Тестова вибірка «Iris». Вибірка складається з даних про квітки ірису, по 50 примірників з трьох видів – Ірис щетиністий (Iris setosa), Ірис віргінський (Iris virginica) і Ірис різнокольоровий (Iris versicolor). Для кожного екземпляра вимірювалися чотири характеристики (в сантиметрах): довжина чашолистки (sepal length); ширина чашолистка (sepal width); довжина пелюстки (petal length); ширина пелюстки (petal width).

Всі дані були спочатку пронормовані на гіперкулю в інтервалі $[-1,1]$ і відцентровані щодо середнього значення.

З метою оцінки ефективності ансамблю самоорганізовних мап Т. Кохонена (SOM^m) результати кластерування порівнювались зі стандартним методом кластерування К-середніх. Для підтвердження якості кластерування було взято індекс Цаліньського-Харабаша який наведено в табл. 3.1 для вибірок RandomMatrix та Iris. Візуалізація результатів кластерування наведена на рис. 3.2 та рис. 3.3.

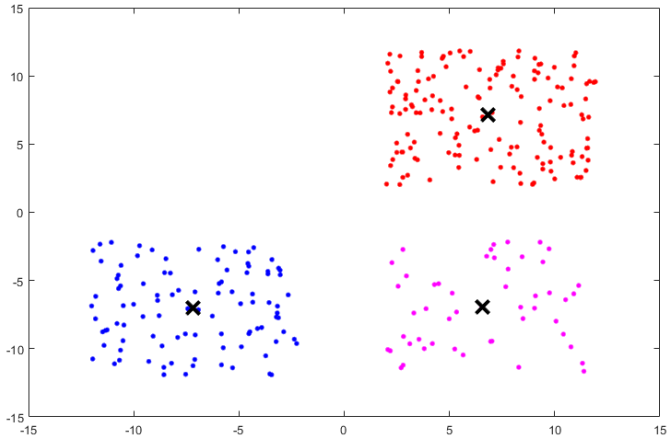
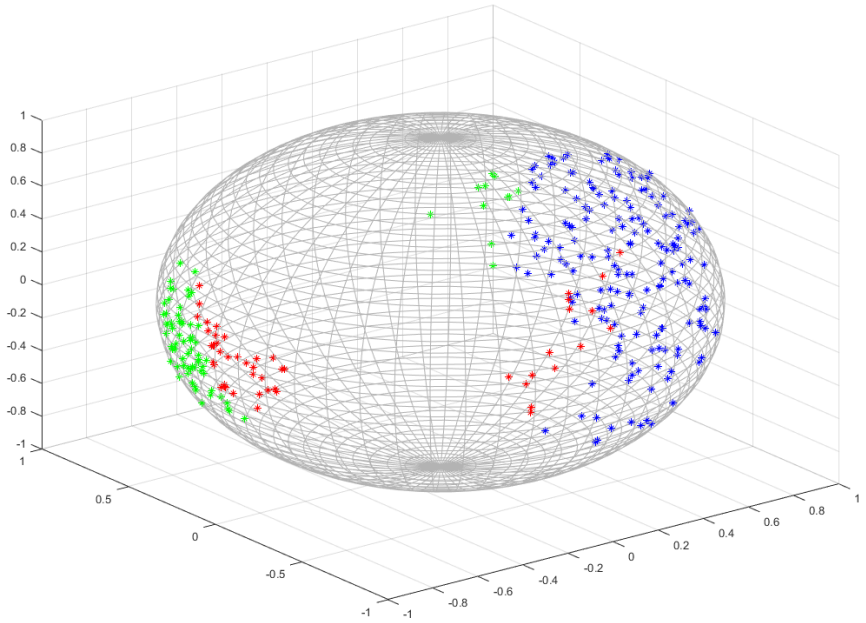


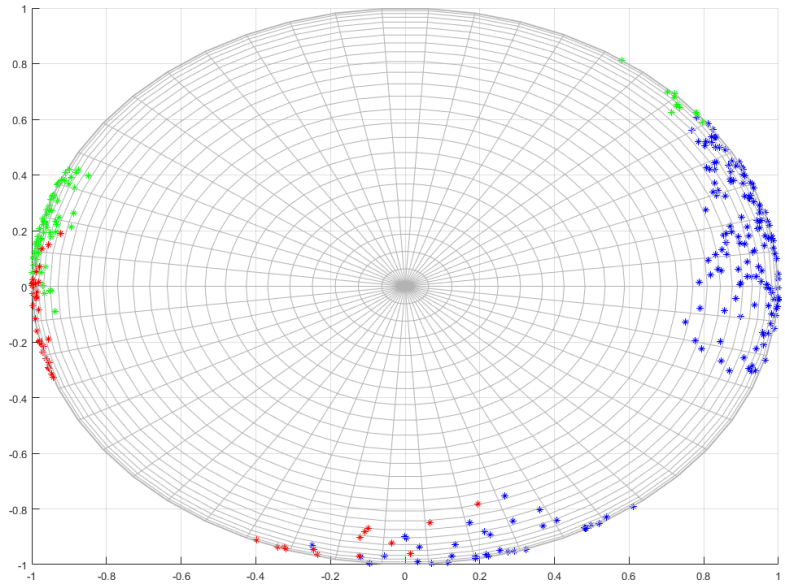
Рисунок 3.1 – Штучно згенерована лінійно розділима вибірка RandomMatrix

Таблиця 3.1 – Індекс Цаліньського-Харабаша для выбірок RandomMatrix та Iris

RANDOM MATRIX		
Метод	SOM^m	k-means
Індекс CH для 2 кластерів	650,119137400817	49,3904185658740
Індекс CH для 3 кластерів	782,603215072022	611,890289394461
Індекс CH для 4 кластерів	585,205208331037	411,869958970689
IRIS		
Метод	SOM^m	k-means
Індекс CH для 2 кластерів	506,384020879337	24,1478668157212
Індекс CH для 3 кластерів	521,993404839107	95,9506726585689
Індекс CH для 4 кластерів	463,871189144183	74,4873397342681

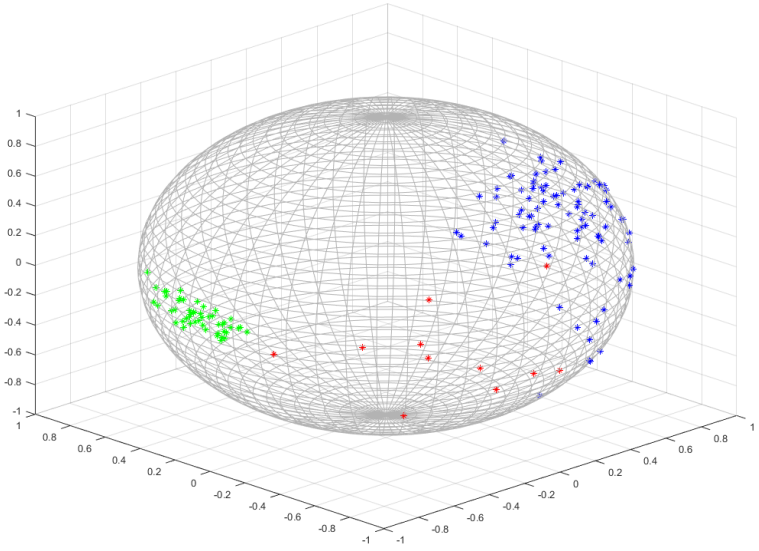


а)

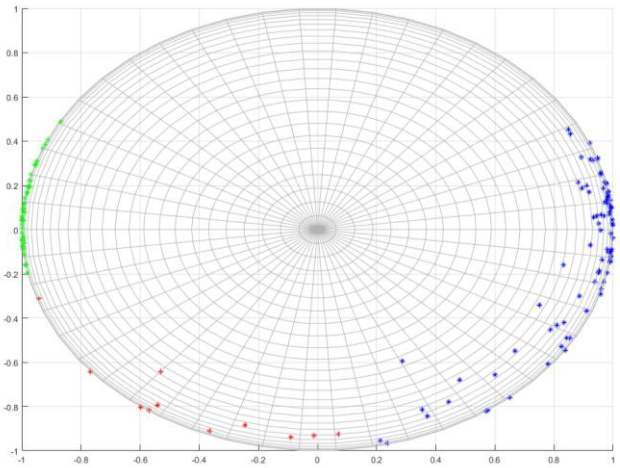


б)

Рисунок 3.2 – Візуалізація вибірки RandomMatrix: а) вид збоку; б) вид зверху



а)



б)

Рисунок 3.3 – Візуалізація вибірки Iris на: а) вид збоку; б) вид зверху

Таким чином, запропоновано online аналог методу X-середніх призначений для вирішення завдання кластерування потоку даних в умовах, коли число кластерів апріорно невідомо. В основі запропонованого підходу лежить ансамбль паралельно включених кластерувальних нейронних мереж Т. Кохонена, що містять різну кількість нейронів. Оптимальна кількість кластерів визначається числом нейронів у найкращій в сенсі якості з кластерувальних нейронних мереж. Введена модифікація методу X-середніх призначена для вирішення завдань в рамках інтелектуального аналізу потоків даних (Data Stream Mining).

3.10 Послідовне ядерне нечітке кластерування великих масивів даних на основі гібридної системи обчислювального інтелекту

У цьому підрозділі запропоновано архітектуру та методи самонавчання гібридної нейро-фаззі системи обчислювального інтелекту для кластерування даних за умов, коли кластери, що формуються, можуть мати довільну форму і взаємно перетинатися. В основі системи, що запропоновано, полягають нечітка узагальнена регресійна нейронна мережа та нейро-фаззі кластерувальна мережа Т. Кохонена, налаштування яких засновано як на лінивому, так і на навчанні, що базується на оптимізації.

Запропонована ядерна нечітка кластерувальна система містить чотири шари опрацювання інформації; перший прихований шар фаззіфікування входового простору, другий прихований шар агрегування, третій прихований шар обчислення центроїдів кластерів в просторі підвищеної розмірності та четвертий вихідний шар для обчислення рівнів належності.

На вхідний нульовий шар системи подаються $(n\times 1)$-вимірні вектори входових сигналів-образів $x(k)=(x_1(k),...,x_i(k),...,x_n(k))^T$ (тут $k=1,2,...,N,...$ – поточний дискретний час), попередньо оброблені, так що $-1\le x_i(k)\le 1$, $\frac{1}{N}\sum_{k=1}^{N}x_i(k)=0$, при цьому потік даних, що надходить, повинен бути розділений на m кластерів довільної форми, що ймовірно перетинаються. Перший прихований шар містить $n\,h$ (по h на кожний вхід) функцій належності $\mu_{li}(x_i)$, $i=1,2,...,n;\ l=1,2,...,h$ та виконує фаззіфікацію входового простору, при цьому кількість функцій належності на кожному вході $h>n$ задається або апріорно, або визначається в процесі самонавчання системи. Другий прихований шар здійснює агрегування рівнів належності, розрахованих у першому шарі, і складається з h блоків множення. Таким чином, перші два прихованих шари системи, що розглядається, по архітектурі співпадають з відповідними шарами відомих

нейро-фаззі систем Такагі–Сугено–Канга, Ванга–Менделя та ANFIS. Основна відмінність системи, що розглядається, від зазначених полягає в тому, що навчання останніх пов‘язано з налаштуванням синаптичних ваг за допомогою тих або інших алгоритмів оптимізації, а налаштування системи, що пропонується нами, засновано на лінивому навчанні та зводиться до налаштування центрів функцій належності. Доцільно зазначити, що відомі нейро-фаззі системи близькі за своєю природою до радіально-базисних нейронних мереж, а запропонована система є нечітким аналогом GRNN, тобто узагальненою регресійною нейро-фаззі системою.

Таким чином, якщо на вхід нейро-фаззі системи подано векторний сигнал $x(k)$, елементи першого прихованого шару обчислюють рівні належності $0 \le \mu_{li}(x_i(k)) \le 1$, при цьому в якості функцій належності частіше за все використовуються одновимірні гаусіани вигляду

$$\mu_{li}(x_i(k)) = \exp\left(-\frac{(x_i(k) - c_{li}(k))^2}{2\sigma_i^2}\right),$$

де $c_{li}(k)$ – параметр центру l-тої функції належності на i-му вході, що налаштовується в процесі самонавчання;

σ_i^2 – параметр ширини.

У другому прихованому шарі обчислюються агреговані значення $\prod_{i=1}^{n} \mu_{li}(x_i(k)) = \varphi_l(k),$ при цьому, якщо на вхід системи подано $(n \times 1)-$ вимірний вектор $x(k)$, на виході другого прихованого шару з’являється $(h \times 1)-$ вимірний образ підвищеної розмірності $\varphi(k) = (\varphi_1(k),...,\varphi_l(k),...,\varphi_h(k))^T$. Надалі нечіткому кластеруванню підлягає послідовність $\varphi(1),\varphi(2),...,\varphi(N),....$ Третій прихований та четвертий вихідний шари системи, що розглядається, складають двошарову нечітку кластерувальну мережу Т. Кохонена, в першому шарі якої відновлюються центроїди кластерів $c_1^K,...,c_j^K,...,c_m^K \in R^h, \quad j = 1,2,...,m$, а у

вихідному – рівні належності $u_j(k)$ кожного вектора $\varphi(k)$ до кожного кластеру з центроїдом c_j^K.

Процес самонавчання системи, що розглядається, складається з налаштування функцій належності першого прихованого шару на базі лінивого навчання і методів еволюційних систем і налаштування центроїдів кластерів, що формуються у третьому прихованому шарі.

Процес налаштування функцій належності першого прихованого шару за для наочності проілюструємо на прикладі двовимірного вхідного вектора $x(k) = (x_1(k), x_2(k))^T$, $k = 1, 2, ...,$ при цьому кількість функцій належності на кожному вході $h = 5$. Таким чином, кількість центрів $c_{li}(k)$, що налаштовуються, визначається значенням $nh = 10$.

У найпростішому випадку лінивого навчання припускається $c_1(0) = x(1), c_2(0) = x(2), ..., c_l(0) = x(l), ..., c_5(0) = x(5)$, а подальше корегування центрів $c_l(k)$ по мірі надходження даних до системи здійснюється згідно до техніки, прийнятої у GRNN.

При послідовному опрацюванні даних більш зручним видається підхід, коли початкові положення центрів функцій належності $c_{li}(0)$ рівномірно розподілені за координатними осями x_1, x_2, при цьому відстань між ними визначається співвідношенням

$$\Delta(0) = \frac{x_{i\max} - x_{i\min}}{h-1} = \frac{2}{h-1} = 0.5$$

для $-1 \le x_i(k) \le 1.$

Таким чином, у випадку багатовимірного вектору входів $x(k) \in R^n$, центри $c_{li}(0)$ рівномірно розподіляються по осям гіперкубу $[-1,\ 1]^n$.

Нехай на вхід системи подане перше спостереження $x(1) = (x_1(1), x_2(1))^T$. Далі на кожній із осей знаходяться центри – «переможці» (аналоги нейронів – «переможців» у SOM) $c_{li}^*(0)$ найближчих до $x_i(1)$ в сенсі відстані

$$d_{li} = |x_i(1) - c_{li}(0)|,$$

тобто

$$c_{li}^*(0) = \arg\min\{d_{1i}, d_{2i}, ..., d_{hi}\}.$$

Пошук центра-«переможця» є за своєю суттю реалізацію процесу конкуренції за Т. Кохоненом з тією лише різницею, що «переможці» по різним осям можуть належати функціям належності з різними індексами l. Далі ці «переможці» підтягуються до компонент вхідного сигналу $x_i(1)$ згідно з WTA-правилом самонавчання, яке може бути записано у вигляді

$$c_{li}(1) = \begin{cases} c_{li}^* + \eta_{li}(k)(x_i(1) - c_{li}^*(0)) - \textit{для переможця}, \\ c_{li}(0) - \textit{для всіх інших}. \end{cases}$$

Для довільних n, h та k WTA-правило самонавчання першого прихованого шару має вигляд

$$c_{li}(k) = \begin{cases} c_{li}^*(k-1) + \eta_{li}(k)(x_i(k) - c_{li}^*(k-1)) - \textit{для переможця при } l = 1,2,...,h; i = 1,2,... \\ c_{li}(k-1) - \textit{для всіх інших}, \end{cases}$$

де $\eta_{li}(k)$ – параметр кроку навчання, який монотонно зменшується в процесі налаштування.

Як вже було зазначено вище, послідовність з $(h\times 1)-$вимірних векторів $\varphi(1),\varphi(2),...,\varphi(N),...$ піддається нечіткому кластеруванню у третьому та четвертому шарах системи, що розглядається.

У класі процедур нечіткого кластерування найбільш формальними з математичної точки зору є алгоритми, що засновані на цільових функціях та вирішують задачу їх оптимізації за тими чи іншими апріорними припущеннями.

Вводячи функцію Лаґранжа

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

(тут $\lambda(k)$ – невід'ємні множники Лаґранжа) і вирішуючи систему рівнянь Каруша–Куна–Таккера, нескладно отримати кінцевий результат у вигляді

$$u_j(k)=\frac{(\left\|\varphi(k)-c_j^K\right\|^2)^{\frac{1}{1-\beta}}}{\sum_{l=1}^{m}(\left\|\varphi(k)-c_l^K\right\|^2)^{\frac{1}{1-\beta}}},$$

$$c_j^K=\frac{\sum_{k=1}^{N}u_j^{\beta}(k)\varphi(k)}{\sum_{k=1}^{N}u_j^{\beta}(k)},$$

$$\lambda(k)=-((\sum_{l=1}^{m}\beta\left\|\varphi(k)-c_l^K\right\|^2)^{\frac{1}{1-\beta}})^{1-\beta}$$

що співпадає при $\beta=2$ з FCM Дж. Бездека, а при $\beta\to 1$ є близьким до чіткого методу К-середніх.

У випадках, коли дані надходять на опрацювання послідовно в online режимі, може бути використана рекурентна версія вигляду

$$u_j(k) = \frac{(\left\|\varphi(k) - c_j^K(k-1)\right\|^2)^{\frac{1}{1-\beta}}}{\sum_{l=1}^{m}(\left\|\varphi(k) - c_l^K\right\|^2)^{\frac{1}{1-\beta}}},$$

$$c_j^K(k) = c_j^K(k-1) + \eta(k)u_j^\beta(k)(\varphi(k) - c_j^K(k-1)),$$

при цьому неважко зазначити, що друге співвідношення є за своєю суттю WTM-правилом самонавчання Т. Кохонена, де співмножник $u_j^\beta(k)$ грає роль функції сусідства.

4 МЕТОДИ ПРОСТОРОВО-ЧАСОВОЇ СЕГМЕНТАЦІЇ ВІДЕОРЯДІВ ТА МЕТРИЧНОГО ТЕМПОРАЛЬНОГО АНАЛІЗУ ЧАСТКОВОЇ СЕМАНТИКИ ДИНАМІЧНОЇ ВІЗУАЛЬНОЇ ІНФОРМАЦІЇ

При розв’язанні задачі аналізу відеоданих за їх вмістом одним з цілком природних підходів бачиться можливість розбиття вихідних даних на однорідні з точки зору заданих критеріїв сегменти для подальшої відповіді про контекст відео. Таке розбиття є темпоральною сегментацію відеоданих, тому, що будь-які відеодані можливо представити у вигляді деякої послідовності відеокадрів, що фактично і є багатовимірним часовим рядом. В рядах, індукованих відеоданими, існують різні види міжкадрової кореляції: наприклад, подібність сусідніх кадрів послідовності, що дозволяє застосувати апарат аналізу часових рядів з метою виявлення розладнань в них. Розладнання і означатимуть переходи між однорідними сегментами, тобто будуть визначати межі цих самих сегментів.

Оскільки багатовимірні часові ряди, відповідні відеопослідовністі, як правило, містять надлишкову інформацію, їх аналіз здійснюється на основі контролю деяких узагальнених характеристик таких, як спектри, кореляційні функції, тренди середніх значень, дисперсії, головні компоненти, тощо.

Задача радикально ускладнюється у випадках, коли обсяг вибірки не фіксований, що притаманне послідовному, покадровому, аналізу відеоданих в реальному або наближеному до реального масштабі часу. У подібних зростаючих вибірках кількість сегментів невідома, а в силу найбільшої варіативності змісту відео і умов реєстрації ніяких апріорних припущень про властивості відеоряду зазвичай зробити не можна. Таким чином, на перший план виступає вже не просто сегментація, а послідовне виявлення моментів зміни властивостей відеоданих. Тому в цьому розділі розглянуто методи просторово-часової сегментації відеопослідовностей.

4.1 Виявлення змін властивостей багатовимірних часових рядів на основі VAR моделі

У цьому підрозділі розглядається задача послідовного (в реальному часі) виявлення властивостей багатовимірних часових рядів в процесі адаптивної ідентифікації моделі VAR процесу

$$B(i) = K_0 + \sum_{l=1}^{N} K_l B(i-l) + \xi(i), \tag{4.1}$$

де $K_0=\{k_{0i}\}$ – $(n\times 1)$-вектор середніх значень;

$K_l = \left\{k_{l_{ij}}\right\}$ – $(n\times n)$ матриці параметрів, N – порядок моделі.

Крім формули (4.1), VAR-модель можливо компактно описати в просторі станів

$$\begin{cases} B(i) = \Pi B(i-1) + \Pi_0 + E(i), \\ Y(i) = CB(i), \end{cases}$$

де

$$B(i) = \begin{pmatrix} B(i) \\ B(i-1) \\ \vdots \\ B(i-N+1) \end{pmatrix}, \Pi_0 = \begin{pmatrix} K_0 \\ \vec{0} \\ \vdots \\ \vec{0} \end{pmatrix}, \Pi = \begin{pmatrix} K_1 & \cdots & K_{N-1} & K_N \\ I_n & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & I_n & 0 \end{pmatrix},$$

$$E(i)=\begin{pmatrix}\xi(i)\\ \vec{0}\\ \vdots\\ \vec{0}\end{pmatrix},\ C=(I_n,0,\ldots,0),\ \ B(i)-(np\times 1),$$

де B – вектор станів;

П – $(np\times np)$-перехідна матриця;

$\vec{0}$ і 0 – $(n\times 1)$ та $(n\times n)$ нульові вектор і матриця відповідно.

Опис (4.1) дозволяє використовувати для аналізу багатовимірних сигналів потужний математичний апарат калманівської фільтрації.

При цьому вихідною інформацією для розв'язання задачі ідентифікації і виявлення змін є тільки сам n-вимірний часовий ряд Ф, значення якого в реальному часі надходять на адаптивний ідентифікатор.

Для спрощення подальших викладок введемо в розгляд матрицю

$$K=(K_0\vdots K_1\vdots\ldots\vdots K_N)$$

та вектор

$$B'(i)=(1,B^T(i-1),\ldots,B^T(i-p))^T$$

розмірності $((Nn+1)\times 1)$ відповідно, після чого перепишемо рівняння (4.1) у вигляді

$$B(i)=KB'(i)+\xi(i), \qquad (4.2)$$

де матриця апріорі невідомих параметрів K містить практично всю необхідну інформацію про властивості контрольованого сигналу.

Завдання ідентифікації полягає в тому, що в відповідність реальному сигналу (4.2) ставиться налаштовувана модель

$$\hat{B}(i) = K(i-1)B'(i), \tag{4.3}$$

матриця параметрів $B'(i)$ якої, уточнюються на кожному такті часу i шляхом мінімізації прийнятого критерію ідентифікації, є деякою функцією різниці розрахованих $B'(i)$ і експериментальних даних $B(i)$. При цьому синтезована модель (4.3) повинна бути працездатною і в режимі прогнозування, а порушення прогнозуючих властивостей може бути ознакою виникнення тих чи інших рекурентних процедур, які можуть бути представлені в узагальненому вигляді [143]

$$\begin{cases} K(i) = K(i-1) + \gamma(i)e(i)B'^{T}(i); \\ e(i) = B(i) - \hat{B}(i) = B(i) - K(i-1)B'(i), \end{cases} \tag{4.4}$$

де $\gamma(i)$ – скалярний або матричний коефіцієнт підсилення алгоритму, що визначає його властивості, і залежний від прийнятого критерію ідентифікації;

$e(i)$ – векторна помилка ідентифікації.

На практиці найбільшого поширення набули алгоритми, пов’язані з критерієм мінімуму суми квадратів помилок ідентифікації

$$I(i) = \sum_{u=1}^{i} \beta(u)\|e(u)\|^2 = \sum_{u=1}^{i}\sum_{j=1}^{n} \beta(u)e_j(u)^2 \tag{4.5}$$

і його модифікації, які визначаються прийнятою системою ваг $\beta(u)$. Найбільш популярним є метод найменших квадратів, у якому всі ваги мають однакову вагу, тобто

$$I(i) = \sum_{u=1}^{i} \left\| e(u) \right\|^2 . \tag{4.6}$$

Алгоритм, відповідний (4.4), (4.6) має вигляд

$$\begin{cases} K(i) = K(i-1) + \dfrac{e(k)B'^{T}(i)P(i-1)}{1 + B'^{T}(i)P(i)B'^{T}}; \\ P(i) = P(i-1) - \dfrac{P(i-1)B'(i)B'^{T}(i)P(i-1)}{1 + B'^{T}(i)P(i-1)X(i)}, \end{cases} \tag{4.7}$$

і при незмінних параметрах (4.2) забезпечує монотонну збіжність оцінок $K(i)$ до істинних значень параметрів K.

Альтернативою (4.6), (4.7) є однокрокова процедура

$$K(i) = K(i-1) + \frac{e(k)B'^{T}(i)}{B'^{T}(i)B'^{T}}, \tag{4.8}$$

породжувана однокроковим же критерієм ідентифікації $I(k) = \left\| e(k) \right\|^2$ і є узагальненням алгоритму Качмажа [144] на векторно-матричну модель (4.3). Володіючи високою швидкодією, процедура (4.8) не має фільтруючих властивостей, а тому не здатна розрізнити зміни в сигналі і вплив стохастичного компонента $\xi(i)$.

У зв’язку з цим представляється доцільним використання алгоритмів з кінцевою пам’яттю, що володіють як згладжувальними, так і слідкуючими властивостями, компроміс між якими і задається величиною пам’яті.

Ознакою виникаючих змін може служити втрата прогнозуючих властивостей моделі (4.3) і необхідність перебудови пам’яті алгоритму.

Повертаючись до критерію (4.5), зауважимо, що в класі породжуваних ним алгоритмів, найбільшого поширення набув експоненціально-зважений метод найменших квадратів з критерієм

$$I(i) = \sum_{u=1}^{i} \beta^{i-u} \left\| e(u) \right\|^2$$

і рекурентною процедурою налаштування

$$\begin{cases} K(i) = K(i) + \dfrac{e(k) B'^{T}(i) P(i-1)}{\beta + B'^{T}(i) P(i-1) B'(i)}; \\ P(i) = \dfrac{1}{\beta} \left(P(i-1) - \dfrac{P(i-1) B'(i) B'^{T}(i) P(i-1)}{\beta + B'^{T}(i) P(i-1) B'(i)} \right), \end{cases} \tag{4.9}$$

де $0 \le \beta \le 1$ – параметр згладжування.

Тут слід зауважити, що ідентифікатор з експоненціальним згладжуванням в загальному випадку є нестійким, що веде до «вибуху параметрів» коваріаційної матриці, який особливо часто виникає при високих розмірностях оброблюваного сигналу $B(i)$.

Таким чином, використання традиційного експоненційно-зваженого рекурентного методу найменших квадратів ускладнюється поганою обумовленістю інформаційної матриці

$$\sum_{u=1}^{i} \beta^{k-u} B'(u) B'^{T}(u),$$

породжуваної високим рівнем кореляції між компонентами $x_j(i)$.

У [145] ця проблема вирішується використанням замість операції обертання зваженої інформаційної матриці, заснованої на формулі Шермана–Моррісона, операції псевдообертання з використанням теореми Гревіля. Отриманий при цьому алгоритм вельми громіздкий з обчислювальної точки зору, що особливо позначається при великих *n*.

У зв'язку з цим нами пропонується ввести в розгляд багатовимірну модифікацію алгоритму експоненційно-зваженої стохастичною апроксимації [146] у вигляді

$$\begin{cases} K(i) = K(i-1) + \dfrac{e(i)B'^{T}(i)}{\beta r(i-1) + \left\|B'(i)\right\|^2}; \\ r(i) = \beta r(i-1) + \left\|B'(i)\right\|^2, \end{cases} \tag{4.10}$$

що є своєрідним компромісом між процедурами (4.8) і (4.9) і володіє необхідними згладжувальними і слідкувальними властивостями.

Механізм адаптації алгоритмів (4.9), (4.10) заснований на «подавленні» застарілої інформації, при цьому динамічні властивості алгоритмів повністю визначаються пам'яттю, на основі якої відбувається уточнення матриці поточних оцінок $K(i)$.

Як зазначалося, найкращими фільтруючими властивостями володіють алгоритми з необмеженою пам'яттю, такі, як рекурентний метод найменших квадратів, в той же час ці процедури мають погані слідкуючи властивості у випадку, якщо характеристики контрольованого сигналу змінюються в часі. У цьому випадку перевагу мають алгоритми з малою пам'яттю типу алгоритму Качмажа, проте ці процедури погано працюють в умовах завад.

Таким чином, при ідентифікації систем, в яких можуть відбуватися зміни, обсяг пам’яті алгоритму слід обирати на основі компромісу між його згладжувальними і слідкувальними властивостями. На жаль, в реальних ситуаціях характеристики збурень, дрейфів і стрибків апріорі невідомі і можуть змінюватися в процесі функціонування контрольованої системи. У цих умовах важко віддати перевагу одному алгоритму з фіксованою пам’яттю, а тому доцільно використовувати алгоритми зі змінною пам’яттю, величину якої можна оперативно змінювати залежно від наявності або відсутності змін.

У [147], [148] запропоновано метод регулювання параметра згладжування, заснований на контролі статистики, що характеризує помилку прогнозування одновимірного сигналу. Передбачається, що налаштування параметрів моделі відбувається за допомогою експоненційно-зваженого алгоритму Калмана–Мейна, який для j-ої компоненти $x_j(i)$ може бути записаний у вигляді

$$\begin{cases} k_j(i) = k_j(i-1) + \dfrac{\left(e(i)B'^T(i)P_j(i-1)\right)_j}{\beta\sigma_i^2 + B'^T(i)P_j(i-1)B'(i)}; \\ P_j(i) = \dfrac{1}{\beta}\left(P_j(i-1) - \dfrac{P_j(i-1)B'(i)B'^T(i)P_j(i-1)}{\beta\sigma_i^2 + B'^T(i)P_j(i-1)B'(i)} \right), \end{cases} \tag{4.11}$$

де $k_j(i)$ – j-й рядок матриці $K(i)$;

$(\bullet)_j$ – j-й рядок відповідного множення матриць.

Якщо ж дисперсії окремих компонент $\xi_j(i)$ вектора збурень невідомі, то в (4.11) можна використовувати оцінку у вигляді

$$\begin{cases} \sigma_j^2(i) = \sigma_j^2(i-1) + p_j(i-1)\left(\sigma_j^2(i-1) - l_j^2(i)\right); \\ p_j(i) = \dfrac{1}{\beta}\left(p_j(i-1) - \dfrac{p_j^2(i-1)}{\beta + p_j(i-1)} \right). \end{cases} \quad (4.12)$$

Також передбачається, що модель досить точно описує контрольований сигнал на часових інтервалах t спостережень, а параметри можуть змінюватися стрибками в довільні моменти часу i_a. Змінне значення $\beta(i)$ регулюється за допомогою статистики

$$T_j(i) = \sum_{u=i-t}^{i} \frac{l_j^2(u)}{\beta(i-1)\sigma_j^2 + B'^T(u)\, p_j(u-1) B'(u)}, \quad (4.13)$$

яка має χ^2 розподіл з t ступенями свободи, при цьому $\beta(0)=1$. Регулювання $\beta(i)$ здійснюється в дискретні моменти часу φt згідно з правилом

$$\beta(i) = \begin{cases} 1, \text{ при } i < t, i = \varphi t, T_j(i) \le \chi_g^2, \\ \beta(i-1) - \Delta\beta, \text{ при } i = \varphi t, T_j(i) > \chi_g^2, \\ \beta(i-1), \text{ при } \varphi t < i < (\varphi - 1)t, \varphi = 1,2,..., \end{cases} \quad (4.14)$$

де χ_g^2 – квантиль закону χ^2, відповідний рівню значущості g;

$\Delta\beta$ – крок регулювання.

Правило (4.14) передбачає зміну β(*i*) при значеннях *i*, що є кратними *t*, при проміжних *i* величина β(*i*) лишається незмінною, а факт змін фіксується в момент реалізації другого співвідношення (4.14).

Громіздкість і інерційність цієї процедури змушує шукати інші більш швидкі та ефективні прийоми виявлення змін.

Так в [149] запропонований метод регулювання параметра згладжування β на основі критерію Манна–Уітні. При цьому контрольованою характеристикою є значення

$$\sum_{u=i-t+1}^{t} \operatorname{sign}(b_j(u) - \hat{b}_j(u)) \geq \gamma, \qquad (4.15)$$

де γ – деяке порогове значення;

t – величина ковзного контрольного вікна;

$$\operatorname{sign}(b_j(u) - \hat{b}_j(u)) = \begin{cases} 0, & \text{при } b_j(u) = \hat{b}_j(u); \\ +1, & \text{при } b_j(u) > \hat{b}_j(u); \\ -1, & \text{при } b_j(u) \leq \hat{b}_j(u). \end{cases}$$

Процес контролю починається зі значення β(1)=0, що відповідає максимальній швидкодії алгоритму (4.10). Експоненційно-зважений рекурентний метод найменших квадратів в цій ситуації, природно, принципово непрацездатний. У процесі ідентифікації можливе виникнення наступних ситуацій:

$$\sum_{u=i-t+1}^{t} \operatorname{sign}(b_j(u) - \hat{b}_j(u)) \geq \delta,$$

що означає домінування стохастичної компоненти ξ_i сигналу b_j над «дрейфовою». У цьому випадку необхідно поліпшити згладжувальні властивості алгоритму, тобто збільшити його за правилом

$$\beta(i) = \beta(i-1) + \Delta\beta.$$

У ситуації (4.15) переважає дрейфова компонента сигналу і алгоритм не встигає відстежувати виниклі зміни. У цій ситуації необхідно зменшити пам’ять

$$\beta(i) = \beta(i-1) - \Delta\beta$$

і зафіксувати факт змін.

Для контролю за змінами багатовимірного часового ряду пропонується використовувати модифікацію (4.15) у вигляді

$$\max_j \left(\sum_{u=i-t+1}^{t} \operatorname{sign}(b_j(u) - \hat{b}_j(u)) \right) \geq \gamma, \qquad (4.16)$$

тобто одночасно контролювати всі компоненти і фіксувати факт змін, якщо розладнання станеться хоча б з однією компонентою b_j (i).

На практиці більш широке поширення отримали не статистичні, а евристичні процедури, такі, як методи Чоу, Робертса–Ріда, Брауна, Тріггa–Ліча,

Шоуна і т.п. [149]-[154], що мають у своїй основі деякі евристики, а тому несуть елемент суб'єктивізму. В умовах дефіциту апріорної та поточної інформації про характеристики та властивості контрольованого сигналу перевагу слід віддати цим методам.

Основа більшості евристичних методів наступна: задається набір значень параметра згладжування β, наприклад, 0; 0.05; 1; 0.1; …; 0.95; 1 і набір характеристик, що визначають якість ідентифікації. Найчастіше це:

– поточна похибка оцінювання *i*-ої компоненти:

$$e_j(i,\beta) = b_j(i) - \hat{b}_j(i,\beta); \tag{4.17}$$

– кумулятивна сума похибок:

$$S_j(i,\beta) = e_j(i,\beta) + S_j(i-1,\beta);$$

– середнє абсолютне значення похибки:

$$d_j(i,\beta) = (1-\beta)\left|e_j(i,\beta)\right| + \beta d_j(i-1,\beta);$$

– середня похибка:

$$\overline{e}_j(i,\beta) = (1-\beta)e_j(i,\beta) + \beta\overline{e}_j(i-1,\beta);$$

– відносна середня похибка:

$$\tilde{e}_j(i,\beta) = (1-\beta)\frac{e_j(i,\beta)}{b_j(i)} + \beta\tilde{e}_j(i-1,\beta);$$

– середня квадратична похибка:

$$\bar{e}_j^2(i,\beta) = (1-\beta)e_j^2(i,\beta) + \beta\bar{e}_j^2(i-1,\beta).$$

Якщо значення обраної контрольованої характеристики перевищує деякий поріг γ, приймається рішення про те, що відбулися зміни і необхідно відповідним чином відкоригувати параметр згладжування β.

Стаціонарний стохастичний сигнал зазвичай стикається з β, який знаходиться в інтервалі $0.7 \le \beta \le 0.99$ [150]. Найпростіша форма контролю полягає в тому, що коли значення $\tilde{e}_i(k,\beta)$ при деякому β перевищує поріг 0.05 [152152], параметр згладжування зменшується за правилом

$$\beta(i) = \beta(i-1) - \Delta\beta$$

і процес ідентифікації триває з новим $\beta(i)$. Якщо значення β перевершує поріг 0.7 ($\beta(i) \le 0.7$), приймається рішення про виникнення зміни в сигналі.

Більш ефективним, але і більш складним є метод Чоу. Тут одночасно налаштовуються три моделі зі значеннями параметрів згладжування β, $\beta + \Delta\beta$ і $\beta - \Delta\beta$. Якщо у момент часу k кращий результат отриманий, наприклад, з

$$\beta(i) = \beta + \Delta\beta,$$

то на наступному такті часу використовується нова трійка параметрів згладжування β, $\beta+\Delta\beta$ і $\beta+2\Delta\beta$. Якщо ж найкраща модель отримана при $\beta(i)=\beta-\Delta\beta$, то формується трійка β, $\beta-\Delta\beta$ і $\beta-2\Delta\beta$, та, нарешті, якщо найкращий результат досягається при $\beta(i)=\beta$, то зберігається набір β, $\beta+\Delta\beta$ і $\beta-\Delta\beta$.

Більш простими і оперативними є методи, що використовують контрольний стежуючий сигнал (трекінг-сигнал), що є індикатором змін контрольованих сигналів [150]-[151]. Існує ряд форм стежуючого сигналу, при цьому його вихід за певні межі свідчить про виниклі зміни.

Р. Брауном в якості стежуючого сигналу запропоновано використовувати вираз [153]

$$T_j^K(i)=\frac{\sum_{u=1}^{i} e_j(u)}{\sqrt{\sigma_j^2(i)}},$$

фізичний зміст якого полягає в тому, що у разі, якщо модель, що налаштовується, адекватна контрольованому сигналу, сума похибок, в силу випадковості, варіює близько 0, не перевищуючи при цьому деяких границь, які встановлюються апріорно для даного рівня ймовірності при певній дисперсії суми похибок прогнозу , яка прагне до значення

$$\lim_{i\to\infty}\sigma_j^2(i)=\frac{1}{1-(1-\beta)^{2(Nn+1)}}\sigma_j^2.$$

Для практичних розрахунків Р. Браун замість середньоквадратичних похибок σ_j запропонував використовувати середнє абсолютне відхилення

$$d = \int_{-\infty}^{\infty} \left| e - M\{e\} \right| p(e) de \approx \sum_{u=0}^{n} \left| e(u) - M\{e\} \right| P_j .$$

Це відхилення пропорційно середньоквадратичному значенню похибки, оскільки

$$\Delta = (e - M\{e\}) / \sigma_j, \; d_j = \sigma_j \int_{-\infty}^{\infty} |\Delta| \, p(\Delta) d\Delta,$$

при цьому для широкого класу розподілів коефіцієнт пропорційності змінюється незначно [151] (для нормального розподілу $\frac{d_j}{\sigma_j} = \sqrt{2\pi} = 0,7979$)

Якщо величина d_j розрахована згідно з виразом

$$d_j(i) = (1-\beta) \left| e_j(i) \right| + \beta d_j(i-1),$$

а стежуючий сигнал –

$$T_j^{\beta}(i) = \frac{\sum_{u=1}^{i} e_j(u)}{d_j(i)},$$

то (1-P_j)% -ні границі можна записати у вигляді

$$\pm \tau_{j,1-P_j} = \pm \frac{\sigma_j}{2} \sqrt{\frac{\pi(1-\beta)}{1-(1-\beta)^{2(Nn+1)}}}.$$

Д. Трігг і А. Ліч запропонували в якості стежуючого сигналу використовувати співвідношення [154]

$$T_j^{TL}(i) = \frac{T_j'(i)}{d_j(i)}, \tag{4.18}$$

де

$$T_j'(i) = (1-\beta)e_j(i) + \beta' T_j'(i-1) \tag{4.19}$$

– не спільна сума відхилень, а згладжена похибка, при цьому має дотримуватися нерівність $\beta' \le \beta$.

При $\beta' = \beta$ стежуючий сигнал буде варіюватися між -1 і +1. Для введення автоматичного зворотного зв'язку Д. Трігг і А. Ліч запропонували розраховувати параметр згладжування згідно співвідношенню $\beta(i)=1-|T^{TL}(i)|$, а розладнання в сигналі фіксувати при істотних змінах $\beta(i)$. Відома також модифікація Шоуна [150] з $\beta(i)=1-|T^{TL}(i-1)|$.

Зростання стежуючого сигналу свідчить про збільшення розбіжності між моделлю і контрольованою послідовністю, для компенсації якого необхідна більш швидка реакція ідентифікатора, яка забезпечується більш низьким значенням параметра згладжування. Таким чином, забезпечується негативний зворотний зв'язок.

У [151] підкреслюється, що при ідентифікації процесів з досить гладким дрейфом, переваги має метод Брауна. Різкі стрибки краще визначаються за допомогою стежуючого сигналу Трігг–Ліча.

Основним недоліком підходу до виявлення змін, розглянутого вище, є значна кількість параметрів налаштовуваної моделі (4.3), що становить $n(Nn+1)$,

та може викликати певні труднощі при великих n і високих частотах надходження інформації на обробку.

Досить зручним математичним апаратом для вирішення завдання виявлення зміни властивостей одновимірних стохастичних послідовностей є експоненційне згладжування [150]-[152], суть якого можна пояснити наступним прикладом.

Введемо в розгляд елементарну модель виду

$$b_j(i) = k_j + \xi_j(i) \tag{4.20}$$

і припустимо, що коефіцієнт k_j час від часу може змінитися стрибком так, як це показано на рис. 4.1.

При цьому необхідно зазначити, що величина і момент часу зміни коефіцієнта k_j апріорі невідомі, а інтервал часу d_j-a_j, протягом якого значення коефіцієнта k_j залишається незмінним, істотно перевищує крок квантування сигналу (час між двома послідовними спостереженнями).

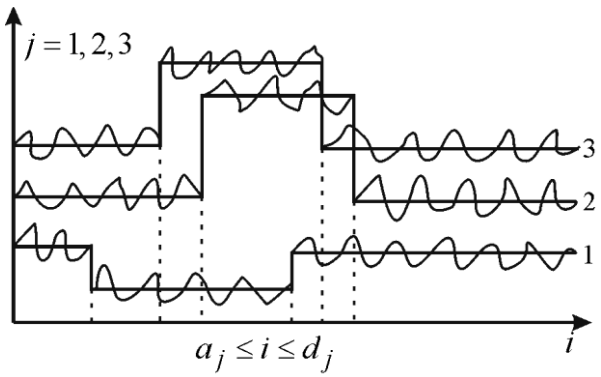


Рисунок 4.1 – Стрибкоподібні зміни середнього рівня часового ряду

Оскільки експоненційне згладжування розроблено для вирішення задачі прогнозування, розглянемо спочатку завдання побудови прогнозу $\hat{b}_j(i-1)$ в момент часу *i*.

Припустимо також, що параметри моделі (4.20) у момент *i*+1 збігаються з тими, які ми отримали до моменту часу *i*. У цьому випадку завдання зводиться до оцінки поточного значення $k_j(i)$ за *i* попереднім спостереженнями.

Оскільки значення k_j з плином часу змінюється, то для отримання цієї оцінки спостереження $b_j(i)$, $b_j(i - 1)$, … слід брати з великою вагою, або спостереження, отримані значно раніше. В принципі, це завдання може бути успішно вирішене за допомогою найпростішого методу ковзного вікна, при цьому, якщо заданий розмір цього вікна *t*, то згладжена оцінка має вигляд

$$\bar{b}_j(i) = \frac{1}{t} \sum_{u=i-t+1}^{k} b_j(u). \qquad (4.21)$$

Швидкість реакції ідентифікатора використовує апарат ковзного вікна, але зміна в сигналі залежить від величини *S* усереднених останніх спостережень. Ця швидкість, збільшується зі зменшенням *t* в відсутність змін і зменшує точність одержуваної оцінки $k_j(i) = \bar{b}_j(i)$, оскільки її дисперсія може бути записана у вигляді

$$\sigma_{k_j}^2 = \frac{\sigma_j^2}{t}.$$

Звідси видно, що точність оцінки $k_j(i)$ та швидкість реакції ідентифікатора суперечать один одному.

З урахуванням очевидного співвідношення

$$\bar{b}_j(i) = \bar{b}_j(i-1) + \frac{b_j(i) - b_j(i-t)}{t},$$

нескладно переписати (4.21) в рекурентній формі

$$\bar{b}_j(i) = \frac{1}{t} b_j(i) + (1 - \frac{1}{t})\bar{b}_j(i-1),$$

або

$$t_j(i) = \alpha b_j(i) + (1-\alpha) t_j(i-1) = \alpha b_j(i) + \beta t_j(i-1). \qquad (4.22)$$

У виразі (4.22) для відмінності експоненціального згладжування від ковзної середньої введено позначення $t_j(i)$ замість $\bar{b}_j(i)$. Величина α, що є аналогом $\frac{1}{t}$ в ковзному середньому, визначає ваги спостережень, присутніх у згладженій оцінці.

З виразу (4.22) випливає, що поточне значення згладженої величини $t_j(i)$ дорівнюватиме попередньому її значенню плюс деяка частка різниці між поточним спостереженням і попереднім значенням згладженої величини. Оскільки операція (4.22) реалізується для всіх спостережень часового ряду, її можна переписати з урахуванням всіх попередніх спостережень у вигляді

$$t_j(i) = \alpha b_j(i) + (1-\alpha)(\alpha b_j(i-1) + (1-\alpha)t_j(i-2)) =$$
$$= \alpha b_j(i) + (1-\alpha)(\alpha b_j(i-1) + (1-\alpha)(\alpha b_j(i-2) + (1-\alpha)t_j(i-3))) =$$
$$= \ldots = \alpha \sum_{u=0}^{i-1} (1-\alpha)^u b_j(i-u) + (1-\alpha)^i b_j(0).$$

Таким чином, величина $t_j(i)$ є якоюсь комбінацією всіх попередніх спостережень, вага яких зменшується за геометричною прогресією з часом. Поточне спостереження має вагу α, значення якого знаходиться в інтервалі [0,1]. Граничне значення $\alpha=0$ відповідає випадку $t=\infty$ в ковзному середньому. При цьому $t_j(i)=t_j(i\text{-}1)$, тобто значення $t_j(i)$ не залежить від нової інформації. Граничне значення означає, що передісторія взагалі не впливає на поточну оцінку, тобто

$$t_j(i) = b_j(i), \quad \sigma^2_{k_j} = \sigma^2_j.$$

Таким чином, точність і швидкість реакції ідентифікатора на швидкість зміни в сигналі повністю залежить від прийнятого значення α. Мала величина α забезпечує більшу точність оцінки b_i при стаціонарному сигналі, але повільну реакцію на зміни, в той час як збільшення α сприятиме збільшенню швидкості цієї реакції.

Зазвичай [153], величина α лежить в межах від 0.01 до 0.3, а число t – в межах від 6 до 200. Оскільки цей діапазон досить великий, у кожній конкретній задачі цей параметр потрібно вибирати спеціальним чином.

Необхідність використання великих значень α (малих значень t) вказує на невідповідність обраної та реальної моделей контрольованого сигналу, тобто може служити ознакою зміни його властивостей.

Вище зазначалося, що при експоненційному згладжуванні вага поточного спостереження має величину α, а ваги попередніх спостережень зменшуються у зворотному часі. У ковзному середньому середня вага останніх спостережень приймається однаковою і рівною $\frac{1}{t}$. Ваги ж усіх попередніх спостережень дорівнюють 0.

У [151] введено поняття середнього «віку» в ковзному середньому у вигляді

$$\rho = \frac{1}{t}(0+1+2+...+i-1) = \frac{t(t-1)}{2t} = \frac{1}{2}(t-1). \qquad (4.23)$$

Як випливає з (4.23), середній «вік» спостережень є середнє «віків» всіх окремих спостережень, взятих з вагами, рівними вагам цих окремих спостережень. При експоненційному згладжуванні вага спостереження, зробленого в момент часу *i-l*, буде $\alpha\beta^l$, тому середній «вік» спостережень може бути записаний у вигляді

$$\rho = (0\alpha\beta^0 + 1\alpha\beta^1 + ... + l\alpha\beta^l + ...) = \alpha \sum_{u=0}^{\infty} u\beta^l.$$

Умова рівності середнього «віку» спостережень в ковзному середньому і при експоненційному згладжуванні дозволяє знайти залежність між параметром α та вікном t в вигляді

$$\frac{\beta}{\alpha}=\frac{1-\alpha}{\alpha}=\frac{t-1}{2},$$

або

$$\begin{cases}\alpha=\dfrac{t-1}{2};\\ \beta=\dfrac{t-1}{t+1}.\end{cases}$$

При цьому особливий інтерес представляє реакція ідентифікатора з експоненційним згладжуванням на скачки, що виникають в контрольованому сигналі. Припустимо, що в момент часу i_a відбувається одиничний стрибок, тобто $k_j^a=k_j+1$ при $i \geq i_a$. Скориставшись стандартним дискретним z-перетворенням [155], з урахуванням того, що $\zeta[1]=\dfrac{z}{z-1}$, можна записати

$$\zeta[k_j(i)]=\frac{\alpha z}{z-\beta^{T_0}}\frac{z}{z-1}$$

або

$$\frac{\alpha z}{z-\beta^{T_0}}\frac{z}{z-1}=\frac{\beta^{T_0}}{\beta^{T_0}-1}\frac{\alpha z}{z-\beta^{T_0}}+\frac{\alpha}{1-\beta^{T_0}}\frac{z}{z-1},$$

де T_0 – період квантування контрольованого сигналу.

Переходячи назад в часову область, отримуємо

$$\beta_j(i)=1\frac{\alpha}{1-\beta^{T_0}}-\alpha\frac{\beta^{T_0}}{1-\beta^{T_0}}\beta^i,$$

Звідки випливає, що ідентифікатор з експоненціальним згладжуванням приходить до нового сталого стану k_j+1 тим швидше, чим менше β (більше α).

Загальна процедура експоненційного згладжування (4.22) призначена для обробки одновимірних сигналів $b_j(i)$. У рамках розглянутої нами проблеми доцільно ввести експоненційне згладжування багатовимірних послідовностей у формі

$$t(i)=AB'(i)+(I-A)t(i-1), \qquad (4.24)$$

де $t(i)=\left(t_1(i),t_2(i),...,t_n(i)\right)^T$;

$A=\mathrm{diag}(\alpha_1,\alpha_2,...,\alpha_n)-n\times n$ діагональна матриця;

I – $n\times n$ одинична матриця.

Якщо для контролю змін використовувати стежуючий сигнал Трігга–Ліча

$$\begin{cases} T_j^{TL}(i) = \dfrac{T'_j(i)}{d_j(i)}, \\ T'_j(i) = \alpha'_j e_j(i) + \beta'_j T'_j(i-1), \\ d_j(k) = \alpha \left| e_j(k) \right| + \beta_j d_j(i-1), \end{cases}$$

за аналогією з (4.24) нескладно ввести його векторний аналог

$$\begin{cases} T^{TL}(i) = \dfrac{T'(i)}{d_j(i)}; \\ T'(i) = A' e_j(i) + (I - A') T'(i-1); \\ d_j(i) = \operatorname{diag}(\alpha_j \left| e_j(i) \right| + (1 - \alpha_j) d_j(i-1)), \end{cases}$$

при цьому, природно, контролюється кожна компонента (n×1)- вектора $T^{TL}(i)$.

Виникнення змін у процесі виявляється шляхом перевірки нерівності

$$\max_i (T^{TL}(i) - T^{TL}(i-1)) \geq \gamma$$

по типу (4.16).

Дещо складніше йде справа з виявленням змін характеристик (дисперсій) збурень $\xi_j(\sigma_j^2)$ і внутрішньої структури багатовимірного часового ряду.

Для контролю дисперсій введемо вектор квадратів помилок (4.17)

$$\Sigma(i) = \left(e_1^2(i), e_2^2(i), ..., e_n^2(i) \right)^T$$

і експоненціально згладжений вектор дисперсій

$$t_{\sigma^2}(i) = A\Sigma(i) + (I - A)t_{\sigma^2}(i-1). \qquad (4.25)$$

Зрозуміло, що для стаціонарного сигналу і $\alpha_j = \frac{1}{i}$ вираз (4.25) описує дисперсії $\sigma^2_{k_j}$ оцінок компонент k_j, проте в нестаціонарної ситуації при $0 \le \alpha \le 1$ зростання компонент вектора (4.25) свідчить про виникнення змін. Оскільки в даній ситуації використання стежуючого сигналу є неможливим, необхідно контролювати умови

$$\max_i (t_{\sigma^2}(i) - t_{\sigma^2}(i-1)) \ge \gamma_{\sigma^2}.$$

Аналіз внутрішньої структури стаціонарної багатовимірної часової послідовності може бути проведений за допомогою її кореляційної матриці виду

$$R(i,\tau) = \frac{1}{i}\sum_{u=1}^{i}\left(B(u) - \bar{B}\right)\left(B(u-\tau) - \bar{B}\right)^T, \tau = 0,1,2,...,\tau_{\max},$$

яка містить інформацію про автокореляційні і взаємно кореляційні функції всіх компонент $b_j(i)$.

Для виявлення змін властивостей можна ввести експоненційно згладжену кореляційну матрицю

$$t_R(i,\tau) = \alpha(B(i) - t(i))(B(i-\tau) - t(i))^T,$$

а контроль вести за допомогою нерівності

$$Sp(t_R(i,\tau)t_R^T(i,\tau)) - Sp(t_R(i-1,\tau)t_R^T(i-1,\tau)) \geq \gamma_R,$$

де $Sp(\bullet)$ – символ сліду матриці;

$Sp(t_R t_R^T)$– квадрат сферичної норми матриці t_R.

Таким чином, на основі методології експоненційного згладжування можна в реальному часі забезпечити контроль за змінами всіх характеристик багатовимірних часових рядів.

4.2 Виявлення зміни властивостей багатовимірних часових рядів на основі аналізу головних компонент

Важливою проблемою при аналізі великих масивів (як за обсягом, так і за розмірністю) спостережень, заданих у формі часових рядів, є завдання їх стиснення з метою виділення латентних факторів, що визначають внутрішню структуру контрольованого сигналу, що в кінцевому підсумку має на меті зробити вихідний часовий ряд більш просто інтерпретуємим з погляду виявлення змін властивостей.

Одним з найбільш ефективних підходів до вирішення цього завдання є апарат факторного аналізу [156], в рамках якого найбільш широке поширення отримав метод головних компонент особливо в задачах розпізнавання образів,

обробки зображень, спектрального аналізу і т. п. і відомий ще як перетворення Карунена–Лоєва.

Вихідною інформацією для аналізу є $i \times n$ матриця спостережень

$$B = \begin{pmatrix} b_1(1) & b_2(1) & \ldots & b_n(1) \\ b_1(2) & b_2(2) & & b_n(2) \\ \vdots & & & \\ b_1(u) & b_2(u) & \cdots & b_n(u) \\ \vdots & & \ddots & \\ b_1(i) & b_2(i) & \cdots & b_n(i) \end{pmatrix},$$

утворена масивом з i n-вимірних векторів спостережень

$$B(u) = (b_1(u), b_2(u), ..., b_n(u))^T,$$

що для задачі інформаційного пошуку в відеоданих є представленням кадру за допомогою деякого вектора характеристик і її кореляційна $(n \times n)$ матриця виду

$$R(i) = \frac{1}{i} \sum_{u=1}^{i} (B(u) - \bar{B})(B(u) - \bar{B})^T = \frac{1}{i} \sum_{u=1}^{i} B^c(u) B^{cT}(u),$$

де $B^c(u) = B(u) - \bar{B}$ – центровані щодо середнього вихідні дані.

Метод головних компонент полягає в проектуванні спостережуваних вихідних даних з вихідного n-вимірного простору в m-вимірний $(n > m \geq 1)$ вихідний і зводиться до знаходження системи $w^1, w^2, ..., w^m$ ортогональних

власних векторів матриці $R(i)$ таких, що $w^1 = (w_1^1, w_2^1, ..., w_n^1)^T$ відповідають найбільшому власному значенню λ, матриці $R(i)$, $w^2 = (w_1^2, w_2^2, ..., w_n^2)^T$ – другому за величиною власному значенню λ і т. д. Інакше кажучи, мова йде про відшукання рішення матричного рівняння

$$(R(i) - \lambda_l I)w^l = 0$$

такого, що

$$\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_n \geq 0, \ \left\| w^l \right\|^2 = 1.$$

Таким чином, в алгебраїчних термінах рішення цієї задачі тісно пов'язане:

– з проблемою власних значень і знаходження рангу кореляційної матриці;

– в геометричному сенсі – це завдання переходу в простір більш низької розмірності з мінімальною втратою інформації;

– в статистичному сенсі – це задача знаходження множини ортонормованих векторів в просторі входів, приймаючих максимальну варіацію даних;

– в алгоритмічній сенсі – це завдання послідовного визначення (виділення) множини власних векторів w^1, w^2, …, w^m шляхом оптимізації кожного з локальних функціоналів, що утворюють глобальний критерій

$$I_w(i)=\frac{1}{i}\sum_{l=1}^{m}\sum_{u=1}^{i}\left(B^{cT}(u)w^l\right)^2$$

при обмеженнях

$$\begin{cases} w^{lT}w^N=0, \ \text{ при } l\neq N; \\ w^{lT}w^N=1. \end{cases}$$

Перша головна компонента, що несе максимум інформації про контрольований сигналі, може бути знайдена шляхом максимізації локального критерію

$$I_w^1(i)=\frac{1}{i}\sum_{u=1}^{i}(B^c(u)w^1)^2$$

за допомогою стандартного методу невизначених множників Лагранжа.

Далі від кожного вектора $B^c(u)$ віднімається його проекція на першу головну компоненту і обчислюється перша головна компонента залишків, що є другою головною компонентою вихідних даних і ортонормальною до першої.

Третя головна компонента обчислюється шляхом проектування кожного вхідного вектора на першу і другу головні компоненти, віднімання цієї проекції від кожного $B^c(u)$ і знаходження першої головної компоненти отриманих залишків, що є третьою компонентою вихідних даних. Решта головних компонент обчислюються рекурсивно згідно з описаною стратегією.

До теперішнього часу розроблено досить розвинене математичне та програмне забезпечення для реалізації перетворення Карунена–Лоєва, об’єднане

одним загальним недоліком: необхідністю апріорного задання матриці X фіксованої розмірності. Якщо ж дані надходять послідовно в реальному часі, стандартні процедури факторного аналізу стають непрацездатними.

У зв'язку з цим доцільним є використання рекурентних процедур реального часу для знаходження власних векторів матриці $R(i)$ шляхом послідовної обробки спостережень багатовимірного часового ряду

$$B(1), B(2), ..., B(i), B(i+1)...$$

без обчислення самої кореляційної матриці.

В [157] описаний штучний нейрон на основі адаптивного лінійного ассоціатора для обчислення першої головної компоненти в реальному часі. На рис. 4.2 наведена схема цього нейрона, модифікована для вирішення завдання виявлення зміни властивостей в багатовимірному сигналі на основі аналізу головної компоненти.

Для попередньо центрованих даних алгоритм навчання має вигляд

$$\begin{cases} w^1(i+1) = w^1(i) + \eta(i+1)(B(i+1) - y(i)w^1(i))y(i+1); \\ y(i+1) = B^T(i+1)w^1(i),\ w^1(0) \neq 0,\ y'(1) = B^T(1)w^1(0), \end{cases} \tag{4.26}$$

де $\eta(i+1)$ – параметр кроку, обраний досить малим для забезпечення стійкої роботи алгоритму.

Алгоритм (4.26) забезпечує нормування вектора $w^1(i)$

$$\left\| w^1(i) \right\|^2 = 1,$$

сам вектор $w'(i)$ є власним вектором матриці $R(i)$, відповідним максимальному власному значенню, а вихідний сигнал $y(i)$ характеризується максимально можливою дисперсією, тобто містить максимум інформації про багатовимірний вхідний сигнал $b_n(i)$.

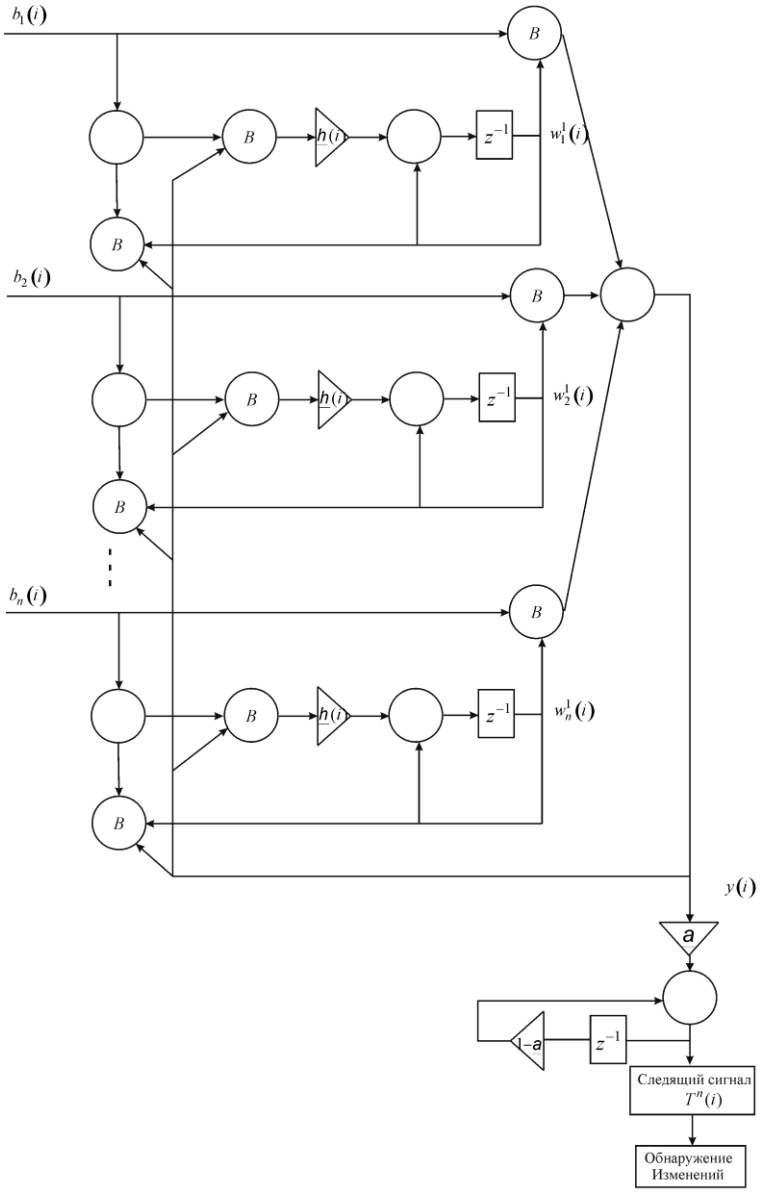


Рисунок 4.2 – Модифікований нейрон для виявлення зміни властивостей головної компоненти багатовимірного часового ряду

Далі вихідний сигнал $y(i)$ піддається експоненційному згладжуванню, фільтруючому шумові компоненти $\xi(i)$, а виявлення зміни властивостей проводиться за допомогою одновимірного стежачого сигналу $T^{TL}(i)$ (4.18), (4.19).

У результаті досліджень на реальних відеорядах отриманий наступний приклад виявлення розладнань багатовимірного часового ряду за допомогою нейронної мережі (рис. 4.3).

Як видно з представленого рисунка, коливання значень кадрів відбуваються практично постійно, однак тренд змінюється тільки в разі значних коливань, таких, як показано на наведених кадрах, що сталися на двох інтервалах відео.

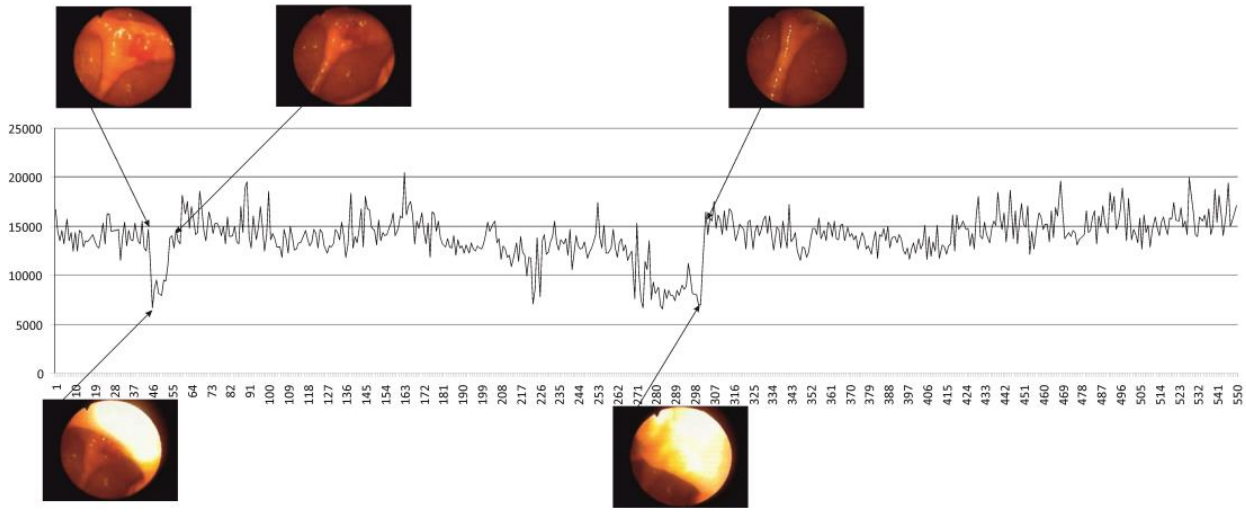


Рисунок 4.3 – Детектування змін у відеоряді

4.3 Методи прогнозування при розв’язанні задачі сегментації відео

Ще одним підходом по виділенню окремих сегментів відео на базі аналізу багатовимірних часових рядів є адаптивне прогнозування.

В основі традиційних математичних методів прогнозування (статистичних, адаптивних, нейромережевих тощо) лежать математичні моделі

того чи іншого виду, одержувані в результаті вирішення задачі структурної та параметричної ідентифікації.

Саме на основі математичних моделей вирішується завдання часової екстраполяції, при цьому в моделі явно чи неявно присутній аргумент дискретного часу. Якщо ж даних для побудови математичної моделі недостатньо, синтез прогнозуючої моделі просто неможливий. У цьому випадку замість часової екстраполяції може бути використано просторове прогнозування (екстраполяція), яке зводиться до оцінки значень векторного поля по окремих спостереженням.

Серед методів просторової екстраполяції в якості одного з найбільш перспективних слід зазначити багатовимірну лінійну екстраполяцію [158], що підтвердила свою ефективність при вирішенні ряду реальних задач проектування та управління складними багатовимірними нелінійними об’єктами.

Спочатку розглянемо метод багатовимірної лінійної екстраполяції для вирішення задачі однокрокового прогнозування n-вимірної нелінійної нестаціонарної часової послідовності стосовно описаної нами моделі. Тобто маємо

$$\bigcup_{i=1}^{N} B(i),$$

тоді нехай аналізований ряд (i-а його компонента) теоретично може бути описаний нелінійною залежністю (NARX-модель) виду

$$\begin{aligned}\hat{B}_i(k) = {} & f_i(B_i(k-1),\ldots,B_i(k-n_{A,i}),B'_1(k-1),\ldots,B'_1(k-n_C),\\ & B'_2(k-1),\ldots,B'_2(k-n_C),\ldots,B'_p(k-l),\ldots,B'_q(k-n_C)) = \\ = {} & f_i(R_{i1}(k),\ldots,R_{i,n_{A,i}}(k),\ldots,R_{i,n_{A,i}+n_C q}(k)),\end{aligned} \tag{4.27}$$

де $f_i(\circ)$ – апріорі невідома нелінійна залежність, що підлягає відновленню на підставі наявних спостережень;

$\hat{B}_i(k)$ – оцінка (прогноз) контрольованої послідовності $B_i(k)$ за даними, наявними до (k-1)-му моменту часу;

$i = 1, \ldots, n$;

$n_{A,I}$ – глибина передісторії, яка обліковується, за контрольованою послідовністю;

$B'_p(k-l)$ – p-а компонента багатовимірного екзогенного сигналу, що впливає на $B_i(k)$;

$l = 1, \ldots, n_C$;

$p = 1, \ldots, q$.

Вираз (4.27) може бути представлений і в векторно-матричній формі

$$\hat{B}(k) = F(B(k-1),\ldots,B(k-n_{A,i}),B'(k-1),\ldots,B'(k-n_C)) = F(R(k)). \qquad (4.28)$$

При наявності репрезентативної навчальної вибірки нелінійні перетворення $f_i(\circ)$ та $F(\circ)$ можуть бути відновлені в процесі навчання тієї чи іншої штучної нейронної мережі, проте якщо ця вибірка мала, нейромережевий підхід виявляється непрацездатним, в той час як багатовимірна лінійна екстраполяція дозволить отримати цілком прийнятні за точністю результати.

Задачу багатовимірної лінійної екстраполяції стосовно проблеми прогнозування багатовимірних часових рядів можна описати, використовуючи позначення (4.27), (4.28), наступним чином [158]. Нехай задана матриця прецедентів (передісторії) у вигляді

$$\mathrm{P} = \begin{pmatrix} R^T(1), & B^T(1) \\ R^T(2), & B^T(2) \\ \vdots & \\ R^T(N), & B^T(N) \end{pmatrix}$$

розмірності $N\times(nn_A+qn_C+n)$. Сама екстраполяція зводиться до знаходження в момент часу N оцінки

$$B'(N+1)=\psi(R(N+1),\mathrm{P}), \tag{4.29}$$

де $\psi(\circ)$ – власне алгоритм екстраполяції, який повинен задовільняти ряду вимог, основним з яких є те, що в результаті його використання повинні точно відновлюватися все слідства матриці прецедентів, тобто

$$\hat{B}(k)=B(k)=\psi(R(k),\mathrm{P}),\ k=1,2,...N. \tag{4.30}$$

Важливо відзначити, що методи прогнозування, засновані на тих чи інших математичних моделях, практично ніколи не забезпечують виконання умови (4.30).

З інших вимог маємо можливість відмітити те, що алгоритм $\psi(\circ)$ повинен бути векторним, тобто

$$\psi=(\psi_1,...,\psi_n),\ \ B_i=\psi_i(Z,\mathrm{P}),\ \ i=1,...,n, \tag{4.31}$$

трудомісткість реалізації повинна зростати по n та N не швидше ніж лінійно, алгоритм повинен бути працездатним при будь-яких N (навіть при N=1). Зауважимо також, що при малих N, не кажучи вже про N=1, досить точна математична модель в принципі не може бути побудована, хоча в разі аналізу відеоданих даний недолік не є настільки критичним.

Слідуючи [158], представимо алгоритм багатовимірної лінійної екстраполяції з евклідової метрикою у вигляді такої послідовності кроків:

Формування передісторії прогнозованого процесу у вигляді матриць:

– $R'=(R(1),R(2),...,R(N))$ – матриця розмірності $(nn_A+qn_C)\times N$;

– $B = (B(1), B(2), ..., B(N))$ – матриця розмірності $n \times N$.

Знаходження вектора вагових коефіцієнтів $\lambda=(\lambda_1, \lambda_2, \ldots \lambda_N)^T$, що доставляють мінімум нормі

$$\| R(N+1) - \sum_{k=1}^{N} \lambda(k) R(k) \|^2 = \left\| R'(N+1) - R'\lambda \right\|^2.$$

Формування оптимального прогнозу у вигляді лінійної комбінації

$$\hat{B}(N+1) = \sum_{k=1}^{N} \lambda_k B(k) = \bar{B}\lambda. \qquad (4.32)$$

Мінімізація виразу (4.32) може бути проведена за допомогою стандартного методу найменших квадратів, в результаті чого отримуємо відоме рішення

$$\lambda = (R'^T R')^{-1} R'^T R(N+1), \qquad (4.33)$$

яке існує тільки тоді, коли матриця $R'^T R'$ невироджена. Оскільки при малих N ($N < nn_A + qn_C$) це не так, замість операції обернення матриць пропонується використовувати операцію псевдообернення [159], в результаті чого приходимо до виразу

$$\lambda = R'^{+} R(N+1), \qquad (4.34)$$

звідки випливає, що фактично задача зводиться до знаходження ортогональної проекції вектора $R(N+1)$ на лінійну оболонку, утворену векторами передісторії R'.

З обчислювальної точки зору ніяких труднощів при реалізації даної процедури не виникає, однак, рішення задачі ускладнюється, якщо дані на

обробку надходять послідовно в реальному часі k=1, 2, …. При цьому всі попередні співвідношення можна переписати в такій формі:

$$R'_k = (R(1), R(2), ..., R(k)); \qquad \bar{B}_k = (B(1), B(2), ..., B(k));$$
$$\| R(k+1) - \sum_{l=1}^{k} \lambda_l R(l) \|^2 = \| R(k+1) - R'_k \Lambda_k \|^2; \tag{4.35}$$

$$\Lambda_k = (\lambda_1, \lambda_2, ..., \lambda_k)^T; \quad \hat{B}(k+1) = \sum_{l=1}^{k} \lambda_l B(l) = \bar{B}_k \Lambda_k ; \tag{4.36}$$

$$\Lambda_k = R'^{+}_k R(k+1). \tag{4.37}$$

В [158] для розрахунку матриці R'^{+}_{k+1} за наявною R'^{+}_k і надійшовшими на обробку $R(k+1)$ та $B(k+1)$ пропонується використовувати формулу Гревіля, хоча набагато простіше застосувати регуляризовану версію (4.33) у вигляді

$$\Lambda_k = (R'^{T}_k R'_k + \gamma I_k)^{-1} R'_k R(k+1), \tag{4.38}$$

де γ – параметр регуляції;

I – $(k \times k)$ – одинична матриця.

Для обробки нестаціонарних часових рядів, характеристики яких непередбачувано змінюються в часі, замість обробки всієї наявної вибірки, доцільно вирішувати задачу на «ковзному вікні», що складається з χ останніх спостережень.

При цьому співвідношення (4.35)-(4.37) можуть бути переписані таким чином:

$$R'_{k,\chi} = (R(k-\chi+1), R(k-\chi+2), ..., R(k));$$
$$\bar{B}_{k,\chi} = (B(k-\chi+1), B(k-\chi+2), ..., B(k)); \quad (4.39)$$
$$\| R(k+1) - \sum_{l=k-\chi+1}^{k} \lambda_l R(l) \|^2 = \| R(k+1) - R'_{k,\chi} \Lambda_{k,\chi} \|^2;$$

$$\Lambda_{k,\chi} = (\lambda_{k-\chi+1}, ..., \lambda_k)^T;$$
$$\hat{B}(k+1) = \sum_{l=k-\chi+1}^{k} \lambda_l B(l) = \bar{B}_{k,\chi} \Lambda_{k,\chi}; \quad (4.40)$$

$$\Lambda_{k,\chi} = R'^{+}_{k,\chi} R(k+1). \quad (4.41)$$

Для реалізації цієї процедури в реальному часі в [158] було запропоновано рекурентний алгоритм псевдообернення на «ковзному вікні», що відрізняється громіздкістю і обчислювальною складністю.

У зв'язку з цим доцільно замість операції рекурентного псевдообернення використовувати модифікацію (4.38) у «віконному» варіанті, при цьому його пакетна форма може бути записана у вигляді

$$\Lambda_{k,\chi} = (R'^{T}_{k,\chi} R'_{k,\chi} + \gamma I_\chi)^{-1} R'_{k,\chi} R(k+1), \quad (4.42)$$

а рекурентна [160] має такий вигляд

$$\begin{cases} \Lambda_{k+1,\chi} = \Lambda_{k,\chi} + \dfrac{\mathrm{P}(k)(R(k+1) - \Lambda^T_{k.\chi} R(k))}{1 + R^T(k)\mathrm{P}(k)R(k)} R(k); \\ \tilde{\mathrm{P}}(k-1) = \mathrm{P}(k-1) + \dfrac{\mathrm{P}(k-1)R(k-\chi)R^T(k-\chi)\mathrm{P}(k-1)}{1 - R^T(k-\chi)\mathrm{P}(k-1)R(k-\chi)}; \\ \mathrm{P}(k) = \tilde{\mathrm{P}}(k-1) - \dfrac{\tilde{\mathrm{P}}(k-1)R(k)R^T(k)\tilde{\mathrm{P}}(k-1)}{1 + R^T(k)\tilde{\mathrm{P}}(k-1)R(k)}; \\ \Lambda_{0,\chi} = 0, \mathrm{P}(k) = \gamma^{-1} I_\chi. \end{cases}$$

Власне ж прогноз обчислюється згідно зі співвідношенням (4.40).

Зрозуміло, що застосування для прогнозування співвідношень (4.40), (4.42) різко спрощує використання методу, однак залишається питання обґрунтованого вибору величини вікна, яке, як правило, задається з суто суб'єктивних міркувань, що знижує ефективність підходу в цілому.

Пропонований нижче метод багатовимірної екстраполяції заснований на використанні деякої близькості (відстані) останнього вектора передісторії $R(N+1)$ до всіх попередніх $R(1)$, …, $R(N)$ і формуванні прогнозу $\hat{B}(N+1)$ за допомогою цієї ж функції.

Реалізація методу складається з послідовності наступних кроків:

1. Розрахунок відстані між вектором $R(N+1)$ і усіма попередніми $R(k)$ на основі деякої функції близькості $d(N+1, k)$ – у найпростішому випадку евклідової метрики

$$\forall k: \quad d(N+1,k)=\left\|B'\left(N+1\right)-B'\left(k\right)\right\|.$$

2. Впорядкування (ранжування) цих відстаней в порядку зростання так, що

$$d^1(N+1,k_1) < d^2(N+1,k_2) < \ldots < d^N(N+1,k_N).$$

3. Відбір перших χ векторів, для яких виконується умова

$$d^{\chi}(N+1,k_{\chi}) \le \varepsilon,$$

де ε – деякий поріг.

4. Формування набору вагових коефіцієнтів λ_l у вигляді

$$\lambda_l = \frac{(d^l)^{-1}}{\sum_{l=}^{\chi}(d^l)^{-1}}, 1 \le l \le \chi,$$

які відповідають умові

$$\sum_{l=}^{\chi} \lambda_l = 1.$$

5. Розрахунок прогнозу

$$\hat{B}(N+1) = \sum_{l=1}^{\chi} \lambda_l R(l).$$

При надходженні нового спостереження процесу $\hat{B}(N+1)$ всі ітерації повторюються.

Таким чином, на кожному кроці у формуванні прогнозу також беруть участь χ спостережень, однак це значення може змінюватися, при цьому ясно, що чим менше χ, тим більше сигнал є нестаціонарним. Нескладно також зауважити, що

$$B(k) = \text{const} \to \chi = N,\ \lambda_l = \frac{1}{N}.$$

Даний метод не вимагає великих обсягів апріорної інформації (мала навчальна вибірка), немає необхідності вирішення додаткових завдань оптимізації або псевдообернення, вкрай простий в обчислювальній реалізації.

Однак для успішного використання всіх традиційних методів прогнозування, вихідна вибірка спостережень – часовий ряд – повинна бути досить репрезентативною. При цьому, чим більше параметрів містить прогнозуюча модель, тим більшою за обсягом повинна бути вихідна інформація. Разом з тим в реальних задачах досить часто виникає ситуація, коли ця вибірка або мала за обсягом, або прогнозований процес є нестаціонарним – містить як нерегулярні тренди, так і раптові стрибки, тому його передісторія не може бути використана для знаходження параметрів моделі.

У зв’язку з вище розглянутим пропонується підхід до синтезу адаптивних прогнозуючих моделей, який проводиться в умовах обмеженої навчальної вибірки, при цьому дані на обробку можуть подаватися через довільні заздалегідь невідомі інтервали часу, що цілком характерно для відеопослідовностей.

Для вирішення даної задачі з успіхом можуть бути використані ортогональні поліноми Чебишева [161], які мають низку корисних властивостей.

Математична модель на основі цих поліномів може бути побудована на основі малої вибірки і містить невелику кількість оцінюваних параметрів. При додаванні нового члена в модель немає необхідності перераховувати вже існуючі параметри, які, в свою чергу, можуть розраховуватися на основі звичайного методу найменших квадратів, а похибка апроксимації розподіляється рівномірно по інтервалу спостережень. Крім того, багато відомих поліномів, наприклад, Лагерра, Ерміта і ін. є окремим випадком поліномів Чебишева.

У загальному випадку формула апроксимації поліномами Чебишева може бути записана у вигляді [161]

$$Q_m(x) = \sum_{l=0}^{m} c_l f_l(B), \tag{4.43}$$

$$c_l = \frac{\sum_{k=0}^{N} y(k) f_l(B(k))}{\sum_{k=0}^{N} f_l^2(B(k))},$$

де $y(0), y(1), ..., y(N)$ – функція, що апроксимується, задана на довільних вузлах $B(0), B(1), ..., B(N)$;

$$f_0(B) = b - \beta_1,\ f_{l+1}(B) = (b - \beta_{l+1}) f_l(B) - \alpha_{l+1} f_{l-1}(B), \quad l = 0,1,...N;$$

$$\alpha_{l+1} = \frac{\sum_{k=0}^{N} B^l(k) f_l(B(k))}{\sum_{k=0}^{N} B^{l-1}(k) f_{l-1}(B(k))};$$

$$\beta_{l+1} = \frac{\sum_{k=0}^{N} B^{l+1}(k) f_l(B(0))}{\sum_{k=0}^{N} B^l(k) f_l(B(k))} - \frac{\sum_{k=0}^{N} B^l(k) f_{l-1}(B(k))}{\sum_{k=0}^{N} B^{l-1}(k) f_{l-1}(B(k))}.$$

При цьому найкращий порядок полінома m, який забезпечує необхідну точність $\sigma_m < \varepsilon$, можна визначити за формулами [161]

$$\sigma_m = \sqrt{\frac{\sigma_m^2}{N-m-1}},$$

$$\sigma_m^2 = \sigma_{m-1}^2 - c_m^2 \sum_{k=0}^{N} B^m(k) f_m(B(k)),$$

$$\sigma_0^2 = \sum_{k=0}^{N} y^2(k) - \frac{1}{N}(\sum_{k=0}^{N} y(k))^2.$$

Аргумент $B(k)$ для розв'язання завдання прогнозування можна перевести в часову шкалу, при цьому безперервний час, в якому відбувається контрольований процес, перетвориться в дискретний в формі

$$k = \frac{B(k) - B(0)}{\Delta},$$

де Δ – мінімальний такт квантування.

При цьому можна перейти від представлення полінома в формі (4.42) до виразу

$$Q_m(k) = c_0\phi_{0N}(k) + c_1\phi_{1N}(k) + ... + c_m\phi_{mN}(k), \qquad (4.43)$$

де коефіцієнти розкладання можна знайти шляхом мінімізації критерію

$$I^{\mathrm{I}}(k) = \sum_{k=0}^{N} (y(k) - Q_m(k))^2,$$

що веде до простих співвідношень [161]

$$c_l = \frac{\sum\limits_{k=0}^{N} y(k)\varphi_{lN}(k)}{\sum\limits_{k=0}^{N} \varphi_{lN}^2(k)},$$

$$c_0 = \frac{1}{N}\sum_{k=0}^{N} y(k).$$

Самі ж ортогональні функції $\phi_{lN}(k)$ відомі заздалегідь [161] і мають такий вигляд

$$\phi_{0N}(k) = 1,$$

$$\phi_{1N}(k) = 1 - 2\frac{k}{N},$$

$$\phi_{2N}(k) = 1 - 6\frac{k}{N} + 6\frac{k(k-1)}{N(N-1)},$$

$$\phi_{3N}(k) = 1 - 12\frac{k}{N} + 30\frac{k(k-1)}{N(N-1)} - 20\frac{k(k-1)(k-2)}{N(N-1)(N-2)}$$

і т. д.

Тут важливо зазначити, що в опис всіх ортогональних поліномів входить обсяг вибірки, тому наведені вище вирази повинні бути відповідним чином модифіковані для того, щоб організувати процеси адаптивної обробки і прогнозування.

У задачах, коли дані надходять на обробку послідовно, доцільно організувати процес адаптивного налаштування параметрів прогнозуючої моделі, а оскільки параметри, що підлягають оцінюванню, входять в опис (4.43) лінійно, то в цьому випадку можливо скористатися методами класичної теорії ідентифікації.

Вводячи в розгляд критерій ідентифікації

$$I^{\mathrm{I}}(k)=\sum_{k=0}^{N}\delta(k)(y(k)-\sum_{l=0}^{m}c_l\phi_{lN}(k))^2$$

(тут $\delta(k)$ – система ваг, яка визначає процес забування застарілої інформації) і мінімізуючи його за параметрами, приходимо до стандартної оцінки зважених найменших квадратів:

$$c_l(N)=\frac{\sum_{k=0}^{N}\delta(k)y(k)\phi_{lN}(k)}{\sum_{k=0}^{N}\delta(k)y_{lN}^2(k)}. \qquad (4.44)$$

Якщо далі ввести вектори

$$c=(c_0,c_1,...,c_m)^{\mathrm{T}},$$

$$\phi(k)=(\phi_{0N}(k),\phi_{1N}(k),...,\phi_{mN}(k))^{\mathrm{T}},$$

оцінку (4.20) можна переписати в векторно-матричній формі

$$c(N)=(\sum_{k=0}^{N}\delta(k)\phi(k)\phi^T(k))^{-1}\sum_{k=0}^{N}\delta(k)\phi(k)y(k) \qquad (4.45)$$

або

$$\begin{cases} c(N)=c(N-1)+\dfrac{\delta(N)\mathrm{P}(N-1)(y(N)-c^T(N-1)\phi(N))\phi(N)}{1+\delta(N)\phi^T(N)P(N-1)\phi(N)}, \\ P(N)=P(N-1)+\dfrac{\delta(N)P(N-1)\phi(N)\phi^T(N)P(N-1)}{1+\delta(N)\phi^T(N)P(N-1)\phi(N)}, \end{cases} \qquad (4.46)$$

при цьому в якості компонентів $\phi_{lN}(k)$ зручно скористатися системою ортонормованих поліномів Чебишева виду [161]:

$$\phi_{0N}(k)=\frac{1}{\sqrt{N}},$$

$$\phi_{1N}(k)=\sqrt{\frac{3}{N(N^2-1)}}(2k-N-1),$$

$$\begin{cases}\phi_{lN}(k)=(A_l k+K_l)\phi_{l-1,N}(k)-H_l\phi_{l-2,N}(k), l=2,3,...m,\\ A_l=\dfrac{2}{l}\sqrt{\dfrac{D_l(D_l-2)}{G_l}},\\ K_l=-\dfrac{N+1}{l}\sqrt{\dfrac{D_l(D_l-2)}{G_l}}=-\dfrac{N+1}{l}A_l,\\ H_l=\dfrac{l-1}{l}\sqrt{\dfrac{D_l(D_l+G_l-2)}{(D_l-4)G_l}},\\ D_l=2l+1,\\ G_l=N^2-l^2.\end{cases} \tag{4.47}$$

Тут слід зауважити, що при спільному використанні рекурентного методу найменших квадратів (4.46) і співвідношень (4.47), на кожному такті необхідно перераховувати не тільки параметри моделі $c(k)$, але і параметри поліномів A_l, K_l, H_l, D_l, G_l, які визначаються обсягом навчальної вибірки.

Зрозуміло, що в адаптивному режимі обробки це ускладнює чисельну реалізацію алгоритму.

Виключити цю незручність можна, застосовуючи в цьому випадку в якості вагової функції $\delta(k)$ в (4.45) «ковзне вікно» розміру

$$\delta(k)=\begin{cases}1 \text{ } при \text{ } N-s+1\le k\le N,\\ 0 \text{ } інакше,\end{cases}$$

що призводить до оцінки

$$c(N) = \sum_{\chi=N-S+1}^{N} \varphi(\chi)\varphi^T(\chi))^{-1} \sum_{\chi=N-s+1}^{N} \varphi(\chi)y(\chi).$$

Тоді замість рекурентного зваженого методу найменших квадратів (4.46), приходимо до адаптивної процедури виду

$$\begin{cases} \mathrm{P}(N) = \mathrm{P}_S(N-1) - \dfrac{\mathrm{P}_S(N-1)\varphi(N)\varphi^T(N)P_s(N-1)}{1+\varphi^T(N)P_s(N-1)\varphi(N)}, \\ \mathrm{P}_S(N) = P(N) + \dfrac{\mathrm{P}(N)\varphi(N-s)\varphi^T(N-s)P(N)}{1-\varphi^T(N-s)P(N)\varphi(N-s)}, \\ p_s(N) = p_s(N-1) + \varphi(N)y(N) - \varphi(N-s)y(N-s), \\ c(N) = \mathrm{P}_S(N)p_s(N). \end{cases}$$

При цьому співвідношення (4.47) можуть бути переписані в формі

$$
\begin{cases}
\varphi_{0S}(k) = \dfrac{1}{\sqrt{s}}, \\
\varphi_{1S}(k) = \sqrt{\dfrac{3}{s(s^2-1)}}(2k-s-1), \\
\varphi_{lS}(k) = (A_l k + K_l)\varphi_{l-1,s}(k) - H_l \varphi_{l-2,s}(k), l = 2,3,...m, \\
A_l = \dfrac{2}{l}\sqrt{\dfrac{D_l(D_l-2)}{G_l}}, \\
K_l = -\dfrac{1+S}{l}\sqrt{\dfrac{D_l(D_l-2)}{G_l}} = -\dfrac{1+s}{l}A_l, \\
H_l = \dfrac{l-1}{l}\sqrt{\dfrac{D_l(D_l+G_l-2)}{(D_l-4)G_l}}, \\
D_l = 2l+1, \\
G_l = s^2 - l^2, s > m.
\end{cases}
$$

Як видно, система ортонормальних поліномів може бути розрахована заздалегідь і не коригуватися в процесі налаштування адаптивної моделі.

Таким чином, задачу прогнозування коротких часових рядів з неравномірно відстоячими спостереженнями пропонується вирішувати за допомогою адаптивної моделі, яка заснована на системі поліномів Чебишева, а також алгоритму ідентифікації на ковзному вікні. Використання даної моделі дозволяє не коректувати структуру поліномів в процесі настройки параметрів моделі.

Відмінною особливістю пропонованої процедури прогнозування є те, що вона проста з точки зору чисельної реалізації, дозволяє значно скоротити час на виконання операції, а також дає можливість обробляти істотно нестаціонарні процеси, що містять як нерегулярні тренди, так і раптові скачки.

Таким чином, запропоновані моделі та методи просторово-часової сегментації відео послідовностей за рахунок аналізу VAR моделі, методу головних компонент для виявлення та відстежування змін в багатовимірному часовому ряді [162]-[166], а також прогнозуючих моделей аналізу відео [167]-[169] дозволяють виділяти однорідні за характеристиками сегменти відео

потоків з метою подальшого метричного порівняння окремих представників цих послідовностей (ключових кадрів) [170], [171], що дозволяє значно скоротити час необхідний для розв’язання задачі контекстного аналізу динамічної інформації.

5 МЕТОДИ СПІВСТАВЛЕННЯ РЕЗУЛЬТАТІВ ТЕМПОРАЛЬНОЇ ОБРОБКИ ПОТОКІВ ВІДЕОІНФОРМАЦІЇ ЗА УМОВ АПРІОРНОЇ НЕВИЗНАЧЕНОСТІ

Традиційною метою сегментації візуальної інформації є побудова розбиття відеоданих. Однак в цілому ряді програм в якості більш прийнятного і досить природного «проміжного» уявлення візуальної інформації виступають класи толерантностей, джерелом експлікації яких є поняття «подібності». Інакше кажучи, неповнота і / або недостовірність даних, недосконаліть алгоритмів кластеризації призводять до того, що набори окремих елементів (межі областей зображень, цілі відеокадри) можливо віднести до декількох сегментів-кластерів. Для подальшого інтелектуального аналізу результатів кластеризації подібне «огрубіння» поряд з деяким ускладненням, однак, створює припущення для прийняття валідних рішень.

Розділ присвячено модифікації популярних методів кластеризації для вирішення завдань сегментації як статичних зображень, так і потоків відео в умовах невизначеності про кількість можливих сегментів, їх форму, «розмитості» кордонів. Так, в теорії і практиці кластеризації поширення отримав метод k-середніх, призначений для роботи в умовах, коли кількість класів відома, а самі вони мають опуклу форму. В умовах невідомого числа класів, особливо при необхідності управління деталізацією розбиття, як вельми ефективні показали себе методи X-середніх і J-середніх, які, однак, до теперішнього часу не були адаптовані для задач обробки зображень. Для класів довільної форми були введені модифікації процедур BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) і CURE (Clustering Using Representatives), що відрізняються як формою подання вихідної інформації, так і

скороченням обсягу необхідних обчислень, що в підсумку дозволяє підвищити швидкодію процесів обробки даних.

5.1 Модифікація методу кластеризації X -середніх в задачах сегментації зображень

Застосування традиційної кластеризації, заснованої на методах k -середніх [172]-[174] та X -середніх [175]-[177] в задачах сегментації зображень обмежується низкою обставин. По-перше, обробку зображень зручніше проводити, представляючи вихідну інформацію не в формі векторів, а в вигляді послідовності фрагментів. Але в цьому випадку при використанні відомих методів k -середніх і X -середніх ці фрагменти попередньо необхідно векторизувати, а отримане рішення – девекторизувати. По-друге, стандартна процедура X -середніх, заснована на Байєсовому підході, дуже громіздка з обчислювальної точки зору. І нарешті, по-третє, апріорне припущення про гауссів розподіл даних в кластерах і їх сферичність природно обмежують можливість застосування цього підходу в задачах сегментації зображень.

Тому було запропоновано ефективну модифікацію методу X -середніх, орієнтовану на вирішення завдання сегментації зображень в умовах, коли вихідна інформація задана у вигляді наборів фрагментів-матриць, а кількість можливих класів-сегментів апріорі невідомо.

Вихідною інформацією для рішення задачі сегментації-кластеризації, як і раніше, є масив матричних спостережень $\mathrm{X} = \{x(1), x(2), \ldots x(\mathrm{N})\}$ $x(k) = \{x_{i_1 i_2}(k)\} \in R^{m \times n}$; $i_1 = 1, 2, \ldots, m$; $i_2 = 1, 2, \ldots, n$; $k = 1, 2, \ldots, \mathrm{N}$, який повинен бути розділений в процесі самонавчання на p сегментів-кластерів, описуваних

центроїдами $C = \{C(1), C(2), ..., C(p)\}$. При цьому число p кластерів апріорно невідомо, а повинно бути визначено безпосередньо в процесі обробки інформації в діапазоні $2 \le p_{min} \le p \le p_{max} \le N - 1$. Основна ідея методу X -середніх полягає в багаторазовому застосуванні до вихідного масиву даних алгоритму k-середніх з різними значеннями p і оцінюванні отриманих результатів за допомогою того чи іншого критерію, заснованого на Байєсовому оцінюванні та, перш за все, оцінках Акаіке, Каса-Вассермана, Шварца [178]-[180].

У загальному випадку процедура k-середніх складається з послідовності двох змінних етапів. При цьому на першому етапі для випадково заданого набору вихідних центроїдів $C(1),\cdots$, $C(l),... C(p)$ наявна множина всіх спостережень $x(k)$ розбивається на p груп так, що кожне спостереження $x(k)$ приписується до найближчого в сенсі відстані

$$D(x(k), C(l)) = (Sp(x(k) - C(l))(x(k) - C(l))^T)^{1/2}$$

центроїду, а на другому – відбувається перерахунок всіх центроїдів. При цьому в якості нових оцінок приймаються середні арифметичні координат всіх спостережень, приписаних до конкретного центроїду.

Ця процедура триває до тих пір, поки не стабілізуються координати всіх центроїдів.

Більш детально в цій ситуації метод k-середніх може бути описаний таким чином.

Нехай задано деяке початкове в загальному випадку довільне розбиття $P_p = \{Cl_1, Cl_1, ..., Cl_p\}$. Нехай після розрахунку відповідних центроїдів

$C = \{C(1), C(2),..., C(p)\}$ виникає ситуація, при якій деяке спостереження $x(k)$ $k = 1, 2,..., N$, яке потрапило в Cl_l, розташоване ближче до $C(q)$ ніж до $C(l)$, тобто

$$Sp(x(k) - C(q))(x(k) - C(q))^T < Sp(x(k) - C(l))(x(k) - C(l))^T .$$

Це означає, що образ $x(k)$ повинен бути переданий з Cl_l в Cl_q. В цьому випадку центроїди кластерів повинні бути відкориговані відповідно до виразів

$$C(l) = (N_l - 1)^{-1}(N_l C(l) - x(k)),$$

$$C(q) = (N_q + 1)^{-1}(N_q C(q) + x(k)).$$

Така корекція призводить до зміни цільової функції

$$\Delta E = \Delta E(x(k), C(l), C(q)) =$$

$$= \frac{N_q}{N_q + 1} Sp\left((x(k) - C(q))(x(k) - C(q))^T\right) - \frac{N_l}{N_l - 1} Sp((x(k) - C(q))(x(k) - C(q))^T).$$

Такі збільшення розраховуються на кожній ітерації алгоритму для всіх наявних спостережень всіх перелічуваних центроїдів. Якщо на якійсь ітерації з'ясується, що всі збільшення невід'ємні, покладається, що досягнуто оптимальне рішення і процес зупиняється. В іншому випадку процес корекції центроїдів триває.

Можна показати, що описана процедура мінімізує цільову функцію

$$E(x(k), C(l)) = \sum_{k=1}^{N} \sum_{l=1}^{p} \mu(x(k), C(l)) D^2(x(k), C(l)) =$$

$$= \sum_{k=1}^{N} \sum_{l=1}^{p} \mu(x(k), C(l)) Sp(x(k) - C(l))(x(k) - \mathrm{C}(l))^T,$$

де $\mu(x(k), C(l)) = \begin{cases} 1, \textit{якщо } x(k) \in Cl_l, \\ 0, \textit{інакше}; \end{cases}$

Cl_l – позначення l-го кластеру.

Координати результуючих центроїдів визначаються співвідношенням

$$C(l) = \frac{1}{N_l} \sum_{x(k) \in Cl_l}^{N} x(k) = \frac{\sum_{k=1}^{N} \mu(x(k), C(l)) x(k)}{\sum_{k=1}^{N} \mu(x(k), C(l))}, \qquad (5.1)$$

де N_l – число точок, які віднесені до l-го кластеру.

Процедура X-середніх також складається з послідовності двох етапів, один з яких називається «Поліпшення параметрів», а другий – «Поліпшення структури» [175]-[177].

При цьому під поліпшенням параметрів розуміється знаходження координат центроїдів за допомогою стандартних k-середніх при фіксованому числі кластерів p, а під поліпшенням структури – нарощування числа кластерів до досягнення необхідної якості сегментації.

Процес обробки починається з задання $p = p_{min}$ і знаходження p_{min} центроїдів за допомогою стандартного алгоритму k-середніх.

Після закінчення цього етапу проводиться оцінка отриманого результату і реалізується другий етап, який полягає в зміні кількості кластерів шляхом розщеплення вже сформованих p_{min} сегментів.

На цьому етапі використовується два можливих варіанти: розщеплення одного випадково обраного кластера на два з початковими центроїдами, що збігаються з двома довільно обраними точками з вихідного кластера, і розщеплення половини зі спочатку сформованих сегментів. Таким чином, в результаті першого варіанту утворюється $p_{min}+1$ кластерів, а в результаті другого – $1{,}5\,p_{min}$.

До знов сформованих класів застосовується та ж процедура k-середніх і проводиться оцінка якості отриманого результату.

Процес розщеплення-кластеризації триває до досягнення $p = p_{max}$. Далі з множини отриманих результатів обирається найкращий з точки зору прийнятого критерію.

Кожен з отриманих результатів з позиції байєсівського оцінювання трактується як альтернативна кластерна модель M_l, $l = 1, 2, \ldots$, p_{max}, серед набору яких повинна бути обрана найкраща, для чого пропонується використовувати критерій Шварца [180] у формі

$$BIC(M_l) = L(l) - \frac{h(l)}{2}\log N,$$

де $L(l)$ – логарифмічна функція правдоподібності для l-тої моделі;

$h(l)$ – число параметрів моделі кластера M_l.

З урахуванням того, що внутрішньо-кластерний розподіл даних є за нормальним законом і при цьому весь масив даних «розбитий» на p кластерів, можна записати вираз для внутрішньо-кластерної дисперсії наступним чином

$$\sigma_l^2 = \frac{1}{N-p} \sum_{x(k) \in Cl_l} Sp(x(k) - C(l))(x(k) - C(l))^{\mathrm{T}},$$

умовної ймовірності

$$P(x(k) \mid x(k) \in Cl_l) = \frac{N_l}{N} \frac{1}{\sqrt{2\pi}\sigma_l^{mn}} exp(-\frac{1}{2\sigma_l^2} Sp(x(k) - C(l))(x(k)C(l)))^{\mathrm{T}}),$$

логарифмічної функції правдоподібності

$$L(l) = \log \prod_{k=1}^{N} P(x(k) =$$

$$= \sum_{k=1}^{N} (\log \frac{1}{\sqrt{2\pi}\sigma_l^{mn}} \exp(-\frac{1}{2\sigma_l^2} Sp(x(k) - C(l))(x(k)C(l))^{\mathrm{T}}) + \log \frac{N_l}{N}),$$

при цьому $h(l) = p$.

Кластерна модель з максимальним значенням критерію Шварца є найкращою, а число відповідних їй кластерів – оптимальним значенням кількості сегментів в зображенні, яке обробляється.

Разом з тим, як вже зазначалося вище, істотні недоліки цього підходу пов’язані, перш за все, з досить жорсткими ймовірнісними припущеннями, які обмежують його використання в задачах обробки зображень.

Серед критеріїв якості кластеризації, що не спираються на статистичні припущення і є придатними для знаходження кількості кластерів в масиві даних, як вельми ефективний показав себе критерій Цаліньского–Харабаша [181], який може бути адаптований на випадок матричних об’єктів.

Отже, нехай задано набір двовимірних даних $\mathrm{X} = \{x(1), \ldots x(N)\}$, $x(k) = \{x_{i_1 i_2}(k)\} \in R^{m \times n}$, $k = 1, 2, \ldots, N$, який деяким чином розбитий на p кластерів з набором центроїдів $C = \{C(1), C(2), \ldots C(p)\} \subset R^{m \times n}$. Для оцінки якості такого розбиття критерій Цаліньского-Харабаша може бути записаний у вигляді

$$CH(p) = \frac{\frac{1}{p-1} SpS_B(p)}{\frac{1}{N-p} SpS_w(p)},$$

де $S_B(p) = \sum_{l=1}^{p} N_l (C(l) - \overline{C})(C(l) - \overline{C})^{\mathrm{T}}$ – матриця міжкластерного розсіювання;

$S_w(p) = \sum_{l=1}^{p} \sum_{k=1}^{N} \mu(x(k), \mathrm{C}(l))(x(k) - C(l))(x(k) - C(l))^{\mathrm{T}}$ – матриця внутрішньокластерного розсіювання;

$C(l) = \frac{1}{N_l} \sum_{x(k) \in Cl_l} x(k) = \frac{\sum_{k=1}^{N} \mu(x(k), C(l)) x(k)}{\sum_{k=1}^{N} \mu(x(k), C(l))}$ – центроїд (центр тяжіння) кластеру Cl_l;

$\overline{C} = \frac{1}{N_l} \sum_{l=1}^{p} N_l C(l)$ – матричний центр тяжіння масиву X.

Таким чином, пропонована модифікація методу X -середніх для обробки зображень на основі k -середніх і критерію Цаліньского–Харабаша може бути реалізована у вигляді послідовності етапів:

– задання $p = p_{min}$, вирішення завдання за допомогою k -середніх і розрахунок $CH(p_{min})$;

– розщеплення будь-якого з кластерів, рішення задачі за допомогою k -середніх і розрахунок $CH(p_{min}+1)$;

– при $CH(p+1) \le CH(p)$ вважається, що p є найкращою оцінкою кількості кластерів в масиві X.

Отже, запропоновано модифікований метод кластеризації X -середніх для вирішення задачі сегментації зображень.

Особливістю запропонованої модифікації є можливість обробки матричних сигналів у відсутності інформації про статистичні характеристики цих сигналів і кількості кластерів, яка автоматично визначається в процесі аналізу вихідного масиву даних.

Ситуація істотно ускладнюється, якщо в процесі обробки інформації кластери перекриваються. Така ситуація досить часто виникає саме при обробці зображень, коли межі між окремими фрагментами зображень мають нечіткий «розмитий» характер.

Інтуїтивно зрозуміло, що в основі вирішення такого завдання повинні лежати процедури нечіткої кластеризації [182], [183], однак всі подібні алгоритми працюють лише при фіксованому значенні p.

Крім того, говорити про імовірнісні закони розподілу спостережень в кластерах також не доводиться. В силу цього етап «Поліпшення структури» також має базуватися на нечіткому підході.

Розглянемо кластеризацію-сегментацію зображень в нечітких умовах, тобто при наявності класів, що перекриваються.

В основу пропонованого підходу покладена матрична модифікація методу нечітких C-середніх (FCM) [183], пов'язана з мінімізацією цільової функції самонавчання

$$E(x(k), C(l)) = \sum_{k=1}^{N} \sum_{l=1}^{p} \mu^{\beta}(x(k), C(l)) Sp(x(k) - C(l))(x(k) - C(l))^{T} \qquad (5.2)$$

при додаткових обмеженнях

$$\sum_{l=1}^{p} \mu(x(k), C(l)) = 1; \qquad (5.3)$$

$$0 < \sum_{l=1}^{p} \mu(x(k), C(l)) < N, l = 1, 2, \ldots p, \qquad (5.4)$$

де $0 \le \mu(x(k), C(l)) \le 1$ – рівні належності матричного спостереження $x(k)$ до кластеру Cl_l;

$\beta \ge 0$ – параметр «фаззіфікації», що задає рівень нечіткості меж між сусідніми сегментами, при цьому в переважній більшості прикладних задач

приймається $\beta \geq 2$, особливо при використанні евклідової (в нашому випадку сферичної) норми в якості оцінки відстані.

Рішення задачі квадратичного програмування, пов'язаної з оптимізацією цільової функції (5.2) при обмеженнях (5.3), (5.4), веде до результату

$$\begin{cases} \mu(x(k), C(l)) = \dfrac{(Sp(x(k)-C(l))(x(k)-C(l))^T)^{\frac{1}{1-\beta}}}{\sum\limits_{l=1}^{p}(Sp(x(k)-C(l))(x(k)-C(l))^T)^{\frac{1}{1-\beta}}}, \\ C(l) = \dfrac{\sum\limits_{k=1}^{N}\mu^{\beta}(x(k), C(l))x(k)}{\sum\limits_{k=1}^{N}\mu^{\beta}(x(k), C(l))}, \end{cases} \tag{5.5}$$

а при $\beta = 2$ приходимо до простого співвідношення [183]

$$\begin{cases} \mu(x(k), C(l)) = \dfrac{(Sp(x(k)-C(l))(x(k)-C(l))^T)^{-1}}{\sum\limits_{l=1}^{p}(Sp(x(k)-C(l))(x(k)-C(l))^T)^{-1}}, \\ C(l) = \dfrac{\sum\limits_{k=1}^{N}\mu^{2}(x(k), C(l))x(k)}{\sum\limits_{k=1}^{N}\mu^{2}(x(k), C(l))}. \end{cases} \tag{5.6}$$

Нескладно помітити, що (5.1) є частковим випадком співвідношень (5.5) і (5.6).

Процедура (5.5) лежить в основі етапу «Поліпшення параметрів» і дозволяє вирішувати задачу нечіткої кластеризації при фіксованому числі кластерів p.

Етап «Поліпшення структури» пов’язаний з послідовним нарощуванням кількості можливих кластерів, стартуючи з мінімального їх числа $p = p_{min} \geq 2$. При цьому, як уже зазначалося вище, використання будь-яких статистичних критеріїв для оцінки якості кластеризації в даній ситуації не є допустимим.

Далі для кожного з отриманих кластерів розраховується матриця внутрішньо-кластерного розсіювання

$$S_W(C) = \sum_{k=1}^{N} \mu^{\beta}(x(k), C(l))(x(k) - C(l))(x(k) - C(l))^T$$

і на її основі – нечітка внутрішньо-кластерная дисперсія

$$\sigma_l^2(p) = \frac{1}{N - p} Sp\, S_W(l).$$

Далі кластер з максимальною нечіткою внутрішньокластерною дисперсією $\sigma_l^2(p)$ розщеплюється на два сегменти, при цьому в якості початкових значень їх центроїдів приймаються два досить далеко віддалених один від одного спостереження, що належать цьому кластеру Cl_l. Після цього знову запускається процедура кластеризації (5.5) з $p = p_{min} + 1$.

Цей процес триває до досягнення необхідної якості сегментації, для оцінки якої в нашому випадку зручно скористатися ентропією нечіткого розбиття [182] у вигляді

$$PE(\text{p}) = -\frac{1}{N}\sum_{k=1}^{N}\sum_{l=1}^{p}\mu(x(k), C(l))\log_a \mu(x(k), C(l)),$$

(де $1 < a < \infty$), а сама ентропія може лежати в діапазоні $0 \le PE(\text{p}) \le \log_a(p)$, при цьому мінімальне значення досягається в разі чіткого розбиття даних, а максимальне – у разі, коли кожне спостереження належить всім кластерам з однаковим рівнем належності.

Необхідно відзначити, що значення ентропії залежить від заданої кількості кластерів p. Для того, щоб уникнути цього впливу, доцільно використовувати нормалізовану ентропію розбиття [181]

$$NPE(\text{p}) = -\frac{1}{N\log_2 p}\sum_{k=1}^{N}\sum_{l=1}^{p}\mu(x(k), C(l))\log_2 \mu(x(k), C(l)), \qquad (5.7)$$

що дозволяє виключити цей вплив.

Таким чином, процес нарощування кількості кластерів триває або до досягнення мінімального значення функції (5.7), або до виконання умови $p = p_{max}$.

Підводячи певний проміжний підсумок, підкреслимо, що запропонована нечітка модифікація методу кластеризації X-середніх для вирішення задачі сегментації зображень. Особливістю запропонованої модифікації є можливість обробки матричних сигналів в умовах класів, що перекриваються, і відсутність апріорної інформації про кількість формованих кластерів, яка автоматично визначається в процесі обробки інформації.

5.2 Ієрархічна агломеративна кластеризація зображень

На сьогодні найбільшого поширення отримали процедури, засновані на розбитті, які поділяють масив інформації, що містить N багатовимірних спостережень, що описуються n-вимірним вектором ознак $x(k) \in R^n$, $k = 1, 2, ..., N$, на p класів (сегментів), де p – основний параметр, що задається, як правило, апріорно з емпіричних міркувань. Такі алгоритми починають свою роботу з деякого довільного розбиття, яке в процесі оптимізації деякої також апріорі заданої цільової функції, заснованої на тій чи іншій метриці (зазвичай евклідовій або манхеттенській), безперервно коригується за допомогою тієї чи іншої ітераційної процедури.

При цьому кожне спостереження, що міститься у вихідному масиві, неодноразово проглядається.

Основною характеристикою кожного формованого кластера є його центр тяжіння, навколо якого групуються спостереження конкретного класу.

Найбільш характерними представниками цього підходу є алгоритми k-середніх, k-медоідів і т.п.

Незважаючи на популярність і досить сувору формалізацію алгоритмів розбиття, їм притаманні і суттєві недоліки. По-перше, вони формують опуклі сегменти, які в реальних зображеннях присутні далеко не завжди.

Звичайно, будь-яку неопуклу фігуру можна покрити множиною кіл досить малого радіуса. Однак при цьому, зростає обчислювальна складність алгоритму. А необхідність неодноразового перегляду кожного вектора-образу робить використання подібних алгоритмів в VLDB вкрай проблематичним.

На відміну від цього підходу ієрархічні алгоритми, які, в свою чергу,

діляться на агломеративні і дівізимні, що автоматично визначають число класів шляхом злиття окремих спостережень в кластери або дроблення вихідного масиву даних на підвибірки – кластери.

Зрозуміло, що в цьому випадку число формованих кластерів може лежати в інтервалі $2 \le p \le N-1$. Ієрархічний підхід може формувати кластери довільної форми, вкрай простий з алгоритмічної точки зору, проте в силу необхідності багаторазового перегляду всіх спостережень, що зберігаються в базі даних, вкрай незручний для обробки інформації, яка міститься в VLDB.

Крім того, одержані за допомогою цього підходу результати дуже чутливі до різного роду збурень і шумів, яки завжди присутні в реальних даних.

Інтуїтивно зрозуміло, що в цілому ряді прикладних задач найбільш простим, зрозумілим, добре інтерпретуємим і наочним є саме ієрархічний підхід, і якби вдалося істотно скоротити обсяг «переглядаємих» даних для формування стійких кластерів, його можна було б рекомендувати і для роботи з VLDB [184], [185]. Розглянемо матричний ітеративний ієрархічний балансовий метод кластеризації.

Історично першим ієрархічним агломеративним алгоритмом кластеризації, орієнтованим на роботу з VLDB, є BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) [185], обчислювальна складність якого лінійно залежить від обсягу оброблюваної вибірки, а можливість обробки даних в послідовному режимі робить його особливо привабливим.

В рамках BIRCH вводиться два основних поняття цього методу: «ознака кластеризації» (Clustering Feature – CF) і «дерево ознак кластеризації» (CF-Tree, CF-дендрограма), при цьому тут під дендрограмою розуміється вкладене угруповання об’єктів, образів, векторів ознак і т. п., що змінюється за певними

правилами на різних рівнях ієрархії.

Використання введених понять дозволяє спростити обчислення, підвищити швидкість обробки і організувати динамічну кластеризацію нових даних.

Щоб ввести ці поняття, припустимо, що l-тий кластер CL_l утворений N_l n-вимірними об'єктами-образами $x(k) \in \mathrm{C}l_l$. Для цього кластера вводиться його центроїд

$$C_l = \frac{1}{N_l} \sum_{\substack{k=1, \\ x(k) \in Cl_l}}^{N_l} x(k),$$

радіус

$$R(l) = \left(\frac{1}{N_l} \sum_{\substack{k=1, \\ x(k) \in Cl_l}}^{N_l} (x(k) - \mathrm{C}(l))^2 \right)^{\frac{1}{2}}$$

і діаметр

$$Dm(l) = \left(\frac{1}{N_l (N_l - 1)} \sum_{\substack{k=1, \\ x(k) \in Cl_l}}^{N_l} \sum_{\substack{q=1, \\ x(q) \in Cl_l}}^{N_l} (x(k) - x(q))^2 \right)^{\frac{1}{2}}.$$

Як неважко помітити, ці ознаки породжені евклідової метрикою, хоча нескладно замінити її і метрикою Мінковського.

Власне ж кластеризація на верхніх рівнях ієрархії реалізується на основі CF-векторів, що визначаються трійкою $CF(l) = (N_l, LS^T(l), SS(l))^T$, де

$$LS = \sum_{\substack{k=1, \\ x(k) \in Cl_l}}^{N_l} x(k),$$

$$SS = \sum_{\substack{k=1, \\ x(k) \in Cl_l}}^{N_l} x(k)^2.$$

Таким чином, розмірність вектора $CF(l)$ є $(n+2) \times l$. Важливо, що CF-вектори мають властивість адитивності, тобто

$$CF(l, l+1) = CF(l) + CF(l+1) = (\mathrm{N}_l + N_{l+1}, LS^T(l) + LS^T(l+1), SS(l) + SS(l+1))^T$$

і можуть уточнюватися в міру надходження нових даних.

Надалі BIRCH оперує тільки з CF, які накопичуються і аналізуються за допомогою CF-дендрограми. Сама ж CF-дендрограма характеризується двома ознаками: фактором розгалуження B, який визначає максимальну кількість елементів в кожному кластері (субкластері) на нижньому рівні ієрархії, і максимальним діаметром субкластеру T в кожному вузлі дерева (рівні ієрархії).

Зрозуміло, що чим більше значення B, тим меншу кількість кластерів буде сформовано, а чим більше T – тим менше рівнів ієрархії матиме дендрограма.

Кластеризація на основі BIRCH відбувається в два етапи: на першому етапі

в результаті однократного перегляду бази даних формується вихідна дендрограма, при цьому на нижньому рівні обробляються вихідні дані $x(k)$, а на верхніх – вектори ознак $CF(l)$.

На другому етапі проводиться кластеризація сформованих на першому етапі субкластерів, при цьому субкластери, що містять малу кількість об'єктів, видаляються як шуми і викиди, а субкластери, чиї центроїди розташовані досить близько, тобто $D(Cl_l, Cl_r) = (C(l) - \mathrm{C}(r)) < \Delta$, зливаються в один.

Таким чином, досягається працездатність процесу кластеризації, а послідовна обробка зводиться до того, що знову надійшовший образ $x(k)$ просто «вставляється» у субкластер з найближче розташованим центроїдом.

Реалізація BIRCH передбачає, що кожен оброблюваний об'єкт описується n-вимірним вектором $x(k)$, а сам процес обробки інформації пов'язаний з векторними операціями.

У ситуації, коли необхідно обробляти двовимірні зображення, вони повинні бути піддані векторизації, що веде до різкого зростання розмірності вектору, після чого власне вирішується задача кластеризації, результат рішення якої далі повинен бути девекторізован.

Процес кластеризації масивів зображень можна спростити, використовуючи замість векторних операцій відповідні матричні операції, при цьому вихідна інформація задається не в формі n-вимірних векторів, а у вигляді матриць $x(k) = \{x_{i_1, i_2}(k)\}$, $i_1 = 1, 2, ..., m$; $i_2 = 1, 2, ..., n$; $k = 1, 2, ..., N$; $x(k) \in R^{m \times n}$.

Далі можна ввести в розгляд центроїд l-го кластеру

$$C_l = \frac{1}{N_l} \sum_{\substack{k=1, \\ x(k) \in Cl_l}}^{N_l} x(k),$$

матричний радіус

$$R_l = \left(\frac{1}{N_l} \sum_{\substack{k=1, \\ x(k) \in Cl_l}}^{N_l} Sp(x(k) - C(l))(x(k) - C(l))^T \right)^{\frac{1}{2}},$$

матричний діаметр

$$Dm(l) = \left(\frac{1}{N_l N_{l-1}} \sum_{\substack{k=1, \\ x(k) \in Cl_l}}^{N_l} \sum_{\substack{q=1, \\ x(q) \in Cl_l}}^{N_l} Sp(x(k) - (x(q))(x(k) - (x(q))^T) \right)^{\frac{1}{2}}$$

і CF-матрицю розмірності $(m \times (n+1))$

$$CF(l) = \begin{pmatrix} N_l & \vdots \\ SS(l) & \vdots \\ \cdots\cdots & LS(l) \\ 0 & \vdots \end{pmatrix},$$

де $LS(l) = \sum_{\substack{k=1, \\ x(k) \in CL_l}}^{N_l} x(k);$

$$SS(l) = \sum_{\substack{k=1, \\ x(k) \in Cl_l}}^{N_l} Sp\, x(k) x^T(k).$$

При цьому в якості міри відстані використовується сферична норма

$$D_S(x(k), x(r)) = (Sp\,(x(k) - x(r))((x(k) - x(r))^T)^{\frac{1}{2}}.$$

Далі може бути реалізована традиційна BIRCH-процедура, де замість CF-вектору використовуються введені CF-матриці. Ця процедура зводиться до виконання послідовності кроків:

– формування CF-матриць і CF-дендрограм;

– попередня кластеризація;

– очищення даних від «викидів» і злиття розташованих близько підкластерів;

– остаточне формування кластерів.

Таким чином, можна говорити про те, що матрична модифікація методу BIRCH проста в чисельній реалізації, досить швидка, робастна до різного роду викидів, допускає послідовну обробку даних.

Головний недолік процедури полягає в тому, що в результаті її застосування формуються тільки опуклі кластери, що обмежує її застосовність і змушує шукати альтернативні підходи.

Перейдемо до розгляду матричного ієрархічного методу кластеризації на основі скороченої репрезентативної вибірки.

Формувати кластери не сферичної форми різних розмірів і щільності дозволяє метод CURE (Clustering Using Representatives) [184], зберігаючи при

цьому працездатність і можливість роботи з VLDB. CURE є своєрідним гібридом методів кластеризації, заснованих на розбитті, і ієрархічних агломеративних алгоритмів.

В рамках цього підходу спочатку формується фіксована множина репрезентативних точок, що характеризує кожен кластер, при цьому в межі це число може бути рівним одиниці, що ріднить CURE з популярними алгоритмами k-середніх і k-медоїдів.

При цьому використання не однієї точки-центроїда, а саме множини дозволяє формувати кластери довільної форми. Далі ці репрезентативні точки, а не вся вибірка з VLDB піддається агломеративній кластеризації.

Таким чином, різко скорочується кількість оброблюваних спостережень, що і дозволяє успішно використовувати цей метод при роботі з дуже великими масивами даних.

Робота CURE реалізується за шість послідовних етапів. На першому етапі випадковим чином формується так звана репрезентативна вибірка така, щоб вона зберігала інформацію про геометрію кластерів. Розмір такої вибірки визначається так званими межами Чернова і задається виразом

$$s \geq \mu N + \frac{N}{N_l}\log\frac{1}{\varepsilon} + \frac{N}{N_l}\sqrt{\left(\log\frac{1}{\varepsilon}\right)^2 + 2\mu\log\frac{1}{\varepsilon}},$$

де $0 \leq \mu \leq 1$.

Цей вираз дає оцінку того, що ймовірність вмісту в репрезентативній вибірці менш ніж μN_l спостережень в кластері Cl_l є меншою, ніж $0 \leq \varepsilon \leq 1$. Тут можна помітити, що популярний метод кластеризації CLARANS [185]-[187]

також оперує з випадковою підвибіркою, проте в CURE обсяг цієї підвибірки задається на основі формальних міркувань.

На другому етапі репрезентативна вибірка розбивається на h частин, тобто формується h вибірок, кожна з яких містить sh^{-1} спостережень.

На третьому етапі за допомогою будь-якого з відомих алгоритмів кластеризації проводиться незалежна предкластерізація кожної з підвибірок-сегментів репрезентативної вибірки, в результаті чого на кожному із сегментів може бути сформовано різна кількість підкластерів.

На четвертому етапі кластери, що містять малу кількість спостережень, розглядаються як шуми і викиди і виключаються з подальшого аналізу і обробки.

Саме четвертий етап забезпечує робастні властивості методу CURE.

На п’ятому етапі проводиться стандартна алгомеративна ієрархічна кластеризація всіх сформованих підкластерів, однак, оскільки на цьому етапі об’єднуються не окремі спостереження, а підкластери, що містять множину спостережень, замість відстані між окремими образами (в нашому випадку матрицями) використовуються міжкластерні відстані такі, як

$$D_{mean}(Cl_l, Cl_r) = \|C_l - C_l\| = \left(Sp\,(C_l - C_r)(C_l - C_r)^T \right)^{\frac{1}{2}},$$

$$D_{ave}(Cl_l, Cl_r) = \frac{1}{N_l N_r} \sum_{x(k)\in Cl_l} \sum_{x(r)\in Cl_r} \left(Sp\,(x(k) - x(r))(x(k) - x(r))^T \right)^{\frac{1}{2}},$$

$$D_{max}(Cl_l, Cl_r) = \max_{\substack{x(k)\in Cl_l \\ x(r)\in Cl_r}} \left(Sp(x(k) - x(r))(x(k) - x(r))^T \right)^{\frac{1}{2}},$$

$$D_{min}(Cl_l, Cl_r) = \min_{\substack{x(k)\in Cl_l \\ x(r)\in Cl_r}} \left(Sp(x(k) - x(r))(x(k) - x(r))^T \right)^{\frac{1}{2}}.$$

І, нарешті, на останньому (шостому) етапі кожне зі спостережень, що міститься в VLDB, але не належить до репрезентативної вибірки, «приписується» до одного зі сформованих кластерів за ознакою мінімальної відстані від цього спостереження до будь-якого з образів репрезентативної вибірки.

Тим самим, можна зробити висновок, що розглянуто задачу кластеризації великих масивів зображень на основі ієрархічного агломеративного підходу. Введено матричні модифікації відповідних методів, що дозволяють обробляти зображення без використання операцій векторизації-девекторізації. Пропоновані процедури кластеризації прості в чисельній реалізації, не вимагають багаторазового перегляду оброблюваних даних, забезпечують послідовну обробку інформації, що надходить, формуючи кластери довільної форми в умовах впливу інтенсивних збурювань, що в свою чергу дозволяє спростити процес співставлення результатів обробки в тому числі і відео потоків.

5.3 Матрична модифікація J-середніх в задачах сегментації зображень

Процедурою кластеризації, що володіє глобальними властивостями, є метод J-means, введений в [188]. Суть методу полягає в тому, що при «застряванні» в локальному екстремумі в його околі відбуваються «стрибки», що виводять процедуру з цього околу в області тяжіння більш «глибоких» екстремумів.

Ці скачки, по суті, є ті випадкові блукання, які виробляються в алгоритмах багатоекстремального випадкового пошуку.

Для опису евристики, пов’язаної з J-середніми, вводиться два основних поняття: «зайняті точки» – спостереження, що збігаються з центроїдами або розташовані в їх малих околах, і «окіл стрибка» – область, в якій відбуваються випадкові рухи, що виводять алгоритм кластеризації з локальних екстремумів.

Суть J-середніх полягає в тому, що при знаходженні деякого центроїда $C(l)$, в якому процедура кластеризації зупиняється, організовується процес

переміщення цього центроїда в незайняті точки, що знаходяться в його околі. Такі скачки «центр ваги – незайнята точка» проводяться до тих пір, поки не буде знайдений більш глибокий екстремум, відповідний меншому значенню цільової функції E.

З формальної точки зору суть J-середніх полягає в наступному: нехай отримано деяке поточне розбиття $P_p = \{Cl_1, Cl_2, ..., Cl_p\}$ з відповідним набором центроїдів $C = \{C(1), C(2), ..., C(p)\}$. В розгляд вводиться набір областей сусідства $H_l(P_p)$, де $l \in \{1, 2, ..., p\}$, а $H_l(P_p)$ визначає всі можливі переміщення центроїда $C(l)$ в точки $x(s)$, що належать його околу, де $s \in \{1, 2, ..., N\}$. Зрозуміло, що при таких переміщеннях постійно коригується і набір центроїдів C_s, набуваючи вигляду

$$C_s = \{C(1), ..., C(l-1), x(s), C(l+1), ..., C(p)\}.$$

Всі сусідства від $l = 1$ до $l = p$ утворюють множину сусідства поточного розбиття P_p. При цьому пошук не зупиняється, поки не буде досягнуто більш глибокий екстремум в околі будь-якого з поточних центроїдів. При цьому здійснюється мінімізація цільової функції виду

$$E(x(k), C(q), x(s) = \sum_{k=1}^{N} \Big(\sum_{q=1(q \neq l)}^{p} \mu(x(k), C(q)) Sp\,(AA^{\mathrm{T}}) - \mu(x(k), x(s)) Sp\,(BB^{\mathrm{T}}) \Big), \qquad (5.8)$$

де $A = x(k) - C(q), B = x(k) - x(s)$;

$$\mu(x(k), C(q)) = \begin{cases} 1, \textit{ якщо } x(k) \in Cl_q, \\ 0, \textit{ інакше}; \end{cases}$$

$$\mu(x(k), x(s)) = \begin{cases} 1, \textit{ якщо } x(k) \in H_l, \\ 0, \textit{ інакше.} \end{cases}$$

Роботу методу J -середніх зручно записати у вигляді послідовності кроків.

1. Задання початкового, досить довільного, розбиття $P_p = \{Cl_1, Cl_2, ..., Cl_p\}$, визначення його центрів $C(1), C(2), ..., C(p)$, і розрахунок цільової функції $E(x(k), C(l))$, що відповідає цьому розбиттю.

2. Визначення всіх незайнятих точок, тобто спостережень, які не збігаються з центроїдами або які не перебувають в їх безпосередній близькості.

3. Стрибки в околах центроїдів, тобто їх заміна незайнятими точками $x(s)$ з розрахунком цільових функцій $E(x(k), C(q), x(s))$ виду (5.8).

4. Прийняття рішень про зупинення або подальший пошуку. Якщо

$$E(x(k), C(q), x(s)) > E(x(k), C(q)),$$

то вважається, що оптимальне рішення було знайдено на попередніх ітераціях, і процес кластеризації зупиняється. В іншому випадку вибирається рішення, відповідне найменшим значенням цільової функції $E(x(k), C(q), x(s))$ для всіх реалізованих $x(s)$. Отримане розбиття покладається початковим і проводиться повернення до кроку 2.

Головна особливість J -середніх, що відрізняє його від всіх методів кластеризації з використанням центроїдів, – це наявність кроку 3 (скачки в околах), вдала реалізація якого визначає ефективність отриманого рішення в цілому. Якість кластеризації може бути покращена, якщо крок 3 реалізувати у формі послідовності підкроків.

3а. Формування нового центроїда $C(p+1)$ в незайнятій точці $x(s)$ і

знаходження «найкращого» центроїда $C(l)$, що дає найбільше значення цільової функції $E(x(k), C(q), x(s))$.

3б. Видалення центроїда $C(l)$ і заміна його на $x(s)$.

3в. Формування нового розбиття P_p^{new}, в якому $C^{new}(l) \equiv x(s)$.

Можна відзначити [188], що в ряді випадків якість одержуваного рішення може бути покращена, якщо після кроку 3 включити стандартну процедуру k- або H-means. Автори статті [188] назвали цей метод J-means+.

Найбільші проблеми, пов'язані з практичним використанням методу J-середніх, виникають при заданні розмірів (діаметра) околу стрибка навколо кожного центроїду.

Занадто малий окіл може не виявити незайняті точки з меншими значеннями цільової функції, а надто великий окіл різко збільшує обчислювальну складність алгоритму, особливо у випадках, коли ці околи перекриваються.

У зв'язку з цим доцільно ввести в розгляд околи змінних розмірів, при цьому процес кластеризації починається в малих околах і якщо в них поліпшення не відбувається, їх розмір починає збільшуватися, захоплюючи нові незайняті точки. В цьому випадку вільним параметром алгоритму є $S_{\max}$ – число, яке визначає скільки разів окіл може збільшуватися, «захоплюючи» більше незайнятих точок. Роботу методу J-середніх з околами, що розширюються, можна записати у вигляді послідовності кроків.

1. Задання початкового розбиття $P_p = \{Cl_1, Cl_2, ..., Cl_p\}$, визначення його центроїдів $C = \{C(1), C(2), ..., C(p)\}$, розрахунок цільової функції $E(x(k), C(l))$, що відповідає цьому розбиттю, вибір умов зупинки і значення параметра $S_{\max}$.

2. Перевірка умов зупинки. Якщо умови зупинки виконуються (в сенсі значень цільової функції $E(x(k), C(l))$), то процес кластеризації закінчується.

3. Розрахунки в околі мінімального діаметра. Покласти $S = 1 \le S_{\max}$.

4. Якщо $S > S_{\max}$, повернутися до кроку 2.

5. Вибрати випадковим чином незайняту точку $x(r)$ в околі заданого діаметра і провести розбиття, використовуючи це спостереження замість центроїду $C(1)$.

6. Використовувати J-means метод, використовуючи в якості початкового розбиття набір центроїдів $C_S = \{C(1),..., C(l-1), x(s), C(l+1),..., C(p)\}$.

7. Якщо забезпечено менше значення цільової функції - закінчити роботу алгоритму. Інакше покласти $S = S + 1$ і йти до кроку 4.

При роботі даного алгоритму додатково в якості умов зупинки можуть бути використані також час роботи процедури, максимальне число ітерацій алгоритму між поліпшенням цільової функції, обмеження на число розширень околів сусідства.

Основною евристикою описаних вище процедур є переміщення центроїдів в незайняті точки і знаходження на цій основі розбиття з кращими значеннями цільової функції. Основною ж проблемою є необхідність багаторазового повного перебору всіх сполучень спостережень і розрахованих центроїдів. Зрозуміло, що при великих N і p підхід, заснований на J-середніх, виявляється неефективним.

У зв’язку з цим доцільно ввести в розгляд матричну модифікацію методу відомого як «Швидкі J-середні» (Fast J-means). Основна мета цієї модифікації – зменшення обчислювальної складності алгоритму J-means. При цьому пошук

проводиться в околах не всіх центроїдів $C(l)$, $l=1,2,...,p$, а тільки в одному єдиному кластері, для вибору якого може бути використана або мінімальна внутрішньокластерна відстань

$$\bar{D}^2 = \frac{1}{N_l} \sum_{x(k)\in Cl_l} Sp(x(k)-\mathrm{C}(l))(x(k)-C(l))^T$$

або максимальна міжкластерна відстань

$$\bar{\bar{D}}^2 = \frac{1}{p} \sum_{x(k)\in Cl_l} Sp\,(\mathrm{C}(q)-\mathrm{C}(l))(\mathrm{C}(q)-C(l))^T\ .$$

Зрозуміло, що найбільш віддалений від інших кластер максимального діаметру має більше за інших шансів містити більш глибокий екстремум. Зрозуміло також, що Fast J-means з точки зору обчислювальної реалізації в p раз простіше стандартного методу J-середніх.

Роблячи проміжний висновок, слід вказати, що запропонована матрична модифікація методу J-середніх, призначена для вирішення завдань сегментації цифрових зображень. Особливістю введеної модифікації є можливість обробки фрагментів зображень без їх попередньої векторизації, введення околів пошуку, що розширюються, а також організація цього пошуку в єдиному околі, де з найбільшою ймовірністю може міститися рішення. У порівнянні зі стандартним методом кластеризації J-середніх пропонована модифікація є простішою з обчислювальної точки зору і вимагає істотно менших обчислювальних витрат. У той же час перетинання класів поки неможливо. Перейдемо до розгляду

сегментації зображень на основі методу нечітких *J*-середніх.

Суть методу нечітких *J*-середніх (fuzzy J-means, FJM) полягає в тому, що при застряванні процесу оптимізації в локальному екстремумі, отриманий центроїд починає деяким досить випадковим чином переміщатися в незайняті (що не збігаються з центроїдами) точки, що знаходяться в найближчому його околі аж до досягнення більш «глибокого» екстремуму – центроїду.

Далі отримане чітке рішення переозначується в нечітке рішення шляхом розрахунку рівнів належності та уточнення центроїдів всіх кластерів.

Реалізація FJM методу проводиться в два етапи: знаходження локальних оптимумів за допомогою стандартного FCM і знаходження більш «глибоких» мінімумів за допомогою FJM-евристики.

На першому етапі реалізується послідовність кроків.

1. Завдання початкового досить довільного розбиття $P_p = \{Cl_1, Cl_2, ..., Cl_p\}$ з центроїдами $C(1)$, $C(2),...,$ $C(\text{p})$, фаззіфікатора β і порогового значення $\varepsilon > 0$, що визначає умови зупинки алгоритму.

2. Розрахунок рівнів належності $\mu(x(k), C(l))$ за допомогою першого співвідношення (5.5) (для довільного β) або першого співвідношення (5.6) (для $\beta = 2$) з центроїдами, отриманими на попередньому етапі.

3. Перерахунок центроїдів $C(1)$, $C(2),...,$ $C(p)$ за допомогою другого співвідношення (5.5) (для довільного β) або другого співвідношення (3.6) (для $\beta = 2$) з рівнями належності, отриманими на попередньому етапі.

4. Оцінка сферичної норми різниці між раніше отриманими центроїдами і центроїдами, розрахованими на третьому кроці.

5. Перевірка умов зупинки. Якщо отримана норма є меншою за ε, то

алгоритм закінчую роботу; якщо ж отримана норма є більшою за ε, то повертаємося до кроку 2 з центроїдами, отриманими на кроці 3.

Ітерації тривають до виконання умови зупинки, а отримане рішення є координатами локального оптимуму задачі нелінійного програмування (5.2) – (5.4).

Другий етап – це фаза стрибків, коли з отриманого локального мінімуму відбуваються випадкові рухи в його околі з метою відшукати більш глибокий екстремум.

В [189]-[190] було показано, що загальна задача оптимізації (5.2) при наявності обмежень (5.3), (5.4) може бути зведена до задачі безумовної оптимізації спеціального виду цільової функції, яка в матричному випадку може бути записана у вигляді

$$E = E(x(k), C(l)) = \sum_{k=1}^{N} (\sum_{l=1}^{p} (Sp(x(k) - C(l))(x(k) - C(l))^T)^{1-\beta})^{1-\beta} \qquad (5.9)$$

для довільних значень фаззіфікатора β та

$$E = E(x(k), C(l)) = \sum_{k=1}^{N} (\sum_{l=1}^{p} (Sp(x(k) - C(l))(x(k) - C(l))^T)^{-1})^{-1} \qquad (5.10)$$

для $\beta = 2$.

Далі з будь-якого з отриманих центроїдів $C(l)$, $l = 1, 2..., p$, відбуваються скачки, тобто обраний центроид замінюється будь-яким спостереженням $x(r)$, після чого розраховується значення цільової функції (5.9) або (5.10) у формі

$$E = E(x(k), x(r)) = \sum_{k=1}^{N} (\sum_{l=1}^{p} (Sp(x(k) - x(r))(x(k) - x(r))^T)^{1-\beta})^{1-\beta} \qquad (5.11)$$

або

$$E = E(x(k), x(r)) = \sum_{k=1}^{N} (\sum_{l=1}^{p} (Sp(x(k) - x(r))(x(k) - x(r))^T)^{-1})^{-1}. \quad (5.12)$$

Якщо виявиться, що для деякого $x(r)$ значення (5.11), (5.12) виявляться меншими ніж (5.9), (5.10), приймається рішення про те, що знайдено новий покращений центроїд кластеру Cl_l, після чого проводиться перерахунок всіх рівнів належності за допомогою першого співвідношення з (5.5) або (5.6). Такі скачки відбуваються в околі кожного з центроїдів, отриманих на першому етапі. Якщо виявиться, що скачки в околах всіх центроїдів не привели до поліпшення значення цільової функції (5.11), (5.12), то або приймається рішення про закінчення процесу оптимізації, або виробляються скачки в околах збільшеного радіусу.

В принципі такий процес може тривати до вичерпання всіх незайнятих точок. Знайдений останнім локальний екстремум покладається глобальним.

Незважаючи на гадану громіздкість, описаний процес оптимізації є досить простим в обчислювальної реалізації.

Отже, загалом запропоновані матричні модифікації методів X -середніх, *J*-середніх та нечітких *J*-середніх, призначені для вирішення задачі сегментації зображень в умовах невизначеної кількості класів, що можливо перетинаються. Зазначені методи дозволяють порівнювати результати темпоральної сегментації

відеопослідовностей, що дозволяє скоротити необхідний час на подальшу обробку динамічних даних, що є особливо важливим при роботі в онлайн або в реальному часі.

6 ЕКСПЕРИМЕНТАЛЬНИЙ АНАЛІЗ НЕЧІТКОЇ ВІДЕОІНФОРМАЦІЇ ЗА УМОВ НЕСТАЦІОНАРНОСТІ ТА НЕВИЗНАЧЕНОСТІ ЩОДО КІЛЬКОСТІ ТА ФОРМИ КЛАСІВ

Розділ присвячено експериментальному аналізу методів сегментації / кластеризації відеоданих, заснованих на уявленні вихідних даних у вигляді багатовимірного часового ряду з подальшим аналізом за допомогою методів, запропонованих в розділах 4-5. Стосовно статичної візуальної інформації, сегментація трактується традиційно: пошук в полі зору областей однорідності, що корелюють з об’єктами сцен. В аспекті обробки відеопослідовностей під сегментацією розуміється побудова розбиття (покриття) відео таким чином, щоб набори зображень з одного кластера-сегмента були в ідеалі однорідні за змістом, на практиці – за характеристиками відеокадрів, зокрема, по «просторовому змісту», який визначається класичної сегментацією. Запропоновані методи сегментації орієнтовані, в першу чергу, на використання в системах CBVIR, тобто на структуризацію даних, що призводить і до можливості, і до необхідності широкої варіації розмірів фрагментів.

6.1 Аналіз методів просторово-часової сегментації відеопослідовностей

Для експериментального аналізу запропонованих підходів по сегментації відео використовувалися відеофрагменти спортивного змісту тривалістю 1500 кадрів або іншими словами тривалістю 1 хвилина кожен, які включали в себе:

1. Відео фрагменти етапу велогонки Тур де Франс.
2. Відео фрагмент польотів літаків на авіашоу.
3. Фрагмент гонки етапу Формули 1.
4. Фрагмент змагань з вітрильного спорту.

На рисунку 6.1 представлені 11-ті кадри кожної з цих відеопослідовностей і їх просторові сегментації для ілюстрації вмісту багатовимірних рядів.

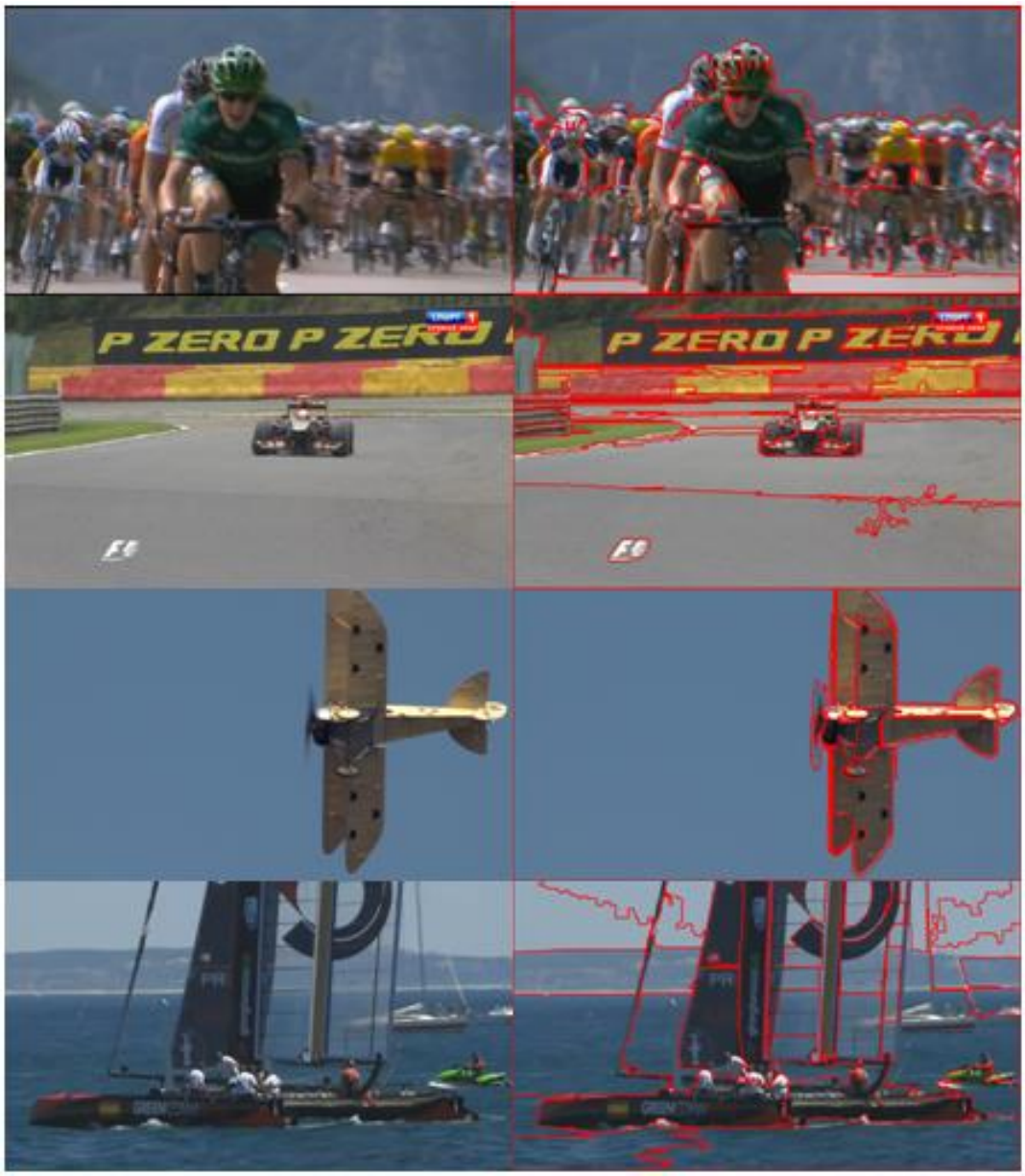


Рисунок 6.1 – Кадри тестових відеопослідовностей

При цьому треба розуміти, що вид одержуваних одновимірних рядів коливається в залежності від обраної моделі. На рисунках 6.2-6.4 наведені приклади аналізованих одновимірних рядів однієї і тієї ж відеопослідовності тривалістю 550 кадрів, на підставі єдиної просторової сегментації і при аналогічному виборі вектора характеристик. Так, на рисунку 6.2 наведено приклад, отриманий на підставі VAR-моделі. На наступній діаграмі (рис. 6.3) наведено результат роботи нейронної мережі. Як можна побачити при порівнянні, в цьому випадку модель більш чутлива до змін, що виражається в

більш стрибкоподібних змінах ряду. Хоча загальний тренд зберігається в обох випадках. Цілком логічним виглядає те, що на більш коротких вибірках кращим буде виглядати другий варіант, тому що при цьому знижується ймовірність пропустити зміни, що виникають. Тобто можна сказати, що даний підхід можливо застосувати для більш «тонкого» аналізу.

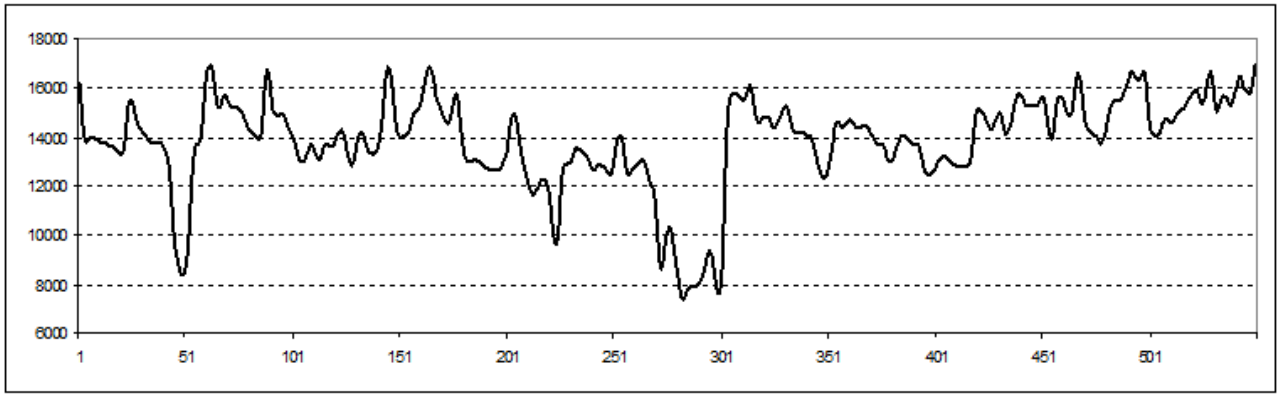


Рисунок 6.2 – Приклад одновимірного ряду, отриманого за допомогоюVAR-моделі

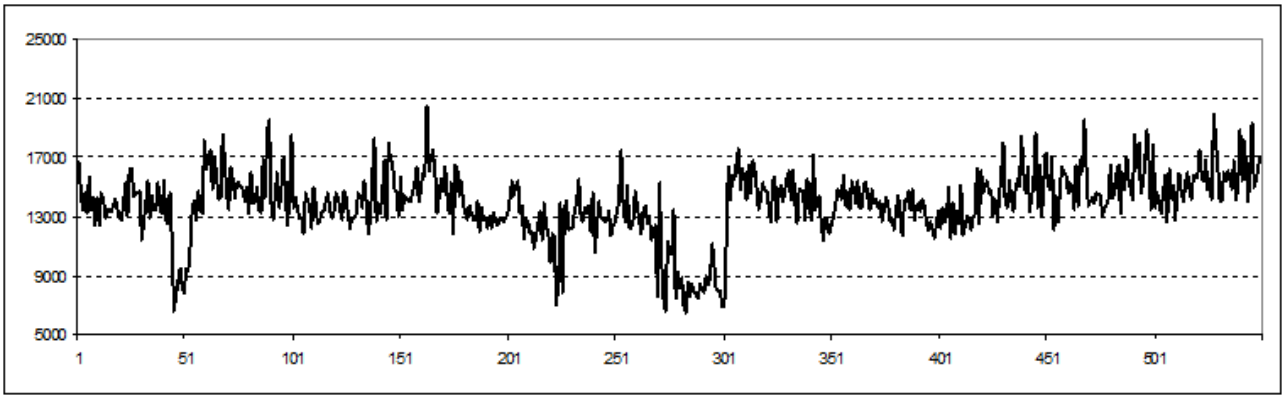


Рисунок 6.3 – Приклад одновимірного ряду, отриманого за допомогою нейронної мережі

Якщо ж нас цікавлять зміни, що відбуваються на великих інтервалах, то перша модель є кращою, тому що в цьому випадку відображаються в основному «грубі» зміни без урахування можливих викидів значень на малих інтервалах

часу. На рисунку 6.4 розглянуто модель, що налаштовується для тієї ж відеопослідовності. В цьому випадку якщо зміни відсутні, то лінія графіка прагне до значення 1. При виникненні ж розладнань в часому ряді відбуваються аномальні викиди значень. При цьому на наведеній ілюстрації чітко спостерігається один з недоліків даного підходу, що виражається в тому, що по суті зміни реєструються з запізнюванням в кілька кадрів, яка необхідна для переналаштування моделі. Цей момент чітко спостерігається в перших декількох кадрах відеопослідовності, коли вибірка даних мінімальна і налаштування моделі ускладнюється. Якщо виключити з розгляду всього 2 початкових значення, то інформативність представлення результату значно поліпшується, як показано на рис. 6.5.

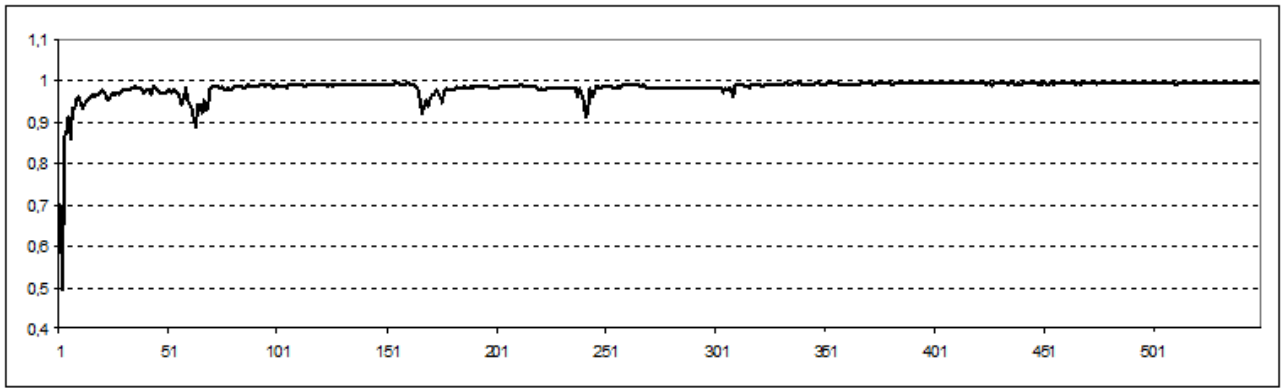


Рисунок 6.4 – Приклад одновимірного ряду, отриманого за допомогою моделі, що налаштовується

Якщо порівняти всі три підходи, то ми бачимо, що в кожному випадку реєструються розладнання в відео на певних інтервалах. По-перше, це інтервал від 40 по 70 кадри, а також неодноразові коливання значень характеристик на інтервалі від 150 до 300 кадру, на якому зареєстровано як максимальне, так і мінімальне значення параметрів для перших двох підходів, і є чітке стабільне відхилення у випадку моделі, що налаштовується.

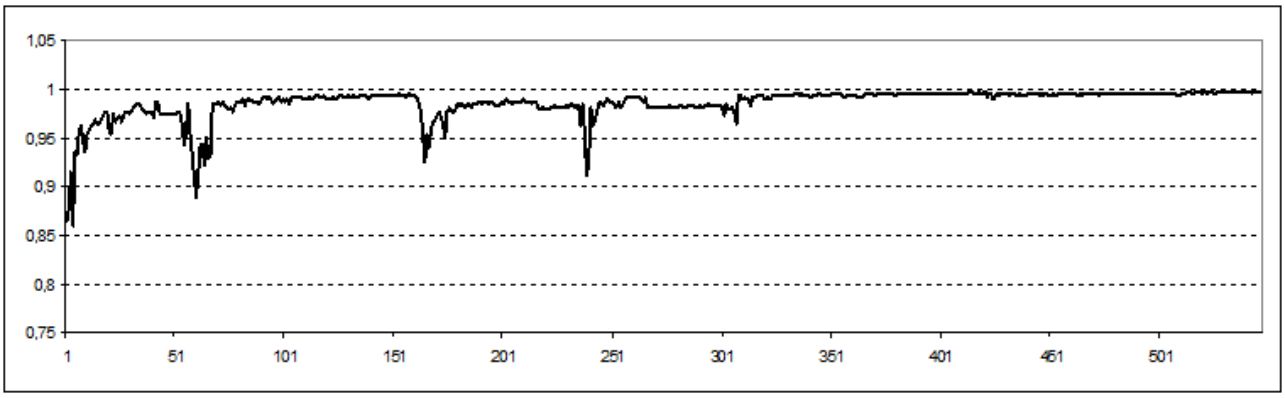


Рисунок 6.5 – Приклад одновимірного ряду, отриманого за допомогою моделі, що налаштовується

Виходячи з наведених експериментальних результатів роботи методів сегментації відеопослідовностей на однакових вхідних даних, можна стверджувати, що кожен з цих підходів реєструє зміни, що відбуваються в відеопослідовності, що в свою чергу дозволяє виділити межі сегментів, і відповідно вирішити задачу сегментації, тобто розбиття всієї послідовності даних на однорідні сегменти.

Однак залишається відкритим питання вибору вектора характеристик з описаних в розділі 3 для найбільш якісного розбиття вихідних відеопослідовностей. На рисунку 6.6 наведено 3 різних варіанти вибору вектора параметрів. При цьому вони були пронормовані з тим, щоб значення можна було порівняти. Як видно з результатів, перший і другий варіанти векторів характеристик дають неоднозначні результати, за якими складно визначити наявність розладнань в послідовності відеоданих. Третій же варіант дає схожі результати з тим вектором, який був обраний нами для проведення експериментів. Слід зазначити, що в першому і другому випадку, на відміну від третього варіанту, як однією з неврахованих при розгляді ознак була обрана площа сегментів просторової сегментації кадрів, виходячи з чого можна стверджувати, що дана характеристика є однією з необхідних для обчислень при побудові характеристичного вектора кадрів відеопослідовності.

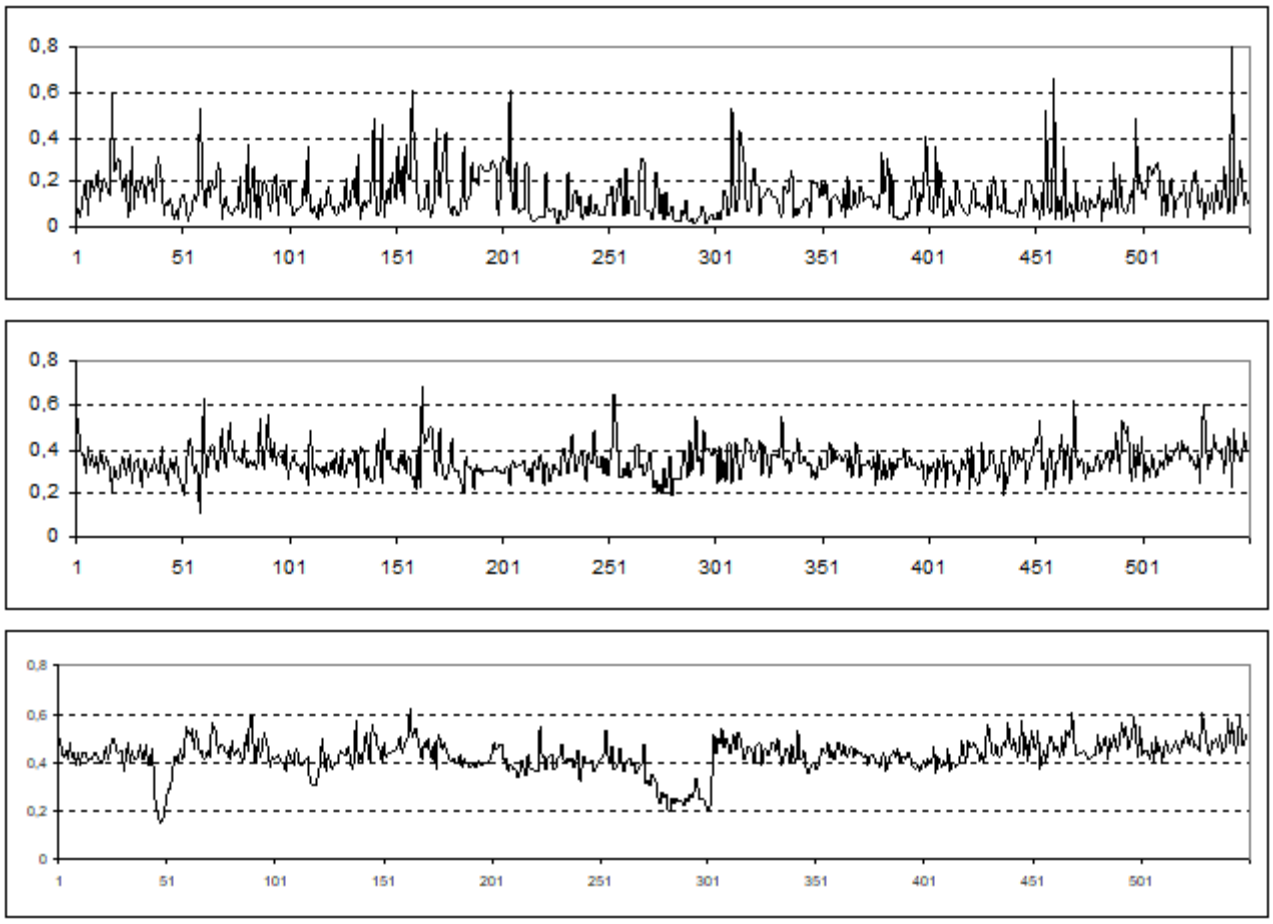


Рисунок 6.6 – Залежність результатів аналізу відео від вибору вектора характеристик

З огляду на той факт, що обробка відеопослідовностей є досить ресурсомісткою процедурою і вимагає до того ж значних витрат часу, цілком логічним виглядають спроби якимось чином полегшити витрати за часом і ресурсами. Одним з інтуїтивно зрозумілих підходів є проріджування, тобто розгляд не кожного кадру відеопослідовності, а через певні проміжки. Таким чином, з одного боку можна значно зменшити час, необхідний на обробку даних, зокрема один з найбільш витратних за часом етапів – просторову сегментацію кадрів. З іншого боку, в результаті такого проріджування відеопослідовності можуть статися втрати інформації про зміни структури часового ряду. На рис. 6.7 наведено приклад використання проріджування на тому ж відео фрагменті. При цьому спостерігаються як позитивні моменти, пов’язані з меншим випадковим розкидом даних в сусідніх кадрах, так і негативні, в тій ситуації,

коли кадр обраний від початку виступав аномальним викидом. Наприклад, зміни, що відбувалися в районі 300 кадру стали менш явними у порівнянні з аналізом всієї послідовності.

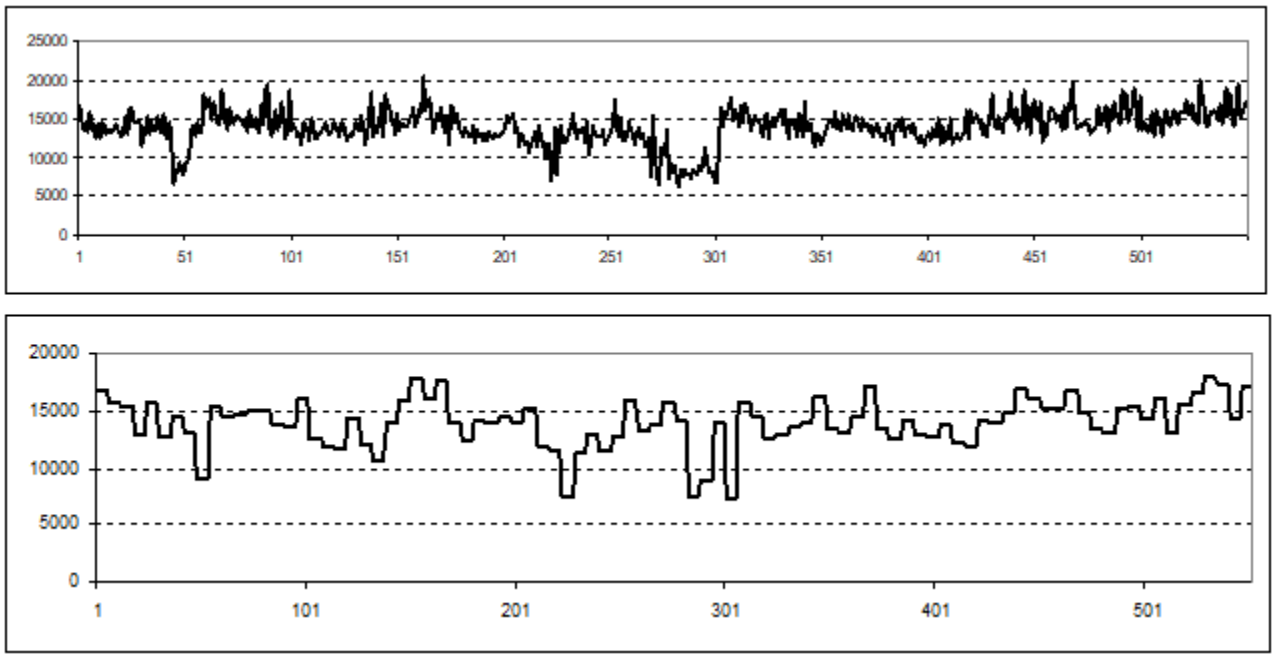


Рисунок 6.7 – Використання всієї вихідної послідовності і кожного 5-го кадру

Далі розглянемо результати сегментації тестових відеопослідовностей. При візуальному розгляді кожних вихідних даних експерти легко відзначають переходи між сценами. На рисунку 6.8 наведено приклад переходу від одного сегмента до іншого для кожної з тестових відеопослідовностей. При цьому іноді виникають ситуації з неявними переходами, які можуть призводити до проблем при детектуванні змін параметрів тимчасового ряду. Така ситуація може виникати при наявності різних ефектів в відеоданих, які покликані згладити візуальні переходи для глядача, але є швидше негативним фактором при математичному аналізі подібних послідовностей. Приклад подібного переходу показаний на прикладі кадрів з відеопослідовності у фрагменті етапу гонок Формули 1. На рис. 6.9 показані 3 кадри з послідовності переходу від одного сегмента до іншого, де на кожному з кадрів присутні елементи 2 різних сегментів відеоданих. Цілком логічно, що виявлення подібних переходів ускладнюється саме «завдяки» наявності характеристик, що належать різним групам даних.

Рисунок 6.8 – Приклади переходів між сегментами в початкових відеопослідовністях

Слід відзначити той факт, що в залежності від складності фону ми можемо отримувати більш явні показники переходів між сегментами. Так, для зйомок авіашоу, де в якості фону виступає небо, перехід від одного сегмента відео до іншого виділяється більш чітко (рис. 6.10), ніж для послідовності, яка відображає події етапу велогонки, на якому часто існує ситуація в якій швидше створюється враження руху не об’єкта, а фону, що досягається з огляду на те, що камера не є стаціонарною і основна її увага спрямована на спортсменів.

Рисунок 6.9 – Приклад кадрів з елементами різних сегментів

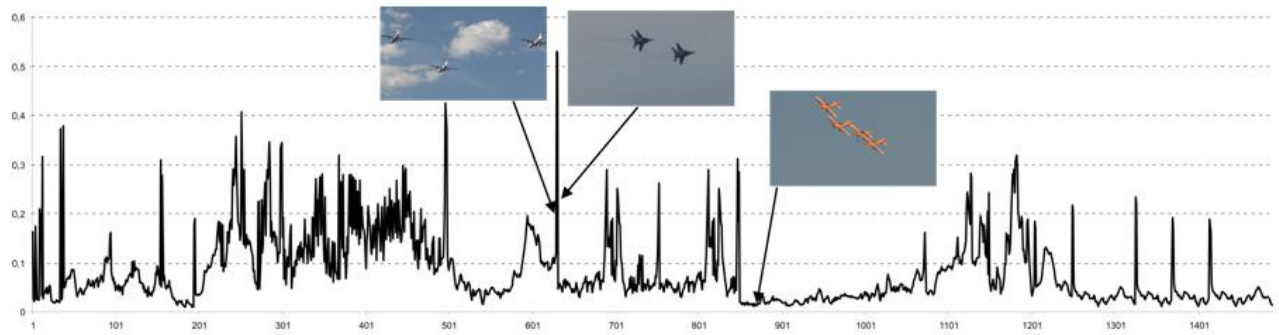


Рисунок 6.10 – Сегментація відеопослідовності «Авіашоу»

При цьому фон є далеко не однорідним, на ньому можуть бути присутніми свої рухомі елементи, наприклад, вболівальники, що також впливає на детектування меж сегментів. Сегментація даної тестової відеопослідовності представлена на рисунку 6.11. З наведеного графіка часового ряду створюється враження, що зміни структури відбуваються практично постійно, але насправді деякі зі сплесків є помилковими, тому для подібного роду послідовностей швидше підійде підхід, який менш різко реагує на найменші зміни структури часового ряду. Зокрема, замість нейронної мережі можна було б використовувати VAR модель подібно до того, як це було зроблено на рис. 6.2.

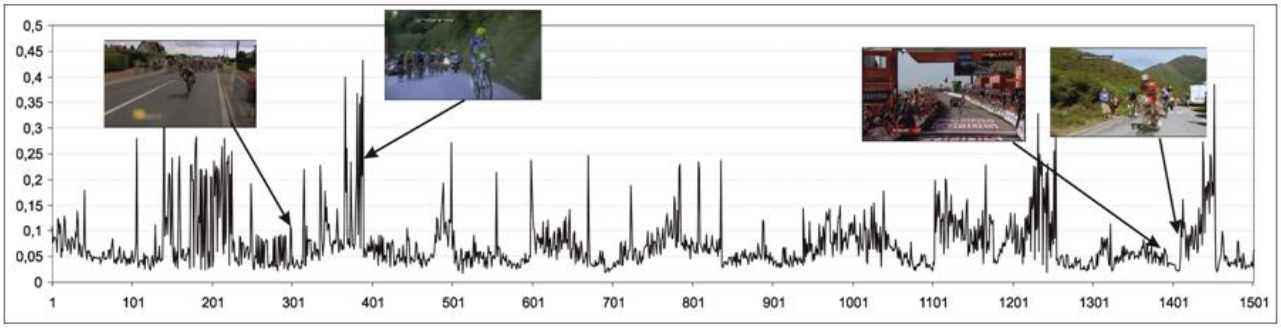


Рисунок 6.11 – Сегментація відеопослідовності «Тур де Франс»

Слід зазначити, що в розглянутих сюжетах «Авіашоу» і «Тур де Франс» в інтерактивному режимі стійко виділяються 19 і 16 сегментів відповідно. Для відеопотоку «Авіашоу» маємо такі сегменти: [1, 19], [20, 80], [81, 142], [143, 199], [200, 280], [281, 311], [312, 354], [355, 389], [390, 433], [434, 520], [521, 603], [604, 651], [652, 725], [726, 807], [808, 875], [876, 951], [952, 1237], [1238, 1414], [1415,

1500]. Для відеопотоку «Тур де Франс» – [1, 316], [317, 457], [458, 538], [539, 576], [577, 624], [625, 679], [680, 734], [735, 823], [824, 869], [870, 924], [925, 967], [968, 1007], [1008, 1063], [1064, 1114], [1115, 1148], [1149, 1500].

До цих пір ми будували сегментації відеопослідовностей на основі однієї і тієї ж просторової сегментації кадрів, в якості якої був обраний алгоритм текстурної сегментації JSeg. Однак, як і у більшості інших методів автоматичної сегментації зображень, він може виконуватися з різними параметрами, які будуть впливати на кількість і характеристики сегментів, на які буде проводитися розбиття окремих кадрів.

Для того щоб з'ясувати, наскільки впливає зміна параметрів сегментації зображень на результат сегментації відеопослідовностей, для тестових відеопослідовностей були побудовані часові ряди на підставі нейронної мережі, в яку подавалися в якості вихідних даних багатовимірні характеристичні вектори, отримані на підставі різних сегментації кадрів. В результаті на рисунку 6.12 представлено 4 варіанти розкладання відеопослідовності «Авіашоу» з різними параметрами сегментації алгоритмом JSeg. Як видно з отриманих послідовностей, різні параметри сегментації методом JSeg впливають тільки на загальні значення характеристик кадру, тобто числові значення рядів різні. Але в той же час вплив на загальний тренд ряду був незначним.

Таким чином, в результаті аналізу відео спортивної тематики можна зробити попередній висновок про доцільність використання моделей аналізу багатовимірних часових рядів, описаних в розділі 4, з метою просторово-часової сегментації відео.

До особливостей спортивного відео варто віднести той факт, що зйомки подібних роликів зазвичай проводяться безліччю відеокамер в онлайн режимі з постійною зміною ракурсу телетрансляції для виділення найбільш цікавою для глядача інформації.

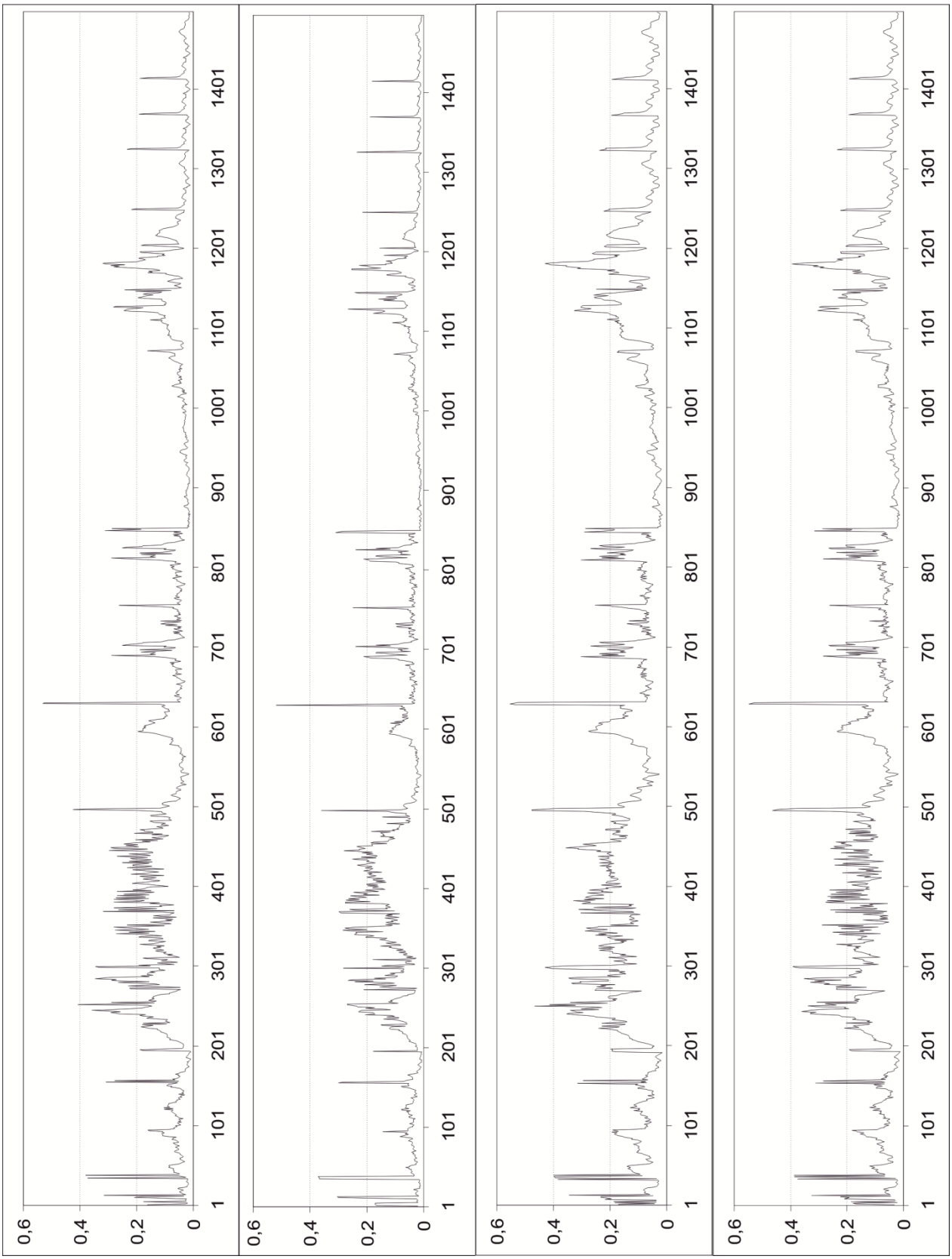


Рисунок 6.12 – Залежність сегментації відео від параметрів сегментації окремих кадрів в ролику «Авіашоу»

В результаті в рамках одного відеоролика можливі абсолютно різні за якістю зйомки сюжети, що може бути пов’язано як з різним місцезнаходженням камер щодо епіцентру подій, що призводить до зображень одних і тих же дій в різному ракурсі і масштабі (наприклад, відео з бортовою камери боліда формули 1 і камери розміщеної за межами треку), так і можливим нестаціонарним розміщенням відеокамери (наприклад, камера на мотоциклі під час велогонок). Абсолютно іншими особливостями володіють художні фільми, в яких зміна сцени диктується сюжетом і жодним чином не пов’язана зі змінами, які відбуваються в реальному часі. Відповідно в таких випадках можливий як факт руху об’єкта в полі зору з фіксованим фоном, так і зміна фону, наприклад, видалення або наближення з фактично нерухомим об’єктом.

Для експериментальної перевірки сегментації даних подібного виду було обрано ролик з кінофільму «Винищувачі», тому що за своїм контекстного змісту його можна порівняти з роликом «Авіашоу». Тривалість ролику також була 1 хвилина. Приклад кадрів і їх сегментації представлений на рисунку 6.13. На рисунку 6.14 наведені результати тимчасової сегментації, аналогічно сегментації в ролику «Авіашоу».

Рисунок 6.13 – Кадри фільму і їх сегментації

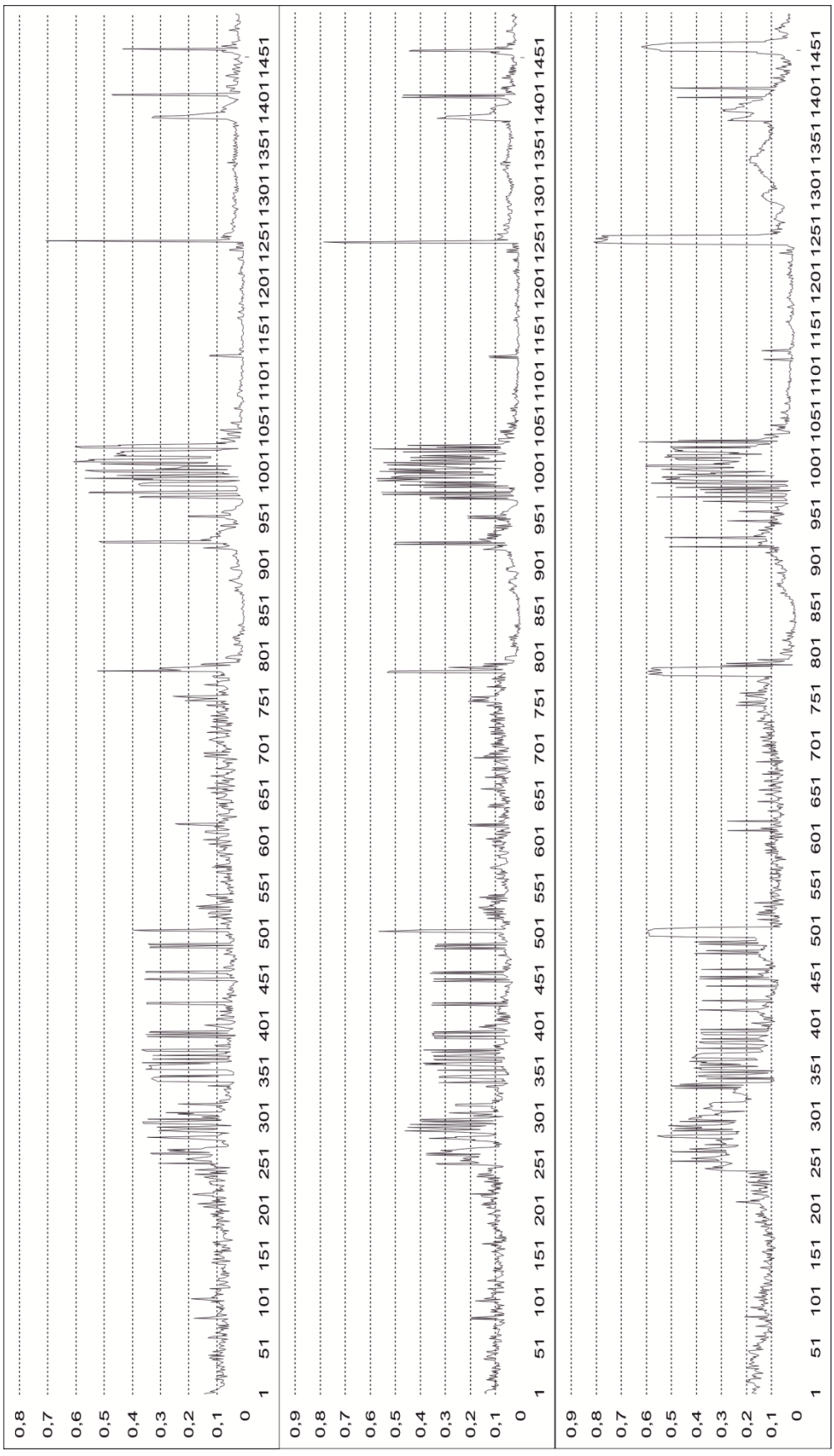


Рисунок 6.14 – Залежність сегментації відео від параметрів сегментації окремих кадрів в ролику «Винищувачі»

Як видно з отриманих результатів, спостерігаються аналогічні попереднім випадкам ситуації, тобто відбувається детектування змін в відеопотоці, що також дозволяє розбити його на множину однорідних сегментів, та залежність від параметрів сегментації кадрів хоч і існує, проте не носить критичного характеру.

6.2 Експериментальний аналіз фрагментної сегментації статичних зображень і відео

Експерименти проводилися із зображеннями з колекцій університету Берклі, що пояснюється наявністю множини результатів інтерактивної сегментації. Ground truth парадигма, тобто порівняння результатів автоматичної сегментації з результатами сегментації, виконаної людиною, як і раніше, є найвірогіднішим інструментом експериментального аналізу методів синтезу розбиття або покриттів поля зору. Всі зображення задавалися в форматі TIFF (Tagged Image File Format) без стиснення для усунення впливу можливих втрат, що особливо важливо при кластеризації безпосередньо в просторі зображень. Дозвіл зображень з метою валідного порівняння результатів фіксувалося на одному рівні – елементів. При необхідності, якщо фрагменти повністю і без залишку не вкладалися на зображення, виконувалося необхідне масштабне перетворення.

Формально просторова сегментація – це пошук кластерів $\{R_i\}_{i=1}^{n}$ при заданій або невідомій їх кількості таких, що $R_i \neq \varnothing$, $\mathcal{R} = \bigcup_{i=1}^{n} R_i$, $\forall i, j \in \{1,2,...,n\} : i \neq j \Rightarrow R_i \cap R_j = \varnothing$ (для фаззі кластеризації ця умова не є

обов'язковою). Для заданого предиката однорідності $\mathcal{P}(\circ)$ для всіх елементів деякої області виконуються умови $\forall i \in \{1,2,...,n\} \Rightarrow \mathcal{P}(R_i) = True$ і $\forall i, j \in \{1,2,...,n\} : i \neq j \Rightarrow \mathcal{P}(R_i \cup R_j) = False$.

Для двох довільних сегментацій $\bar{R}$, $\bar{Q}$, тобто $\mathcal{R} = \bigcup_{i=1}^{n} R_i = \bigcup_{j=1}^{m} Q_j$, кожна з яких може бути ground truth, нормована метрика порівняння «просторового змісту» мала вигляд

$$\rho(\bar{R},\bar{Q}) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} |R_i \Delta Q_j||R_i \cap Q_j|}{1+\sum_{i=1}^{n}\sum_{j=1}^{m} |R_i \Delta Q_j||R_i \cap Q_j|}. \qquad (6.1)$$

На рис. 6.15 наведено типові зображення з колекцій університету Берклі, використані для вивчення запропонованих в розділі 5 методів сегментації.

Тут же показані результати медіанної фільтрації, яку, поряд з гістограмним перетворенням, доцільно використовувати при фрагментній сегментації в силу того, що в цьому випадку ослаблені вимоги до алгоритмів (до уваги береться просторова зв'язність кластерів і не використовуються жодні додаткові характеристики і припущення). В якості критерію подібності фрагментів-точок використовувалася зважена по каналах RGB евклідова метрика.

Рисунок 6.15 – Приклади вихідних зображень (верхній ряд)
і результатів медіанної фільтрації (нижній ряд)

Рисунок 6.16 ілюструє ground truth еталони, для яких слід підкреслити семантичну орієнтацію сегментації. Так, наприклад, без додаткових умов небо не може представлятися одним кластером. Рисунок 6.17 ілюструє модифікацію методу CLARANS при оперуванні різними розмірами фрагментів і заданому числі кластерів – 6.

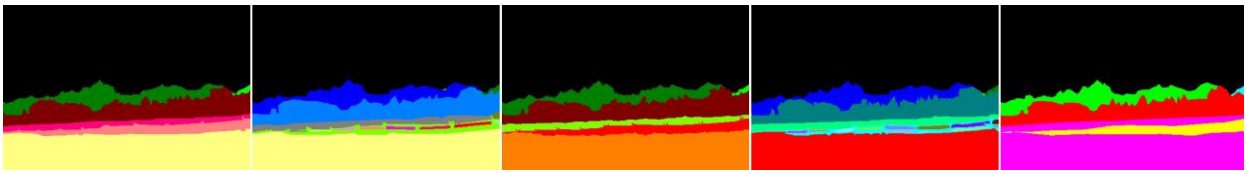


Рисунок 6.16 – Приклади інтерактивної сегментації
середнього зображення, представленого на рис. 6.15

Відзначимо, що збільшення розмірів фрагментів істотно погіршує візуальне сприйняття, але для задачі структуризації «просторовий зміст» зображення зберігається.

На рис. 6.18 показано результати кластеризації модифікованим методом

CURE того ж зображення з фрагментами при зміні кількості кластерів від 4 до 8. Неважко помітити, що вже при кількості кластерів 4 і 5 результати сегментації можуть успішно використовуватися для контекстного пошуку із запитами за зразком.

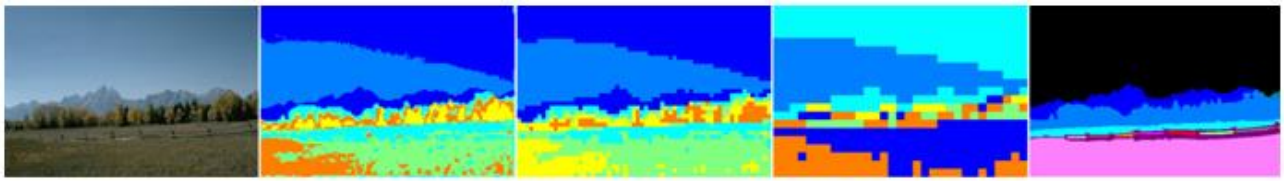


Рисунок 6.17 – Сегментація з фрагментами (3×3), (8×8), (16×16) елементів

При невідомому числі кластерів на основі критерію Цалінского–Харабаша, що базується на відношенні слідів матриць міжкластерного і внутрішньокластерного розсіювання, проведено експериментальний аналіз методу X-середніх.

На рис. 6.19 показано, починаючи з 4 кластерів, послідовність вибору кластера, що підлягає дробленню, і власне результати його дроблення фрагментною модифікацією методу *K*-середніх із зупинкою на 8 кластерах. Результати кластеризації на кожному кроці наведено праворуч.

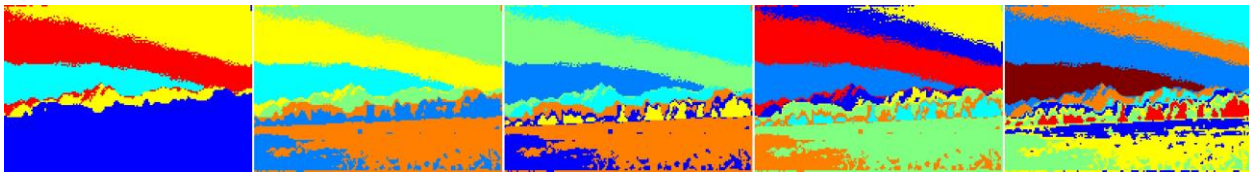


Рисунок 6.18 – Сегментація на 4, 5, 6, 7, 8 кластерів

Як видно з рисунків 6.17 – 6.19, при роботі з візуальною інформацією далеко не завжди єдиний критерій вибору кластера для подальшого його

дроблення забезпечує валідну сегментацію. Так, в останньому прикладі при 7 кластерах вибирається область з «бліковою» структурою, яка не є домінуючою при передачі «просторового змісту». У зв’язку з цим модифікується метрика (6.1) для вкладення сегментів.

На рис. 6.20 показано послідовне дроблення кластерів від 2 від 8 з урахуванням метричних властивостей вкладення сегментів з фрагментами пікселів.

У лівому верхньому кутку наведено результати інтерактивної сегментації.

Слід зазначити, що валідність дроблення кластерів підвищилася, але є цілий ряд незв’язних несуттєвих елементів кластерів.

На рис. 6.21 ці несуттєві підмножини ще більш помітні. Крім цього, особливо слід підкреслити негативний вплив застосування методу *K*-середніх, якому властива залежність від стартових умов. В даному випадку верхній і середній ряди сегментованих зображень помітно різняться.

Усунути локальні похибки, які продукуються запропонованими методами за рахунок використання фрагментів і відмови від обліку зв’язності безпосередньо при пошуку кластерів, можна, використовуючи бінарну морфологію, оскільки кожен кластер можна інтерпретувати як двійкове зображення.

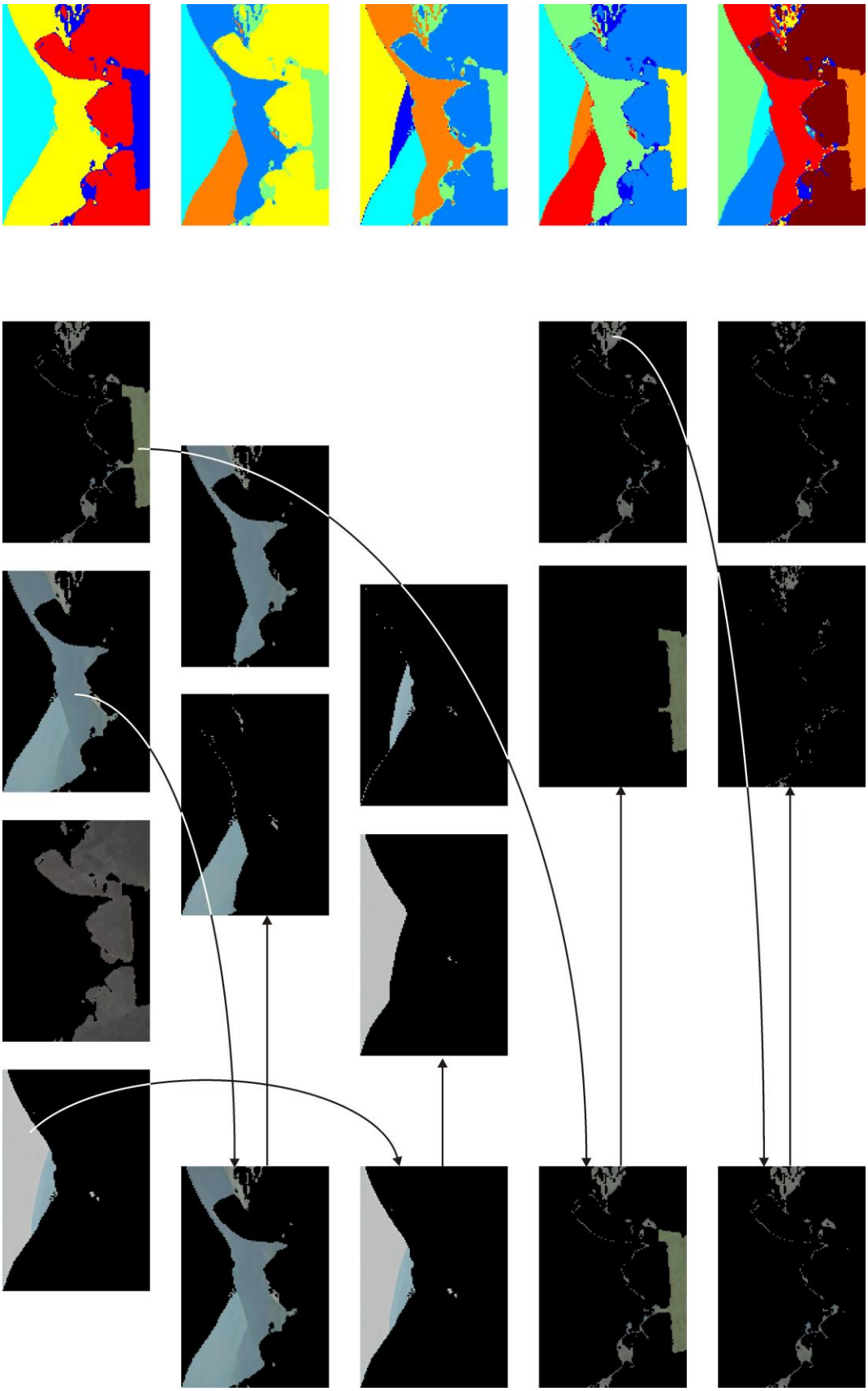


Рисунок 6.19 – Приклад ітераційного дроблення кластерів

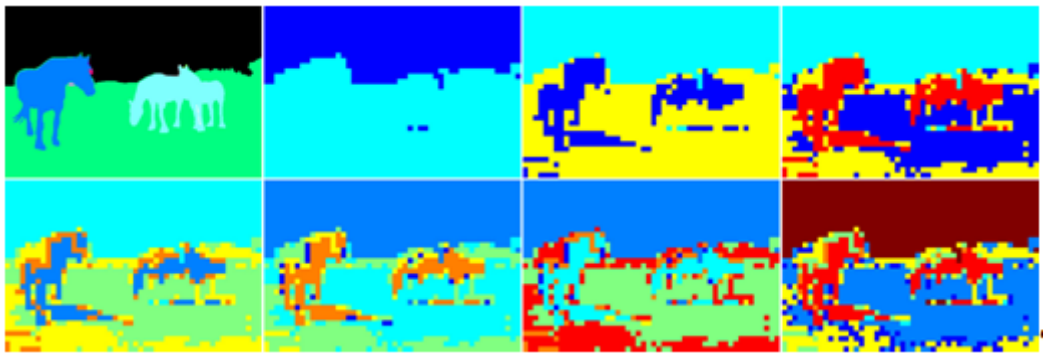


Рисунок 6.20 – Приклад пошуку кількості кластерів (від 2 до 8)

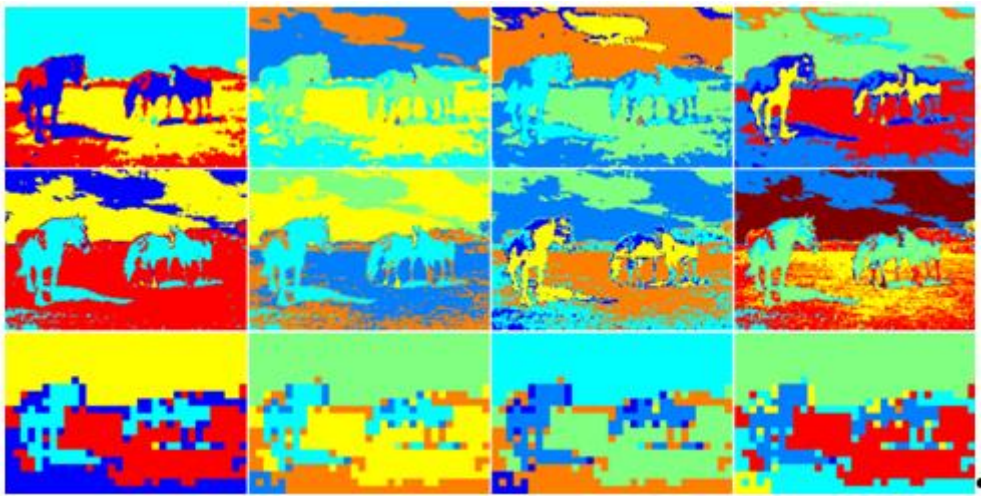


Рисунок 6.21 – Приклади побудови 4, 5, 6, 7 кластерів з фрагментами (3×3) (верхній та середній ряди) та (16×16) елементів

Серед операцій бінарної морфології виділимо операції взяття нутрощі γ і замикання φ двоградаційних зображень.

Нехай $\mathcal{R}=\bigcup_{i=1}^{n}R_i$ – розбиття (покриття), визначене в полі зору $\mathrm{D}\subset\mathbb{Z}^2$. На основі базових операцій бінарної морфології:

– розширення (dilation) – $\delta_{\mathrm{H}}(R_i)=\{x\in R_i:[(\mathrm{H}+x)\bigcap R_i]\subset R_i\}$;

– звуження (erosion) – $\varepsilon_{\mathrm{H}}(R_i)=\{x\in R_i:(\mathrm{H}^r+x)\subset R_i\}$

(H, $\mathrm{H}^r=\{-x\in R_i:x\in\mathrm{H}\}$, $\mathrm{H}\subseteq\mathrm{D}$ – фіксований структурний елемент) введена і

використана операція $\gamma_H(R_i) = \delta_H(\varepsilon_H(R_i))$, що видаляє дрібні об'єкти і тонкі частини великих об'єктів, що призводить до поділу об'єктів, з'єднаних тонкими лініями, і операція, що заповнює дрібні порожнечі на класах еквівалентності (толерантності), що об'єднує сусідні об'єкти і яка веде аналіз багатозв'язних об'єктів до обробки однозв'язних областей.

На рис. 6.22 показано результати морфологічної фільтрації.

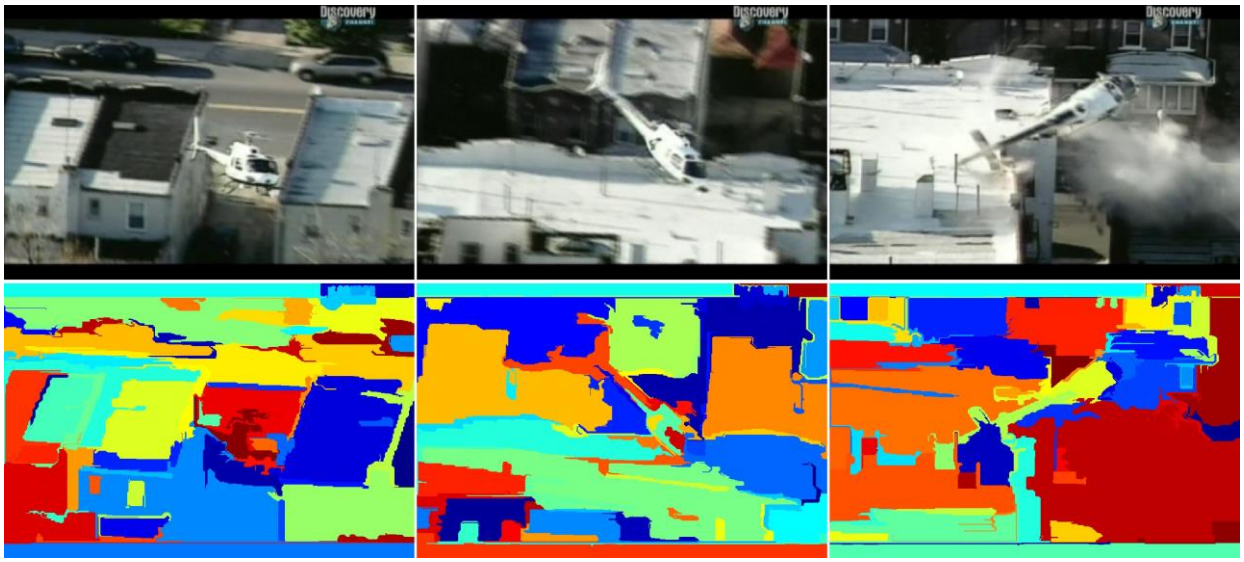


Рисунок 6.22 – Приклади сегментації після морфологічної обробки

Далі розглянемо використання цих підходів при аналізі відеопослідовностей. Експерименти проводилися з відеорядами, що складаються з 750, 1200 і 1500 відеокадрів, що відповідає тривалості відео 30, 48 і 60 секунд. Використовувався медіаконтейнер AVI (Audio Video Interleaved) з відношенням сторін 4: 3, частотою 25 кадр / сек. Для валідного зіставлення результатів роздільна здатність фіксувалася на рівні пікселів. Як відеопотоки використані сюжети з 40 фільмів "Destroyed in seconds". Кожен відеокадр представлявся в форматі TIFF без стиснення.

Розглянемо формальні аспекти експериментальних досліджень сегментації відеорядів. Нехай дана часова послідовність $X = \{x_1, x_2, ..., x_N\}$, точніше кажучи,

зростаюча в часі вибірка (по суті – відеоряд), що забезпечує online обробку. Тут кожен член ряду представляється відеокадром (фрагментом або набором довільно розташованих на зображенні фрагментів) $x_i = \{\mathrm{B}_{kl}\}_{i=1}^{N}$ (B_{kl} – значення кольору в точці $(k,l) \in R^2 \bigcap R^2$, тобто фактично $k \in \{1,2,...,\mathrm{K}\}, l \in \{1,2,...,\mathrm{L}\}$). Тут K і L визначають розміри фрагменту. Для спрощення нотації вважаємо, що початок системи координат для кожного фрагмента переноситься в лівий верхній кут. На $\mathrm{X} \times \mathrm{X}$ задана метрика (6.1), яка визначає відстані між парами сегментації елементів $x_i, x_j \in \mathrm{X}$. Якщо $\rho(x_i, x_j) \le \varepsilon$ (ε – поріг, що визначається апріорно або динамічно), зображення або фрагменти належать одному кластеру. Під i-им сегментом $[x_{\alpha(i)}, x_{\beta(i)}]$ довжиною (потужністю кластера) $n_i = \beta(i) - \alpha(i) + 1$ відеоряду $\mathrm{X} = \{x_1, x_2, ..., x_N\}$ розуміється множина $\{x_{\alpha(i)}, x_{\alpha(i)+1}, ..., x_{\beta(i)-1}, x_{\beta(i)}\}$, в якій всі пари $x_m, x_n \in [x_{\alpha(i)}, x_{\beta(i)}]$ схожі і, як правило, не схожі при $\alpha(i) - 1 \ge 1,\ \beta(i) + 1 \le N$. У задачі сегментації відеорядів безсумнівний інтерес викликають випадки ослаблення вимог за подібністю всіх пар $x_m, x_n \in [x_{\alpha(i)}, x_{\beta(i)}]$, тому що точки $x_{\alpha(i)}, x_{\alpha(i)+1}, ...$ надходять послідовно і аналіз здійснюється в кожен момент часу. Інакше кажучи, для «нарощування» сегмента $[x_{\alpha(i)}, x_m]_k$ в точці x_m перевіряється схожість лише з $1 \le l \le m - \alpha(i)$ зображеннями або фрагментами. Зі сказаного випливає, що множина фрагментів, що підлягає кластеризації належить різним відеокадрам. У послідовності окремих фрагментів (вони можуть і перетинатися на зображеннях), якщо їх розміри помітно менше розмірів повних зображень, може спостерігатися і сталість їх подібності, наприклад, при попаданні фонових складових переднього або заднього плану, і локальна зміна яскравості властивостей. Інакше кажучи, реальна кластеризація відеоряду повинна визначатися на основі голосування

локальних результатів. Відзначимо, що справжні межі сегментів при проведенні експериментів визначалися шляхом покадрового порівняння відео.

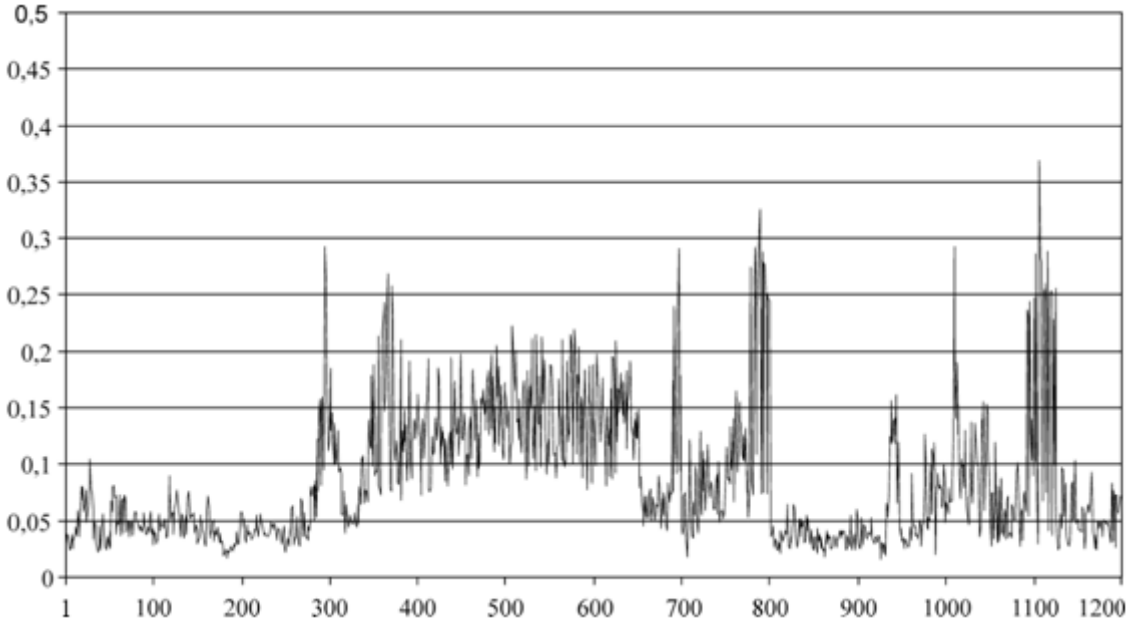


Рисунок 6.23 – Приклад виділення меж відеосегментів

На рис 6.23 наведено приклад попарних послідовних різниць просторових сегментації відеокадрів, що підтверджує правомочність наведеної вище схеми контролю сегментації відеоряду в часі. Рисунок 6.24 ілюструє просторову сегментацію 4 зображень відеоряду, а рис. 6.25 – сегментацію фрагмента (500×120) зображення субфрагментами (3×3), (5×5) та (8×8). При цьому для першого і третього випадку фрагмент був адаптований до (501×120) і (504×120) пікселів відповідно.

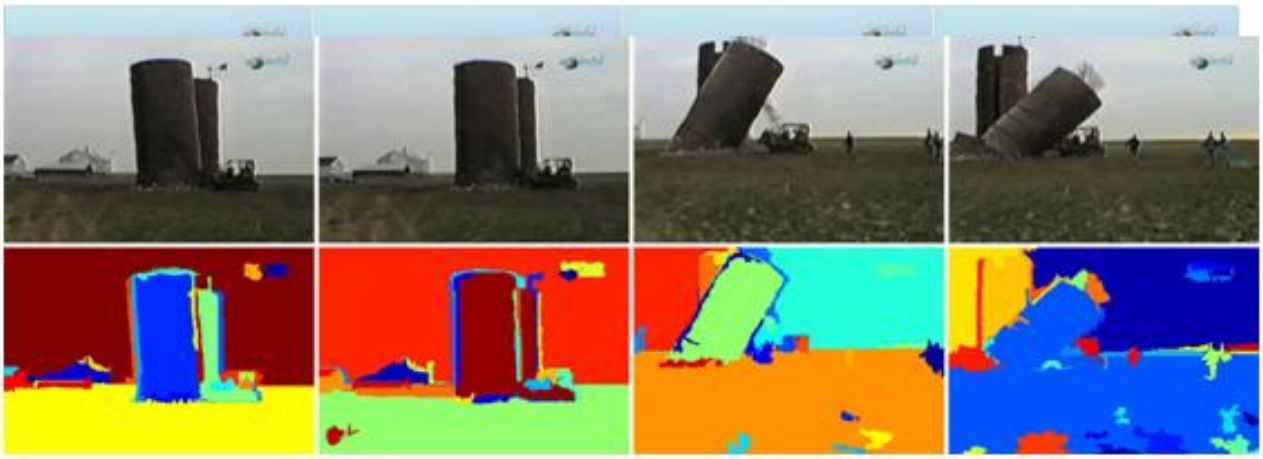


Рисунок 6.24 – Приклад сегментації зображень відеоряду

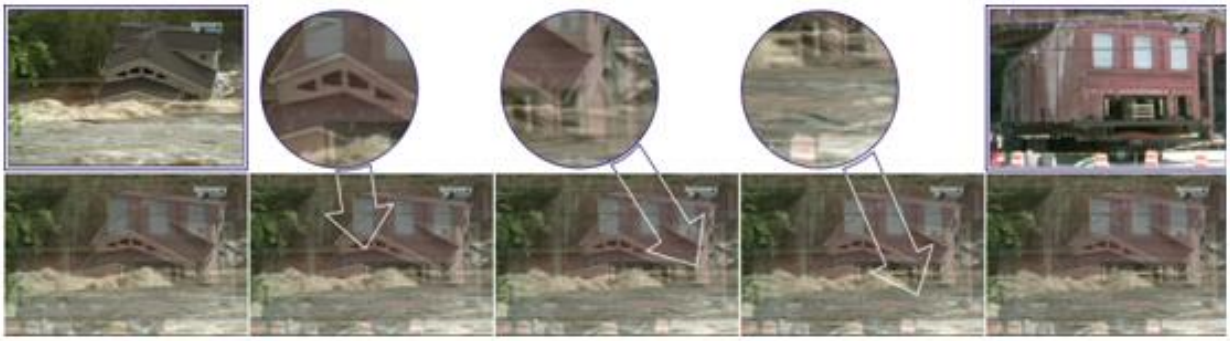


Рисунок 6.26 – Ефект «розчинення» відеокадрів

Слід особливо підкреслити, що толерантність при сегментації відеорядів має більш явну природу, ніж при просторовій кластеризації. Якщо в останньому випадку – це зазвичай нечіткості меж областей, то в першому – плавні зміни сюжетів (в тому числі повільне панорамування відеодатчики). На рис. 6.26 як приклад показано «розчинення» відеокадрів (тривалість зміни сюжету становить 50 відеокадрів). Ясно, що однозначне віднесення відеокадрів до одного з кластерів неможливо. Більш того – на подібних зображеннях просторова сегментація часто не має сенсу, однак попарна відмінність фрагментів послідовних зображень все-таки дозволять детектувати подібні випадки.

При темпоральній сегментації багатовимірних часових рядів, індукованих відеопослідовностями, в силу широкої варіативності властивостей цих рядів слід комплексно використовувати запропоновані online методи виявлення розладнань, адаптивно вибираючи розбиття або покриття відео. Фрагментні методи темпоральної обробки відепослідовностей (кластеризації і прогнозу) створюють передумови локалізації областей інтересу, що може помітно підвищувати точність і надійність аналізу відео при тих же обчислювальних ресурсах.

ВИСНОВКИ

Дослідження проводились на стику підходів, що існують або розвиваються у межах таких напрямків, як Dynamic Data Mining, Data Stream Mining, гібридні системи обчислювального інтелекту, еволюційні коннекціоністські системи. Кожен з цих напрямків має свої особливості, що обмежують коло задач, що вирішуються. Особливо «проблемними» є задачі, що пов’язані з опрацюванням потоків даних за умов суттєвої невизначеності, причому принципові труднощі виникають у випадку, коли ця невизначеність є структурною.

Тому важливим і новим є створення адаптивних систем обчислювального інтелекту, функціонування яких відбувається одночасно з навчанням (самонавчанням), при цьому в режимі реального часу повинні налаштовуватися не лише синаптичні ваги, як це відбувається у традиційних нейронних мережах, але й функції активації, належності та архітектура системи.

Було визначено необхідність об’єднання ідей, що лежать в основі Dynamic та Temporal Data Mining, Data Stream Mining, обчислювального інтелекту, еволюційних систем та Data Science для вирішення широкого класу задач опрацювання інформації таких, як класифікація, кластеризація, прогнозування, емуляція, тощо в ситуаціях, коли дані надходять в online режимі у формі багатовимірних часових рядів або потоків відео, що характеризуються високою розмірністю та об’ємом (Big Data), значним рівнем апріорної невизначеності.

Основою досліджень є підходи та методи, що базуються на новому науковому напрямку «Computational Intelligence in Big Data», який щойно

почав розвиватися у всьому світі.

Особливістю дослідження є те, що на основі запропонованих систем вирішується цілий спектр задач інтелектуального аналізу даних, що базуються на навчанні та самонавчанні, апроксимації та екстраполяції, архітектурах, що зростають та скорочуються, при цьому було створено уніфіковані архітектури, що здатні вирішувати різні задачі.

Ще однією особливістю є те, що часові ряди та потоки відео мають різну структуру, при цьому «пряма» векторизація зображення знищує важливі зв'язки, що містяться у відео. Тому необхідно було створити методи, що здатні в online режимі опрацьовувати як векторні, так і матричні сигнали.

В результаті проведених досліджень створено новий клас адаптивних систем обчислювального інтелекту, що умовно можуть бути названі як еволюційні адаптивні нео-фаззі системи, що здатні змінювати свою архітектуру безпосередньо в процесі опрацювання потоків даних. Оскільки висока розмірність вихідних даних не дозволяє використовувати («концентрація норм») традиційні критерії навчання і, перш за все, класичний квадратичний, було введено нові цільові функції, а відповідно і нові методи навчання, що комбінують у собі як оптимізаційний підхід, так і «ліниве» (lazy) та «активне» (active) навчання, що зараз інтенсивно розвиваються у всьому світі.

Розроблено методи, на основі яких створено інтелектуальні системи обробки потоків даних високої розмірності.

Розроблено та досліджено архітектури адитивних адаптивних швидкісних (оптимальних за швидкодією) нео-фаззі моделей та вейвлет-нео-фаззі систем, що забезпечують підвищену швидкодію при оброблянні

потоків даних у реальному часі, а також методи їх навчання. В основу синтезу були покладені ідеї адаптивних адитивних узагальнених моделей, еволюційних гібридних систем обчислювального інтелекту та глибинного навчання (Deep Learning), що зараз інтенсивно розвивається у всьому світі.

Розроблено адаптивні методи нечіткого опрацювання інформації, що надходить у формі потоків даних у реальному часі, на основі нейро-фаззі- та нео-фаззі-підходів, що не потерпають від ефектів «прокльону розмірності» та «концентрації норм». Цей ефект було досягнуто на основі використання нечітких цільових функцій спеціального типу з використанням поліноміальних фаззіфікаторів та автоматичним зважуванням компонент векторів спостережень, що дозволяє виділити найбільш цінну інформацію (в сенсі задачі, що розглядається) в процесі online опрацювання даних. Запропоновано та досліджено швидкодіючі адаптивні методи навчання систем, що розглядаються. Введено еволюційні системи із змінною архітектурою, отримані результати щодо поширення підходу на нечислові шкали.

Таким чином, розроблено та досліджено групу нових методів для вирішення задачі адаптивної нечіткої кластеризації потоків даних високої розмірності, що підтверджується результатами імітаційного моделювання та експериментального аналізу даних на тестових та реальних задачах.

Розроблено методи нечіткого опрацювання даних, що здатні оцінювати апріорі невідому кількість класів та кластерів, що можуть змінюватись у процесі аналізу інформації.

В основі запропонованого підходу полягає ідея ансамблю кластерувальних нейро-фаззі-систем, що паралельно опрацьовують потік інформації, при цьому кожен з членів ансамблю розрахований на різну

кількість можливих кластерів та різні параметри цільових функцій. У вихідному шарі ансамблю формулюється оптимальне рішення, яке може змінюватися в процесі online опрацювання інформації у випадку нестаціонарності вхідного потоку.

Розроблено адаптивні методи нечіткого опрацювання інформації, що є ефективними за умов кластерів та класів довільної форми та високого рівня їх перетинання. В основі полягають ядерні процедури підвищення розмірності вихідного простору, що використовують ідеї «лінивого» навчання та концепції «нейрона в точках даних». Введено та досліджено нечіткі цільові функції кластеризації, що є ефективними за умов високої розмірності перетвореного простору ознак.

Створено систему просторово-часової сегментації відеорядів та метричного темпорального аналізу часткової семантики динамічної візуальної інформації на базі методів просторово-часової сегментації відеопотоків із виявленням змін в багатовимірних часових рядах, зокрема на основі налаштовуваних моделей, експоненціального типу і аналізу головних компонент, що дозволяє класифікувати відеосегменти в online режимі при послідовному аналізі відеопотоків.

Розроблено методи і моделі співставлення результатів темпоральної обробки потоків відеоданих за умов апріорної невизначеності, які дозволяють значно скоротити час, необхідний для розв’язання задач контекстного аналізу даних.

Розроблено рекомендації з використання запропонованих методів та моделей інформаційного пошуку у випадках запитів «за зразком» на основі експериментальних досліджень потоків відеоданих з різним змістом і характеристиками відеосигналів.

Розроблено та досліджено систему співставлення результатів аналізу об’єктів, що представлено візуальною інформацією, та сформульовано практичні рекомендації для користувачів на основі експериментальних досліджень потоків відеоданих з різним змістом і характеристиками відеосигналів.

Розроблені методи просторово-часової сегментації дозволяють виділяти сегменти відео з однорідною за змістом структурою/характеристиками вхідних даних. Це дозволяє скоротити змістовний опис даних за рахунок заміни всього сегменту його єдиним представником – ключовим кадром, що значно скорочує кількість порівнянь, яку потрібно зробити при контекстному аналізі відео послідовностей.

В той же час розроблені методи фрагментної кластеризації дозволяють зосередити свою увагу на більш вагомих частинах відео і отримувати для них кращі результати.

Таким чином, в результаті проведених досліджень створено нові методи, математичне забезпечення та їх теоретичне обґрунтування для розв’язання проблем таких нових наукових напрямів, що сьогодні інтенсивно розвиваються, як Dynamic Data Mining та Data Stream Mining на основі гібридних систем обчислювального інтелекту і, перш за все, нейро-фаззі-систем, нео-фаззі-систем та вейвлет-нейро-фаззі-систем із спеціалізованою архітектурою.

Розроблено нові системи та швидкі методи їх навчання для обробки послідовностей нечітких даних високої розмірності за умов невизначеності щодо їх властивостей, при цьому інформація на обробку надходить у online режимі.

Особливістю введених систем є можливість роботи при високому рівні апрірної та поточної невизначеності: характеру нестаціонарності, кількості та форми класів-кластерів, значному їх перетинанні, можливості виникнення нових класів у процесі надходження інформації.

Розроблені методи створення інтелектуальних систем обробки потоків даних високої розмірності є науково обґрунтованими, базуються на сучасному апараті обчислювального інтелекту, Big Data, еволюційних конекціоністських системах. Результати досліджень перевірено шляхом їхнього тестування та верифікації як на стандартних репозиторних даних (UCI), так і при розв'язанні суто практичних задач, пов'язаних з медичною діагностикою, обробкою зображень, Text Mining, обробкою реальних даних, наданих кримінальною поліцією.

Запропоновані нові підходи до розробки адаптивних систем обчислювального інтелекту для online опрацювання нестаціонарних потоків даних можуть бути використані для розв'язання низки практичних завдань і, перш за все, темпоральної обробки потоків відео, Web Mining, Medical Data Mining, Content Based Information (Image, Video) Retrieval, прогнозування, класифікації та кластеризації, а також визначення розладнань у технічних, виробничих системах, системах спеціального призначення та Green IT, для медичної діагностики.

Отримані результати дозволяють підвищити ефективність розробки систем спостереження за об'єктами, що представлені у вигляді відеоінформації. Це може бути застосовано у системах спостереження за рухомими об'єктами (наземними, морськими, повітряними), що в кінцевому підсумку створює передумови для створення систем швидкого пошуку неструктурованої мультимедійної інформації (зображень, відеопотоків) із

запитами у природній для користувача формі.

Зокрема, запропоновані математичні методи можуть бути застосовані для розробки підсистем «Інтелектуального дому (Smart House) для людей з обмеженими фізичними можливостями», для моделювання та прогнозування кризових (надзвичайних) ситуацій у Ситуаційному центрі національної поліції України (СЦНПУ).

Результати досліджень також використовуються в науково-дослідній роботі студентів, в атестаційних роботах магістрів за спеціальністю 122 – комп'ютерні науки (спеціалізації «Системи штучного інтелекту» та «Інформатика»), а також у дисертаційних роботах докторантів та аспірантів за спеціальністю 05.13.23 – системи та засоби штучного інтелекту.

Отримані результати дозволили оновити лекційні курси «Інтелектуальний аналіз даних», «Штучні нейронні мережі: архітектура, навчання та застосування», що викладаються при підготовці бакалаврів; «Нейромережеві методи обчислювального інтелекту», «Глибинне навчання нейронних мереж», «Нечіткі моделі і методи обчислювального інтелекту», «Кластеризація та класифікація даних», «Комп'ютерний зір», що викладаються при підготовці магістрів; «Гібридні системи обчислювального інтелекту», «Методи обробки та розпізнавання візуальної інформації», що викладаються аспірантам.

Синтезовані інтелектуальні системи підвищеної швидкодії підтвердили свою ефективність при опрацюванні у реальному часі потоків даних високої розмірності та великих масивів даних за умов апріорної та поточної невизначеності-нечіткості. Отримані результати дослідження відповідають світовому рівню і на цей час не мають аналогів, що підтверджується низкою публікацій у провідних закордонних журналах, що

індексуються в наукометричних базах Scopus та Web of Science.

Зазначені наукові результати є актуальними та є підґрунтям для створення нових підходів і методів для опрацювання великих та надвеликих обсягів даних різної фізичної природи на основі глибинного навчання.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Montgomery, D.C. Forecasting and Time Series Analysis / D.C. Montgomery, L.A. Johnson, J.S. Gardiner. – N.Y.: Mc Graw-Hill, 1990. – 394 p.

2. Pham, D.T. Neural Networks for Identification, Prediction and Control / D.T. Pham, X. Liu. – London: Springer–Verlag, 1995. – 238 p.

3. Masters, T. Neural, Novel & Hybrid Algorithms for Time Series Prediction / T. Masters. – N.Y.: John Wiley & Sons, Inc., 1995. – 514 p.

4. Pokropińska, A. Financial Prediction with neuro-fuzzy systems / A. Pokropińska, R. Scherer // Lecture Notes in Computer Science (Proc. of Int. Conf. Artificial Intelligence and Soft Computing "ICAISC-2008"). – Springer-Berlin-Heidelberg. – 2008. – 5097. – P. 1120-1126.

5. Chan, K. Efficient time series matching by wavelets / K. Chan // Proc. of 15th IEEE Int. Conf. on Data Engineering. – 1999. – P. 126-133.

6. Востров, Г.Н. Сегментация и анализ временных рядов на основе стохастической фрактальной модели / Г.Н. Востров, М.В. Поляков, В.В. Любченко // Труды Одесского политехнического университета. – Одесса. – 2001. – 1. – С. 109-114.

7. Last, M. Data Mining In Time Series Databases / M. Last, A. Kandel, H. Bunke. – World Scientific Publishing, 2003. – 192 p.

8. Bezdek, J.C. Pattern Recognition with Fuzzy Objective Function Algorithms / J.C. Bezdek. – N.Y.: Plenum Press, 1981. – 234 p.

9. Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition / [F. Hoeppner, F. Klawonn, R. Kruse, T. Runkler] – Chichester: John Wiley & Sons, 1999. – 300 p.

10. Klawonn, F. Fuzzy shell cluster analysis / F. Klawonn, R. Kruse, H. Timm // Learning, Networks and Statistics [Eds. Della Riccia G., Lenz H.J., Kruse R.] – Wien: Springer-Verlag. - 1997. – P. 105–120.

11. Chui, C.K. An Introduction to Wavelets / C.K. Chui. – New York: Academic, 1992. – 264 p.

12. Daubechies, I. Ten Lectures on Wavelets / I. Daubechies. – Philadelphia, PA: SIAM, 1992. – 228 p.

13. Meyer, Y. Wavelets: Algorithms and Applications / Y. Meyer – Philadelphia, PA: SIAM, 1993. – 133 p.

14. Lekutai, G. Self-tuning control of nonlinear systems using neural network adaptive frame wavelets / G. Lekutai, H.F. van Landingham // Proc. IEEE Int. Conf. on Systems, Man and Cybernetics. – Piscataway, N.J. – 1997. – 2. – P. 1017-1022.

15. Bodyanskiy, Ye. An adaptive learning algorithm for a wavelet neural network / Ye. Bodyanskiy, N. Lamonova, I. Pliss, O. Vynokurova // Expert Systems. – 2005. – 22. – №. 5 – P. 235-240.

16. Bodyanskiy, Ye. Learning wavelet neuron based on the RASP-function / Ye. Bodyanskiy, V. Kolodyazhniy, I. Pliss, O. Vynokurova // Радиоэлектроника. Информатика. Управление. – № 1(11). – 2004. – С. 118-122.

17. Бодянский, Е.В. Адаптивная гибридная вэйвлет-нейронная сеть для решения задачи прогнозирования и эмуляции / Е.В. Бодянский, Е.А. Винокурова, Н.С. Ламонова // Сб. науч. трудов 12-й международной конференции по автоматическому управлению «Автоматика 2005», Т.3. – Харьков: Изд-во НТУ «ХПИ». – 2005. – С. 40-41.

18. Бодянский, Е.В. Треугольный вэйвлет и формальный нейрон на его основе / Е.В. Бодянский, Е.А. Винокурова // Сб. наук. праць 3-ої Міжнародної науково-практичної конференції «Математичне та программне забезпечення інтелектуальних систем» (MPZIS-2005) – Дніпропетровськ: ДНУ. – 2005. – С. 14-15.

19. Billings, S.A. A new class of wavelet networks for nonlinear system identification / S.A. Billings, H.-L. Wei // IEEE Trans. on Neural Networks. – 2005. – 16. – №. 4. – P. 862-874.

20. Szu, H.H. Neural network adaptive wavelets for signal representation and classification / H.H. Szu, B. Telfer, S. Kadambe // Opt. Eng. – 1992. – 31. – P. 1907-1916.

21. Zhang, Q.H. Wavelet networks / Q.H. Zhang, A. Benveniste // IEEE Trans. on Neral Networks. – 1992. – v. 3. – №. 6. – P. 889-898.

22. Dickhaus, H. Classifying biosignals with wavelet networks / H. Dickhaus, H. Heinrich // IEEE Eng. Med. Biol. Mag. – 1996. – 15. – № 5. – P. 103–111.

23. Predicting chaotic time series with wavelet networks / [L.Y. Cao, Y.G. Hong, H.P. Fang, G.W. He] // Phys. D. – 1995. – 85. – P. 225–238.

24. Zhang, J. Wavelet neural networks for function learning / J. Zhang, G.G. Walter, Y. Miao, W.N.W. Lee // IEEE Trans. on Signal Process. – 1995. – 43. – № 6. – P. 1485–1497.

25. Zhang, Q.H. Using wavelet network in nonparametric estimation / Q.H. Zhang // IEEE Trans. on Neural Networks. – 1997. – 8. – №. 2. – P. 227–236.

26. Soltani, S. On the use of wavelet decomposition for time series prediction / S. Soltani // Neurocomputing. – 2002. – 48. – P. 267–277.

27. Huber, P.J. Robust Statistics / P.J. Huber. – N.Y.: John Wiley & Sons, 1981 – 320 p.

28. Хампель, Ф. Робастность в статистике. Подход на основе функций влияния / Ф. Хампель, Э. Рончетти, П. Рауссеу, В. Штаэль. – М.: Мир, 1989. – 512 с.

29. Rey, W.J.J. Robust statistical methods / W.J.J. Rey // Lecture Notes in Mathematics. – Berlin; Heidelberg; N.Y.: Springer-Verlag, 1978. – Vol. 690. – P. 105– 120.

30. Holland, P.W. Robust regression using iteratively reweighted least-squares / P.W. Holland, R.E. Welsch // Communications in Statistics – Theory and Methods. – 1977. – Vol. A6. – P. 813–827.

31. Cichocki, A. Adaptive analogue network for real time estimation of basic waveforms of voltages and currents / A. Cichocki, T. Lobos // IEE Proc.C. – 1992. – 139. – 4. – P. 343-350.

32. Cichocki, A. Neural Networks for Optimization and Signal Processing / A. Cichocki, R. Unbehauen. – Chichester: Wiley, 1993. – 544 p.

33. Li, S.-T. Function approximation using robust wavelet neural networks / S.-T. Li, S.-C. Chen // Proc. of the 14th IEEE Int. Conf. on Tools with Artificial Intelligence. – 2002. – P. 483-488.

34. Karayiannis, N.B. Fast learning algorithm for neural networks / N.B. Karayiannis, A.N. Venetsanopoulos // IEEE Trans. on Circuits and Systems. – 1992. – II. – 39. – P. 453-474.

35. Бодянский, Е.В. Робастный алгоритм обучения радиально-базисной адаптивной фаззи-вэйвлет-нейронной сети / Е.В. Бодянский, Е.А. Винокурова // Адаптивные системы автоматического управления. – Днепропетровск: Системные технологии. – 2007. – № 11(31). – С. 3-15.

36. Yamakawa, T. Wavelet neural networks employing over-complete number of compactly supported non-orthogonal wavelets and their applications / T. Yamakawa, E. Uchino, T. Samatu // IEEE Int. Conf. on Neural Networks, Orlando, USA. – 1994. – P. 1391-1396.

37. Yamakawa, T., The wavelet network using convex wavelets and its application to modeling dynamical systems / T. Yamakawa, E. Uchino, T. Samatu // The Trans. on the IEICE. – 1996. – J79-A. – № 12. – P. 2046-2053.

38. Yamakawa, T. A novel nonlinear synapse neuron model guaranteeing a global minimum – Wavelet neuron / T. Yamakawa // Proc. 28-th IEEE Int. Symp. on Multiple-Valued Logic. – Fukuoka, Japan: IEEE Comp. Soc., 1998. – P. 335-336.

39. Bodyanskiy, Ye. Double-Wavelet Neuron based on analytical activation functions / Ye. Bodyanskiy, N. Lamonova, O. Vynokurova // Int. J. Information Theory and Applications. – 2007. – 14. – P. 281-288.

40. Bodyanskiy, Ye. Recurrent learning algorithm for double-wavelet neuron / Ye. Bodyanskiy, N. Lamonova, O. Vynokurova // Proc. XII -th Int. Conf. "Knowledge – Dialogue – Solution", Varna. – 2006. – P. 77-84.

41. Бодянский, Е.В. Робастный гибридный двойной вэйвлет-нейрон в задачах обработки динамики показателей гомеостаза при остром стресс-повреждении / Е.В. Бодянский, Е.А. Винокурова, А.А. Павлов // Матеріали міжнародної наукової конференції «Інтелектуальні системи прийняття рішень та проблеми обчислювального інтелекту». – Херсон. – 2008. – 3(1). – С. 56-59.

42. Moody, J. Fast learning in networks of locally-tuned processing units / J.Moody, Darken C.J. // Neural Computation. – 1989. – 1. – P. 281-294.

43. Moody, J. Learning with localized receptive fields / J. Moody, C. Darken // Proc. of the 1988 Connectionist Models Summer School [Eds. D. Touretzky, G. Hinton, T. Sejnowski]. – 1988. – San Mateo: Morgan-Kaufmann. – P. 133-143.

44. Park, J. Universal approximation using radial-basis-function networks / J. Park, I.W. Sandberg // Neural Computation. – 1991. – 3. – P. 246-257.

45. Leonard, J.A. Using radial basis functions to approximate a function and its error bounds / J.A. Leonard, M.A. Kramer, L.H. Ungar // IEEE Trans. on Neural Networks. – 1992. – 3. – P. 614-627.

46. Bishop, C.M. Neural Networks for Pattern Recognition / C.M. Bishop. – Oxford: Clarendon Press, 1995. – 482 p.

47. Reyneri, L.M. Unification of neural and wavelet networks and fuzzy systems / L.M. Reyneri // IEEE Trans. on Neural Networks. – 1999. – 10. – P. 801-814.

48. Jang, J.S.R. Functional equivalence between radial basis function networks and fuzzy inference systems / J.S.R. Jang, C.T. Sun // IEEE Trans. on Neural Networks. – 1993. – 4. – P. 156-159.

49. Hunt, K.J. Extending the functional equivalence of radial basis function networks and fuzzy inference systems / K.J. Hunt, R. Haas, R.M. Smith // IEEE Trans. on Neural Networks. – 1996. – 7. – P. 776-781.

50. Mitaim, S. What is the best shape for a fuzzy set in function approximation? / S. Mitaim, B. Kosko // Proc. 5th IEEE Int. Conf on Fuzzy Systems “Fuzz-96”. – 1996. – Vol. 2. – P. 1237-1213.

51. Mitaim, S. Adaptive joint fuzzy sets for function approximation / S. Mitaim, B. Kosko // Proc. Int. Conf. on Neural Networks “ICNN-97”. - 1997. - P. 537-542.

52. Бодянский, Е.В. Радиально-базисная фаззи-вэйвлет-нейронная сеть с многомерными вэйвлет-активационными функціями / Е.В. Бодянский, А.А. Беднарская, Е.А. Винокурова // Зб. наук. праць Міжнародної наукової конференції «Інтелектуальні системи прийняття рішень та прикладні аспекти інформаційних технологій». – Херсон. – 2007. – 3. – С. 11-13.

53. Бодянський, Є.В. Прогнозуюча радіально-базисна вейвлет-нейронна мережа з гіпереліпсоїдальними рецепторними полями, що настроюються / Є.В. Бодянський, Г.О Беднарська, О.А. Винокурова // Материалы междунар. науч.-практ. конф. «Информационные технологии и информационная безопасность в науке, технике и образовании "ИНФОТЕХ-2007"». – Ч. 2. – 2007. – Севастополь: Изд-во СевНТУ. – С. 3-6.

54. Винокурова, О.А. Алгоритм навчання радіально-базисної фаззі-вейвлет-нейронної мережі / О.А. Винокурова, Г.О. Беднарська, І.П. Плісс // Прикладна радіоелектроніка. – 2007. – 6. – № 3. – С. 427-431

55. Itakura, F. Maximum prediction residual principle applied to speech recognition / F. Itakura // IEEE Trans. on Acoustics, Speech, and Signal Processing. – 1975. – 23. – P. 67-72.

56. Бодянский, Е.В. Робастный алгоритм обучения радиально-базисной адаптивной фаззи-вэйвлет-нейронной сети / Е.В. Бодянский, Е.А. Винокурова // Адаптивные системы автоматического управления. – Днепропетровск: Системные технологии. – 2007. – № 11(31). – С. 3-15.

57. Bodyanskiy, Ye. Robust learning algorithm for wavelet-neural fuzzy network based on Polywog wavelet / Ye. Bodyanskiy, O. Vynokurova // Системные технологии. – 2008. – № 3(56). – Т. 2 – С. 129-134.

58. Bodyanskiy, Ye. Outliers resistant learning algorithm for radial-basis-fuzzy-wavelet-neural network in stomach acute injury diagnosis tasks / Ye. Bodyanskiy, O. Pavlov, O. Vynokurova // [Eds. by K. Markov, K. Ivanova, I. Mitov] Internation Book Series “Information Science and Computing”, Number 2. – Sofia: Institute of Information Theories and Application FOI ITHEA, 2008. – P. 55-62.

59. Бодянский, Е.В. Адаптивные алгоритмы идентификации нелинейных объектов управления / Е.В. Бодянский // АСУ и приборы автоматики. – Харьков: Выща шк., 1987. – Вып. 81. –С. 43-46.

60. Bodyanskiy, Ye. An adaptive learning algorithm for a neuro-fuzzy network / Ye. Bodyanskiy, V. Kolodyazhniy, A. Stephan [Ed. By B. Reusch] // “Computational Intelligence. Theory and Applications”. – Berlin Heidelberg New York: Springer. – 2001. – P. 68-75.

61. Jang J.-S.R. Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence / J.-S.R. Jang, C.-T. Sun, E. Mizutani. – Upper Saddle, NJ: Prentice Hall, 1997. – 614 p.

62. Бодянський, Є.В. Адаптивні вейвлет-нейро-фаззі конструкції обробки нестаціонарних нелінійних сигналів у системах обчислювального інтелекту / Є.В. Бодянський, О.А. Винокурова // Праці IV-ї міжнародної школи-семінару «Теорія прийняття рішень». – Ужгород, УжНУ, 2008. – С. 24-25.

63. Винокурова, Е.А. Робастный алгоритм обучения адаптивной нейро-фаззи системы / Е.А. Винокурова, Н.С. Ламонова // Матеріали VII Міжнародної науково-практичної конференції “Математичне та програмне забезпечення інтелектуальних систем”. – Дніпропетровськ: ДНУ. - 2009. – С. 58.

64. Bodyanskiy, Ye. Forecasting adaptive wavelet-neuro-fuzzy-network using Polywog activation function / Ye. Bodyanskiy, G. Bednarska, O. Vynokurova // Proc. 3-rd International Conference “Advanced Computer Systems and Networks: Design and Application”. – Lviv. – 2007. – P.79-82.

65. Bodyanskiy, Ye. A learning algorithm for forecasting adaptive wavelet-neuro-fuzzy network / Ye. Bodyanskiy, I. Pliss, O. Vynokurova // Proc. XIII -th Int. Conf. "Information Research & Applications". – Varna (Bulgaria). – 2007. – P. 211-218.

66. Bodyanskiy, Ye. Adaptive wavelet-neuro-fuzzy network in the forecasting and emulation tasks / Ye. Bodyanskiy, I. Pliss, O. Vynokurova // Int. J. on Information Theory and Applications. – 2008. – 15 (1). – P. 47-55.

67. Бодянский, Е.В. Гибридная вэйвлет-нейро-фаззи-архитектура, основанная на адаптивных вэйвлонах в задачах интеллектуальной обработки данных / Е.В. Бодянский, Е.А. Винокурова // Матеріали міжнародної науково-технічної конференції «Автоматизація: проблеми, ідеї, рішення». – Севастополь: СевНТУ. – 2009. – С. 9-12.

68. Бодянський, Є.В. Інтелектуальна обробка даних на основі гібридної вейвлет-нейро-фаззі системи на адаптивних W-нейронах / Є.В. Бодянський, О.А. Винокурова // Наукові праці: Науково-методичний

журнал. – Вип. 104. – Т. 117. – Комп'ютерні технології. – Миколаїв: Вид-во ЧДУ ім. Петра Могили, 2009. – С. 88-98.

69. Abiyev, R.H. Fuzzy wavelet neural networks for identification and control of dynamic plants - A novel structure and a comparative study / R.H. Abiyev // IEEE Transactions on Industrial Electronics. – 2008. – 55(8). – P. 3133-3140.

70. Bodyanskiy, Ye. Radial-basis-fuzzy-wavelet-neural network with adaptive activation-membership function / Ye. Bodyanskiy, O. Vynokurova, E. Yegorova // Int. J. on Artificial Intelligence and Machine Learning. – 2008. – Vol. 8. – II. – P. 9-15.

71. Bodyanskiy, Ye. Hybrid radial-basis neuro-fuzzy wavelon in the non-stationary sequences forecasting problems / Ye. Bodyanskiy, O. Vynokurova // Proc. of 2nd Int. Conf. on Inductive Modelling. – 2008. – Kyiv. – P. 144-147.

72. Бодянский, Е.В. Адаптивный вэйвлон и алгоритм его обучения / Е.В. Бодянский, Е.А. Винокурова // Управляющие системы и машины. – 2009. – 1 (219). – С. 47-53.

73. Narendra, K.S. Identification and control of dynamic systems using neural networks / K.S. Narendra, K. Parthasarathy // IEEE Trans. on Neural Networks, 1990, 1. – P. 4-26.

74. Chiu, S. Fuzzy model identification based on cluster estimation / S. Chiu // Journal of Intelligent and Fuzzy Systems. – 1994. – 2. – №3. – P. 267-278.

75. Juang, C.-F. A recurrent self-organizing neural fuzzy inference network / C.-F. Juang, C.-T. Lin // IEEE Trans. Neural Networks. – 10 (4). – 1999 – P. 828-845.

76. Juang, C.-F. A TSK-type recurrent fuzzy network for dynamic systems processing by neural network and genetic algorithm / C.-F. Juang // IEEE Trans. Fuzzy Systems. – 10 . – 2002. – P. 155-170.

77. Abiyev, R.H. Identification and control of dynamic plant using fuzzy wavelet neural networks / R.H. Abiyev, O. Kaynak // Proc. of IEEE Int. Symposium on Intelligent Control. – USA: San Antonio. – 2008. – P.1295-1301.

78. Bezdek, J.C. Fuzzy Models and Algorithms for Pattern Recognition and Image Processing / J.C. Bezdek, J. Keller, R. Krisknapuram, N.R. Pal.– N.Y.: Springer Science+Business Media, Inc., 2005. – 776 p.

79. Keller, J. M. A fuzzy k-nearest neighbor algorithm / J.M. Keller, M.R. Gray, J.A. Givens, Jr. // IEEE Transactions on Systems, Man, and Cybernetics. – 1985. – V. 15, № 4. – P. 580-585.

80. Bodyanskiy, Ye. Matrix neuro-fuzzy self-organizing clustering network / Ye. Bodyanskiy, V. Volkova, M. Skuratov // Scientific J. of Riga Technical University “Computer Science, Information Technology and Management Sci.”. – 2011. - №49. – P. 54-58.

81. Krishnapuram, R. A possibilistic approach to clustering / R. Krishnapuram, J. Keller // IEEE Trans. on Fuzzy Systems. – 1993. – Vol. 1. – P. 98–110.

82. Krishnapuram, R. The possibilistic C-means algorithm: Insights and recommendations / R. Krishnapuram, J. Keller // IEEE Transactions on Fuzzy Systems. – 1996. – V. 4, № 3. – P. 385-393.

83. Pal, N.R. A possibilistic fuzzy C-means clustering algorithm / N.R. Pal, K. Pal, J. Keller, J.C. Bezdek // IEEE Transactions on Fuzzy Systems. – 2005. – V. 13, № 4. – P. 517-530.

84. Bodyanskiy, Ye. Adaptive matrix fuzzy c-means clustering / Ye. Bodyanskiy, V. Volkova, M. Skuratov // Proc. 19th East-West Fuzzy-Kolloquium. – Zittau-Goerlitz: HS, 2012. – P.111-116.

85. UCI Repository of machine learning [Електронний ресурс] / Irvine: University of California, Department of Information and Computer

Science. – Режим доступу: www / URL: http://archive.ics.uci.edu/ml/datasets.html. – Загол. з екрану.

86. Fahlman, S. E. The cascade-correlation learning architecture. Advances in Neural Information Processing Systems / S. E. Fahlman, C. Lebiere ; Ed. by D. S. Touretzky. – San Mateo, CA : Morgan Kaufman, 1990. – P. 524–532.

87. Prechelt, L. Investigation of the Cascor family of learning algorithms / Prechelt L. // Neural Networks. – 1997. – vol. 10. – P. 885–896.

88. Schalkoff, R. J. Artificial Neural Networks / Schalkoff R. J. – N. Y. : The McGraw-Hill Comp., 1997. – 528 p.

89. Avedjan, E. D. Cascade neural networks / E. D. Avedjan, G. V. Barkan, I. K. Levin // Avtomatika i telemekhanika. – 1999. – No. 3. – P. 38–55.

90. Bodyanskiy, Ye. The cascaded orthogonal neural network / Ye. Bodyanskiy, A. Dolotov, I. Pliss, Ye. Viktorov ; Eds. by K. Markov, K. Ivanova, I. Mitov // Information Science & Computing. – Sofia, Bulgaria : FOI ITHEA. – 2008. – Vol. 2. – P. 13–20.

91. Bodyanskiy, Ye. The cascaded neo-fuzzy architecture and its on-line learning algorithm / Ye. Bodyanskiy, Ye. Viktorov ; Eds. by K. Markov, P. Stanchev, K. Ivanova, I. Mitov // Intelligent Processing. – 9. – Sofia : FOI ITHEA, 2009. – P. 110–116.

92. Bodyanskiy, Ye. The cascaded neo-fuzzy architecture using cubicspline activation functions / Ye. Bodyanskiy, Ye. Viktorov // Int. J. «Information Theories & Applications». – 2009. – vol. 16, No. 3. – P. 245–259.

93. Bodyanskiy, Ye. The cascade growing neural network using quadratic neurons and its learning algorithms for on-line information processing / Ye. Bodyanskiy, Ye.Viktorov, I. Pliss ; Eds. by G. Setlak, K.

Markov // Intelligent Information and Engineering Systems. – 13. – Rzeszov-Sofia : FOI ITHEA, 2009. – P. 27–34.

94. Kolodyazhniy, V. Cascaded multi-resolution spline-based fuzzy neural / V. Kolodyazhniy, Ye. Bodyanskiy ; Eds. by P. Angelov, D. Filev, N. Kasabov // Proc. Int. Symp. on Evolving Intelligent Systems. – Leicester, UK : De Montfort University, 2010. – P. 26–29.

95. Bodyanskiy, Ye. Cascaded GMDH-wavelet-neuro-fuzzy network / Ye. Bodyanskiy, O. Vynokurova, N. Teslenko // Proc 4th Int. Workshop on Inductive Modelling «IWIM 2011». – Kyiv, 2011. – P. 22–30.

96. Bodyanskiy, Ye. Hybrid cascaded neural network based on wavelet-neuron / Ye. Bodyanskiy, O. Kharchenko, O. Vynokurova // Int. J. Information Theories & Applications. – 2011. – vol. 18, No. 4. – P. 335–343.

97. Bodyanskiy, Ye. Evolving cascaded neural network based on multidimensional Epanechnikov's kernels and its learning algorithm / Ye. Bodyanskiy, P. Grimm, N. Teslenko // Int. J. Information Technologies & Knowledge. – 2011. – vol. 5, No. 1. – P. 25–30.

98. Kruschke, J. K. Benefits of gain: speed learning and minimum layers backpropagation networks / J. K. Kruschke, J. R. Movellan // IEEE Trans. on Syst., Man. And Cybern. – 1991. – vol. 21. – P. 273–280.

99. Chan, L. W. An adaptive learning algorithm for backpropagation networks / L. W. Chan, F. Fallside // Computer Speech and Language. – 1987. – vol. 2. – P. 205– 218.

100. Silva, F. M. Speeding up backpropagation / F. M. Silva, L. B. Almeida ; Ed. by R. Eckmiller // Advances of Neural Computers. – North-Holland : Elsevier Science Publishers. – B. V., 1990. – P. 151–158.

101. Veitch, A. C. A modified quickprop algorithm / A. C. Veitch, G. Holmes // Neural Computation. – 1991. – vol. 3. – P. 310– 311.

102. Bodyanskiy, Ye. An adaptive learning algorithm for a neurofuzzy network / Ye. Bodyanskiy, V. Kolodyazhniy, A. Stephan ; Ed. by

B. Reusch // Computational Intelligence: Theory and Applications. – Berlin-Heidelberg-New-York: Springer, 2001. – P. 68–75.

103. Otto, P. A new learning algorithm for a forecasting neurofuzzy network / P. Otto, Ye. Bodyanskiy, V. Kolodyazhniy // Integrated Computer-Aided Engineering. – 2003. – vol. 10, No. 4. – P. 399–409.

104. Kaczmarz, S. Angenaeherte Ausloesung von Systemen linearer Gleichungen / Kaczmarz S. // Bull. Int. Acad. Polon. Sci. – 1937. – Let. A. – P. 355–357. 24.

105. Kaczmarz, S. Approximate solution of systems of linear equations / Kaczmarz S. // Int. J. Control. – 1993. – vol. 53. – P. 1269–1271.

106. Widrow, B. / Adaptive switching circuits / Widrow B., Hoff Jr. M. E. // 1960 URE WESCON Convention Record. – N. Y. : IRE, 1960. – Part 4. – P. 96–104.

107. Бодянский, Е. Адаптивно прогнозиране на нестационарни процеси / Е. Бодянский, И. Плисс, Н. Маджаров // Автоматика и изчислительна техника. – 1983. – № 6. – С. 5–12.

108. Бодянский, Е. В. Адаптивное обобщенное прогнозирование многомерных случайных последовательностей / Е. В. Бодянский, И. П. Плисс, Т. В. Соловьева // Докл. АН УССР. – 1989. – Сер.А. – №m9. – С. 73–75.

109. Bodyanskiy, Ye. Adaptive generalized forecasting of multivariate stochastic signals / Ye. Bodyanskiy, I. Pliss // Proc. Latvian Sign. Proc. Int. Conf. – Riga, 1990. – V. 2. – P. 80–83.

110. Gan G. Data Clustering: Theory, Algorithms and Application / G. Gan, Ch. Ma, J.Wu. - Philadelphia: SIAM, 2007. - 455 p.

111. Xu R. Clustering, IEEE Press Series on Computational Intelligence / R. Xu, D.C. Wunsch. - Hoboken, NJ: John Wiley & Sons, Inc., 2009. - 370 p.

112. Pelleg D., Moor A. X-means: extending K-means with efficient estimation of the number of clusters. - Proc. 17th Int. Conf. on Machine Learning. - San Francisco: Morgan Kaufmann, 2000. - P.727-730.

113. Ishioka T. An expansion of X-means for automatically determining the optimal number of clusters.–Proc. 4th IASTED Int. Conf. Computational Intelligence.– Calgary, Alberta, 2005. – P.91-96.

114. Bifet A. Adaptive Stream Mining: Pattern Learning and Mining from Evolving Data Streams / A. Bifet. - Amsterdam: - IOS Press, 2010. - 224 p.

115. Kohonen, T. Self-Organizing Maps / T.Kohonen. - Berlin: Springer-Verlag, 1995.- 362 p.

116. Strehl A., Ghosh J. Cluster Ensembles – A knowledge reuse framework for combining multiple partitions. – Journal of Machine Learning Research. – 2002. – February. – P.583-617.

117. Topchy A., Jain A.K., Punch W. Clustering ensembles: models of consensus and weak partitions. – IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2005. – 27. – P.1866-1881.

118. Alizadeh, H., Minaei-Bidgoli, B., Parvin, H. To improve the quality of cluster ensembles by selecting a subset of base clusters. - Journal of Experimental & Theoretical Artificial Intelligence. - 2013. – 26. – P.127-150.

119. Charkhabi M., Dhot T., Mojarad S.A. Cluster ensembles, majority vote, voter eligibility and privileged voters. – Int. Journal of Machine Learning and Computing. – 2014. – 4. – P.275-278

120. Bodyanskiy Ye. Computational intelligence techniques for data analysis - Lecture Notes in Informatics– P-72. – Bonn : GI, 2005. – P.15–36.

121. Бодянский Е.В., Руденко О.Г. Искусственные нейронные сети: архитектуры, обучение, применения. – Харьков : ТЕЛЕТЕХ, 2004. – 372 с.

122. Аналіз та обробка потоків даних засобами обчислювального інтелекту : монографія / Є. В. Бодянський, Д. Д. Пелешко, О.А. Винокурова, С. В. Машталір, Ю. С. Іванов. - Львів : Вид-во Львів. політехніки, 2016. - 235 с.

123. Murphy P. M. UCI Repository of machine learning databases. / P. M. Murphy, D. Aha. – URL: http://www.ics.uci.edu/mlearn/MLRepository.html. CA: University of California, Department of Information and Computer Science, 1994.

124. Kohonen, T. Self-Organizing Maps / T. Kohonen // Berlin: Springer-Verlag. – 1995. – 362 p.

125. Bezdek, J.-C. Pattern Recognition with Fuzzy Objective Function Algorithms [Text] / J. C. Bezdek. – N.Y.: Plenum Press, 1981. – 272p.

126. Tsao, E.C.-K. Fuzzy Kohonen clustering networks [Text] / E.C.-K. Tsao, J.C. Bezdek, J.C. Tsao, N.R. Pal // Pattern Recognition. – 1994. – №27. – pp. 757–764.

127. Pascual – Marqui, R. D. Smoothly distributed fuzzy C-means: a new self-organizing map / R. D. Pascual – Marqui, A.D. Pascual – Montano, K. Kochi, J.M. Caroso // Pattern Recognition. – 2001. – №34. – pp. 2395–2402.

128. MacDonald, D., Fyfe C. Clustering in data space and feature space : ESANN'2002 Proc. European Symp. on Artificial Neural Networks. Bruges (24-26 April 2002). – Belgium. – 2002. – pp. 137-142.

129. Girolami, M. Mercer kernel-based clustering in feature space / M. Girolami // IEEE Trans. on Neural Networks. – 2002. – Vol. 13. – №3. – pp. 780-784.

130. Camastra, F. A novel kernel method for clustering / F. Camastra, A. Verri // IEEE Trans. on Pattern Analysis and Machine Intelligence. – 2005. –№5. – pp. 801-805.

131. Schölkopf, B. Learning with Kernels / B. Schölkopf, A. Smola // Cambridge, M.A.: MIT Press. – 2002. – 648 p.

132. Kacprzyk, J. Springer Handbook of Computational Intelligence / J. Kacprzyk, W. Pedrycz. – Berlin Heidelberg: Springer – Verlag, 2015. – 1634 p.

133. Haykin, S. Neural Networks and Learning Machines / S. Haykin – N.Y. :Prentice Hall, 2009. – 1634 p.

134. Cortes, C. Support Vector Networks / C. Cortes, V. Vapnik // Machine Learning. – 1995. – №20. – pp. 273–297.

135. Parzen, E. On the estimation of a probability density function and the mode / E. Parzen // Ann. Math. Statist. – 1962. – №38. – pp. 1065-1076.

136. Specht, D.F. A general regression neural network / D.F. Specht // IEEE Trans. on Neural Networks. – 1991. – Vol 2. – pp. 568-576.

137. Zahirniak, D. Pattern recognition using radial basis function network. / D. Zahirniak, R. Chapman, S. Rogers, B. Suter, M. Kabrisky, V. Piati // Proc 6th Ann. Aerospace Application of Artificial Intelligence Conf. – Dayton, OH. – 1990. – pp. 249-260.

138. Cover, T.M. Geometrical and statistical properties of systems of linear inequali-ties with applications in pattern recognition / T.M. Cover // IEEE Trans. on Electronic Computers. – 1965. – №14. – pp. 326-334.

139. Angelov, P. Evolving Rule-based Models: A Tool for Design of Flexible Adaptive Systems / P. Angelov // Heidelberg-New York: Springer-Verlag. – 2002. – 211 p.

140. Kasabov, N. Evolving Connectionist Systems / N. Kasabov – London: Springer-Verlag. – 2003 – 307 p.

141. Angelov, P. Evolving computational intelligence systems / P. Angelov, N. Kasabov // Proc. 1st Int. Workshop on Genetic Fuzzy Systems. – Granada, Spain. – 2005. – pp. 76-82.

142. Lughofer, E. Evolving Fuzzy Systems – Methodologies and Applications / E. Lughofer. – Studies in Fuzziness and Soft Computing. – Springer-Berlin. – 2011. – 410 p.

143. Льюинг Л. Идентификация систем. Теория для пользователя. – Москва: Наука, 1991. – 432с.

144. Kaczmarz S. Approximate solution of system of linear equations // Int. J. Control. – 1993. – 57. – No 5. – P. 1269-1271.

145. Руденко О.Г., Бодянский Е.В., Плисс И.П. Адаптивный алгоритм прогнозирования случайных последовательностей // Автоматика. – 1979. – №1. – С. 54-57.

146. Бодянський Є.В., Плісс І.П., Соловйова Г.В. Синтез квазіпрямих адаптивних регуляторів // Доповіді АН УРСР. – Сер.А. – 1987. – №1. – С.59-61.

147. Шильман С.В. Итеративное линейное оценивание с регулируемым объемом предистории // Автоматика и телемеханика. – 1983. – №5. – С. 93-98.

148. Михеев А.П., Шильман С.В. Итеративная фильтрация с регулируемым взвешенным накоплением информации // Динамические системы: управление, адаптация и оптимизация. – Горький, 1983. – С.94-106.

149. Chow E.Y., Willsky A.S. Issues in the development of a general design algorithm for reliable faiture detection // Proc. 19-th IEEE Conf. Decis. Ant Contr. – Albuquerque, 1980. – P. 1006-1012.

150. Montgomery D.C., Johnson I.A., Yardiner J.S. Forecasting and Time Series Analysis. – N.Y.: McYraw-Hill, 1990. – 394p.

151. Чуев Ю.В., Михайлов Ю.Б., Кузьмин В.И. Прогнозирование количественных характеристик процессов. – Москва: Сов. радио, 1975. – 400с.

152. Льюис К.Д. Методы прогнозирования экономических показателей. – М.: Финансы и статистика, 1986. – 133с.

153. Brown R.G. Smoothing, Forecasting, and Prediction of Discrete Time Series. – N.Y.: Preutice Hall, 1963. – 468p.

154. Trigg D.W., Leach A.G. Exponential smoothing with an adaptive response rate // Operational Research Quarterly. – 1967. – V.18. – No1. – P.53-59.

155. Изерман Р. Цифровые системы управления. – Москва: Мир, 1984. – 541с.

156. Иберла К. Факторный анализ. – Москва: Статистика, 1980. – 398с.

157. Cichocki A., Unberhauen R. Neural Networks for Optimization and Signal Processing. – Stuttgart: Teubner, 1993. – 526p.

158. Растригин Л.А., Пономарев Ю.П. Экстраполяционные методы проектирования и управления – М.: Машиностроение, 1986. – 120 с.

159. Алберт А. Регрессия, псевдоинверсия и рекуррентное оценивание. – М.: Наука, 1977. – 224 с.

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

161. Чебышев, П.Л. Об интерполировании. Избранные труды / П.Л. Чебышев. – М.: Физматгиз, 1965. – 661 с.

162. Бодянський Є.В. Виявлення змін у потоці відеоданих на основі аналізу багатовимірних часових рядів / Бодянський Є.В.,

Машталір С.В.// Доповіді Національної академії наук України. –2012. – №11. – С.30-33.

163. Гончаренко, М.О. Детектирование изменений сцены в потоке видеоданных / М.О. Гончаренко, С.В. Машталир // Электротехнические и компьютерные системы. – 2012. – №7(83). – С. 143-147.

164. Машталир, С.В. Анализ пространственно-временной сегментации видеопотоков / Машталир С.В. // Междунар. науч. конф. Интеллектуальные системы принятия решений и проблемы вычислительного интеллекта ISDMCI’2016: сб. научных трудов, Железный Порт – 24-28 Мая. – Херсон: ПП Вишемирський В. С., – 2016. – С.295-297.

165. On-line video segmentation using methods of fault detection in multidimensional time sequences / Y. Bodyanskiy, D. Kinoshenko, S. Mashtalir, O. Mikhnova // International Journal of Electronic Commerce Studies. – 2012. – Vol. 3, No. 1. – P. 1-20.

166. Kinoshenko, D. temporal video segmentation via spatial image segmentation /D. Kinoshenko, S. Mashtalir, V. Shlyakhov// International Journal "Information Technologies & Knowledge". – 2013. – Vol.7, No 3. – P. 212-219.

167. Мантула, Е.В. Адаптивное прогнозирование временных рядов при неравностоящих наблюдениях / Е.В. Мантула, С.В. Машталир // Бионика интеллекта. – 2013. – № 2 (81). – С.53-56.

168. Мантула, Е.В. Адаптивная полиномиальная нейросетевая прогнозирующая модель временных рядов и ее обучение / Е.В. Мантула, С.В. Машталир // Восточно-Европейский журнал передовых технологий. Математика и кибернетика – прикладные аспекты. – 2014. – № 2/4(68) – С.16-20.

169. Mantula, E. Method of adaptive forecasting based on multidimensional linear extrapolation / E. Mantula., S. Mashtalir // International Journal of Research in Engineering and Science. – 2013. – Vol., 1 No. 4. – P. 31-36.

170. Mashtalir S., Mashtalir V. Sequential temporal video segmentation via spatial image partitions / IEEE First International Conference on Data Stream Mining & Processing (DSMP), 2016. – p. 239-242.

171. Mashtalir S., Mikhnova O. Detecting Significant Changes in Image Sequences // In: Hassanien A., Mostafa Fouad M., Manaf A., Zamani M., Ahmad R., Kacprzyk J. (eds) Multimedia Forensics and Security. Springer 2017. – p.161-191.

172. Абрамов, С.К. Мера содержания фона на основе энтропии для поиска и сортировки изображений в базах данных / С.К. Абрамов, В.В. Лукин, Н.Н. Пономаренко // Радіоелектронні і комп'ютерні системи. – 2007. – № 2 (21). – С. 24-28.

173. Алберт, А. Регрессия, псевдоинверсия и рекуррентное оценивание; пер с англ. под ред. Я.З. Цыпкина. – М.: Наука, 1977. – 224 с.

174. Богучарский, С. И. Анализ текстур в последовательности изображений на основе векторного квантования // С.И. Богучарский, С.В. Машталир // Радиоэлектроника, информатика, управление. – 2014. – №2(31). – С. 94-99.

175. Ishioka, T. An expansion of X-means for automatically determining the optimal number of clusters / T. Ishioka // Computational Intelligence (CI 2009): Proc. of 4th IASTED Int. Conf., Honolulu, Hawaii, USA, 17-19 August, 2009, Calgary: ACTA Press. – 2009. – P. 91-96.

176. Ishioka, T. Extended k-means with an efficient estimation of the number of clusters / T. Ishioka // Intelligent Data Engineering and

Automated Learning – IDEAL 2000. Data mining, Financial Engineering, and Intelligent Agents: Proc. of 2nd Int. Conf. (K.S. Leung, L.-W, Chan eds), Hong Kong, China, 13-15 December, 2000, Lecture Notes in Computer Science, vol. 1983, Berlin-Heidelberg New York: Springer Verlag. – 2000. – P. 17-22.

177. Pelleg, D. X-means: Extending k-means with efficient estimation of the number of clusters / D. Pelleg, A. Moore // Machine Learning (ICML 2000): Proc. of 17th Int. Conf., Stanford, CA, USA, 29 June – 2 July, 2000, San Francisco: Morgan Kaufmann Publishers Inc., 2000. – P. 727-730.

178. Bozdogan H. Model selection and Akaike's information criterion (AIC): The general theory and its analytical extensions // Psychometrica. – 1987. – Vol. 52, Iss. 3 – P. 345-370.

179. Kass, R.E., Wasserman L. A reference Bayesian test for nested hypotheses and its relationship to the Schwarz criterion / Kass, R.E., L.Wasserman // Journal of the American Statistical Association. – 1995. – Vol. 90, No. 40. – P. 928-934.

180. Schwarz, G. Estimation the dimension of a model/ G. Schwarz // The Annals of Statistics. – 1978. – Vol.6, Iss. 2. – P. 461-464.

181. Xu, R. Clustering / R. Xu, D.C. Wunsch. – Hoboken: John Wiley&Sons, 2008. – 358 p.

182. Bezdek J. Pattern Recognition with Fuzzi Objective Functions Algorithms. – N.Y.: Plenum Press., 1981. – 272 p.

183. Bodyanskiy, Ye. Matrix neuro-fuzzy self-organizing clastering network/ Ye. Bodyanskiy, V. Volkova, M. Skuratov // Sci. J. of Riga Techn. Uni. 'Computer Science, Information Technology and Management Science' – 2011. – Vol. 49. – P. 54-58.

184. Yuha, S. CURE: an efficient clustering algorithm for large databases / S. Yuha, R. Rastogi, K. Shim // Management of Data: Proc. of the

1998 ACM SIGMOD Int. Conf. (A. Tiwary, M. Franklin. eds.), Seattle: 1-4 June, 1998, N.Y.: ACM Press, 1998. – P. 73-84.

185. Zhang, T. BIRCH: An efficient data clustering method for very large databases / T. Zhang, P. Ramakrishnan, M. Livny // Management of Data: Proc. of the 1996 ACM SIGMOD Int. Conf (J. Widom. ed.), Montreal: 4-6 June, 1996, N.Y.: ACM Press, 1996. – P. 103-114.

186. Kaufman, L. Finding groups in data: an introduction to cluster analysis / L. Kaufman, P.J. Rousseeuw. – N.Y.: John Wiley&Sons, 1990. – 342 p.

187. Bogucharskiy, S. On matrix modification of CLARANS clustering method in large video surveillance databases / S. Bogucharskiy, V. Mashtalir / Вісник Національного університету «Львівська політехніка». Комп'ютерні науки та інформаційні технології. – Львів: Видавництво Львівської політехніки, 2014. – № 800. – С. 211-216.

188. Hansen, P. J-means: a new local search heuristic for minimum sum of squares clustering / P. Hansen, N. Mladenović // Pattern Recognition. – 2001. – Vol. 34, Iss. 2. – P. 405-413.

189. Bogucharskiy, S.I. Image segmentation with fuzzy J-means method // S.I. Bogucharskiy, A.G. Kagramanyan, O.D. Mikhnova // Системні технології. Рег. міжвуз. збірник наук. праць. – Дніпропетровськ: НметАУ, ІВК «Системні технології». – 2014. – Вип. 6 (95). – С.27-34.

190. Bogucharskiy, S. Fuzzy J-Means image segmentation / S. Bogucharskiy, V. Mashtalir, O. Mikhnova // Advances in Data Science: proc. International Workshop and Networking Event, Poland, Holny Mejera, 6-8 May 2015. – Bialystok: BUT, 2015. – P. 15-16.