Харківський національний університет радіоелектроніки

На правах рукопису

Копаліані Дар'я Сергіївна

УДК 004.032.26

Еволюційні нейро-фаззі мережі з каскадною структурою для інтелектуального аналізу данних

05.13.23 — системи та засоби штучного інтелекту

Дисертація на здобуття наукового ступеня
кандидата технічних наук

Науковий керівник
Бодянський Євгеній Володимирович,
доктор технічних наук, професор

Харків — 2015

ЗМІСТ

РОЗДІЛ 1

ОГЛЯД СТАНУ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

РОЗДІЛ 2
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 [23, 33]. Ця ідея лежить в основі систем обчислювального інтелекту, що еволюціонують [37, 47].

Мабуть, найбільш відомою реалізацією цього підходу є каскадно-кореляційні нейронні мережі [27, 49, 55], привабливі високою ефективністю та простотою налаштування як синаптичних вагових коефіцієнтів, так і топології мережі. Така мережа напочатку містить лише один пул (ансамбль) нейронів, які навчаються назалежно 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 процесс 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 ар-

хітектуру новими каскадами, доки вона не досягне бажаної якості вирішення завдання для заданого набору даних.

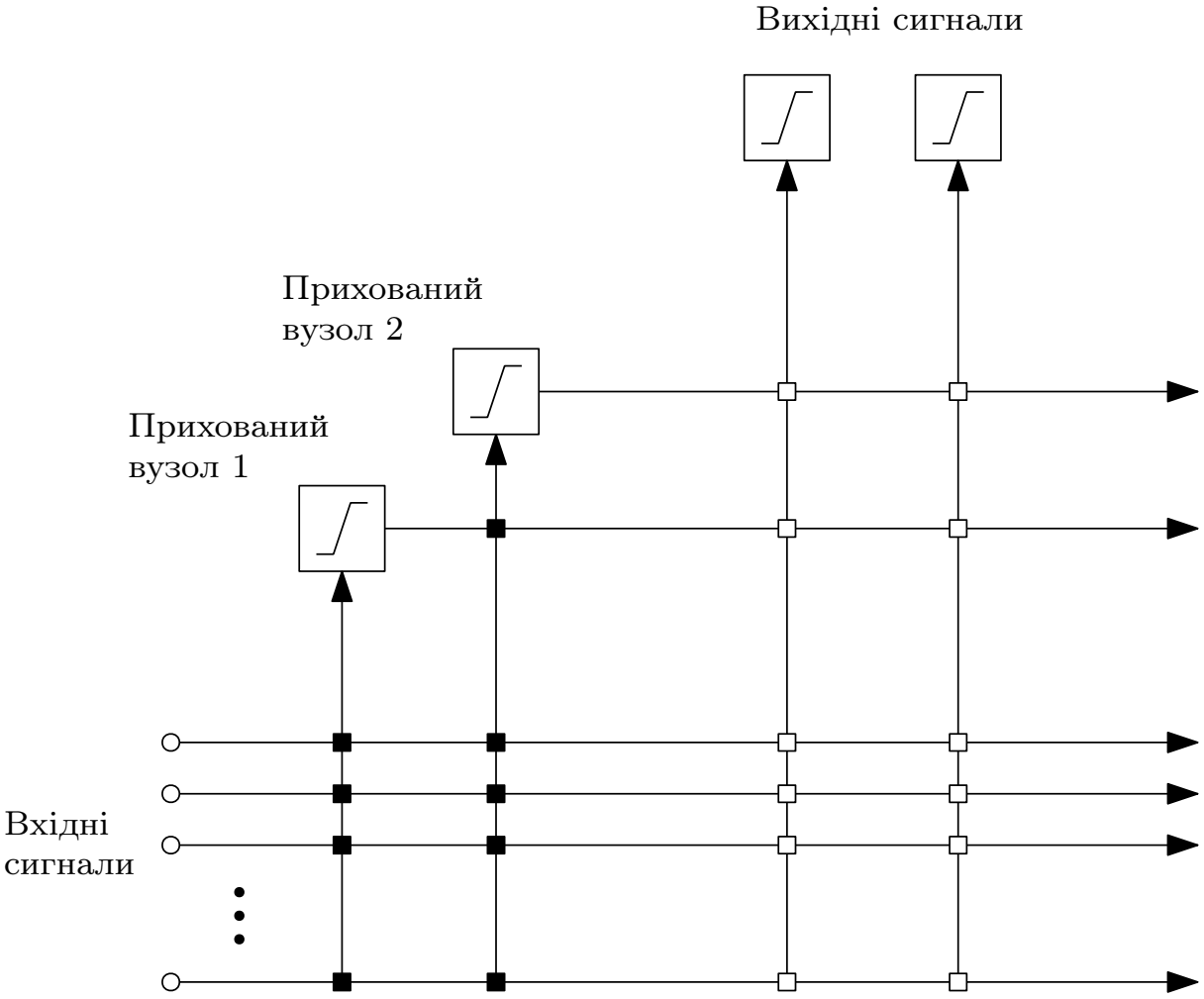

Рис. 2.1. Архітектура каскадної системи (за Фальманом та Лєб'єром) після додавання двох прихованих вузлів. Вхідні сигнали, що надходять до вертикальних ліній, сумуються; вагові коефіцієнти, позначені □, – зафіксовані, позначені ■, – налаштовуються

Автори найпопулярнішої каскадної нейронної мережі, що еволюціонує, CasCorLA (схему наведено на рис. 2.1), Фальман та Лєб'єр, використовували елементарні персептрони Розенблатта з традиційними сигмоїдальними функціями активації і коригували синаптичні вагові коефіцієнти за допомогою QuickProp-алгоритму [27], 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 багатоепохового навчання [11, 12, 15–17, 34, 41, 74] 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.

2.1. Архітектура оптимізованої каскадної нейронної мережі

Архітектура пропонованої гібридної системи з оптимізованим пулом нейронів у кожному каскаді наведена на рис. 2.2.

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

$$x(k) = (x_1(k), x_2(k), \ldots, x_n(k))^T, \tag{2.1}$$

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

Ці сигнали подаються на входи кожного нейрона в мережі $N_j^{[m]}$ ($j = 1, 2, \ldots, q$ – кількість нейронів у тренувальному пулі, $m = 1, 2, ...$ – номер каскаду) з вихідним сигналом $\hat{y}_j^{[m]}(k)$. Далі вихідні сигнали кожного каскаду $\hat{y}_j^{[m]}(k)$ 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

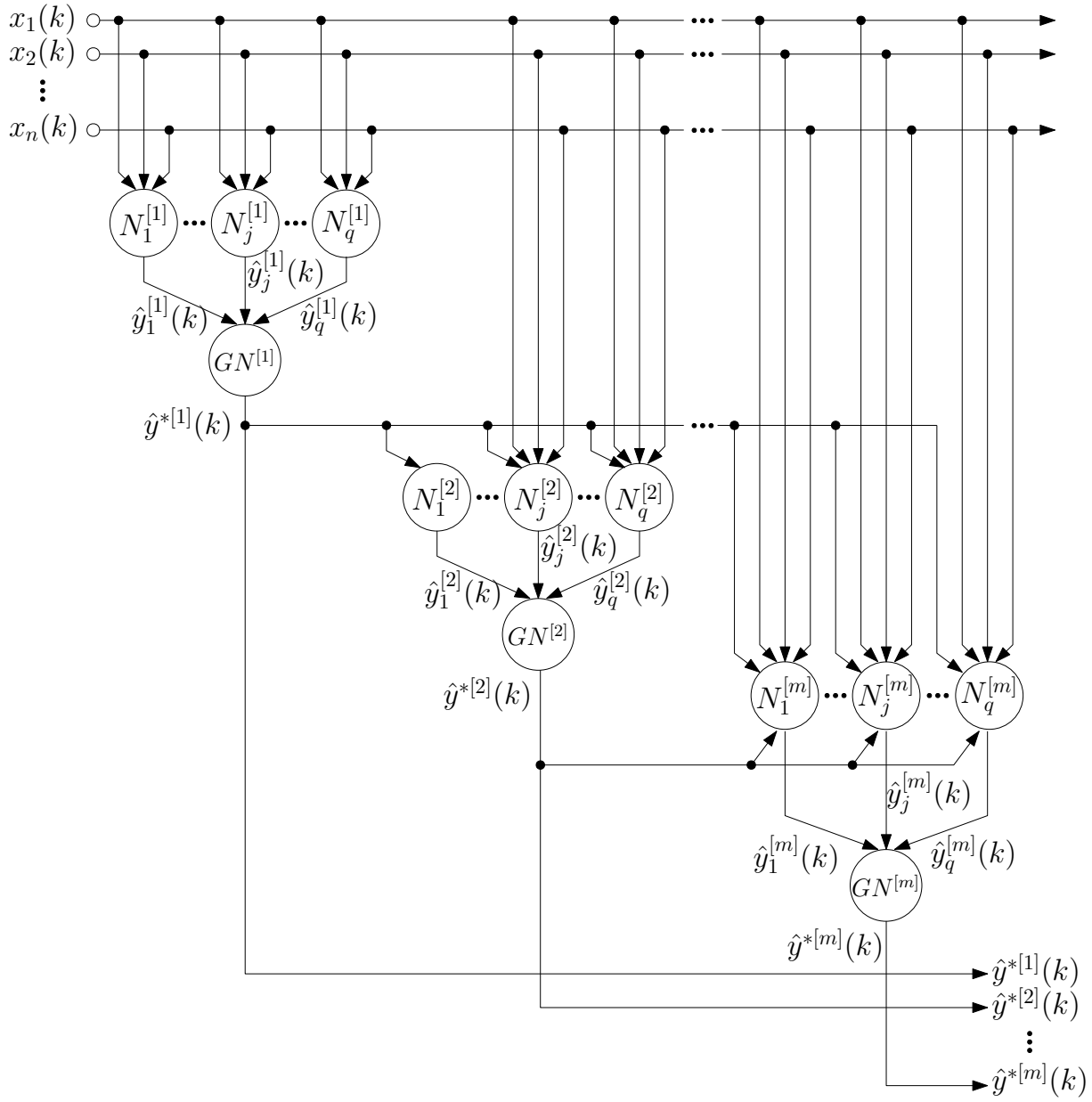

Рис. 2.2. Архітектура гібридної системи з оптимізованим пулом нейронів

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.

2.2. Навчання елементарних персептронів Розенблатта у каскадній оптимізованій системі

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

$$0 < \sigma_j^m\left(\gamma_j^{[m]} u_j^{[m]}\right) = \frac{1}{1 + e^{-\gamma_j^{[m]} u_j^{[m]}}} < 1, \tag{2.2}$$

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

$$\hat{y}_j^{[1]} = \sigma_J^{[1]}\left(\gamma_j^{[1]} \sum_{i=0}^{n} w_{ji}^{[1]} x_i\right) = \sigma_j^{[1]}\left(\gamma_j^{[1]} w_j^{[1]T} x\right), \tag{2.3}$$

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

Вихідні сигнали другого каскаду дорівнюватимуть

$$\hat{y}_j^{[2]} = \sigma_J^{[2]}\left(\gamma_j^{[2]}\left(\sum_{i=0}^{n} w_{ji}^{[2]} x_i + w_{j,n+1}^{[2]} \hat{y}^{*[1]}\right)\right), \tag{2.4}$$

вихідні сигнали m-ого каскаду матимуть вигляд

$$\begin{aligned} \hat{y}_j^{[m]} &= \sigma_j^{[m]}\left(\gamma_j^{[m]}\left(\sum_{i=0}^{n} w_{ji}^{[m]} x_i + w_{j,n+1}^{[m]} \hat{y}^{*[1]}\right.\right. \\ &\left.\left. + w_{j,n+2}^{m} \hat{y}^{*[2]} + \cdots + w_{j,n+m-1}^{[m]} \hat{y}^{*[m-1]}\right)\right) \\ &= \sigma_j^{[m]}\left(\gamma_j^{[m]} \sum_{i=0}^{n+m-1} w_{ji}^{[m]} x_j^{[m]}\right) = \sigma_j^{[m]}\left(w_j^{[m]T} x^{[m]}\right), \end{aligned} \tag{2.5}$$

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

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

У якості критерію навчання можна використовувати загальноприйняту квадратичну функцію

$$\begin{aligned} E_j^{[m]} &= \frac{1}{2}\left(e_j^{[m]}(k)\right)^2 = \\ &= \frac{1}{2}\left(y(k)-\hat{y}_j^{[m]}(k)\right)^2 = \\ &= \frac{1}{2}\left(y(k)-\sigma_j^{[m]}\left(\gamma_j^{[m]} w_j^{[m]T} x^{[m]}(k)\right)\right)^2, \end{aligned} \tag{2.6}$$

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

Градієнтну оптимізацію критерію (2.6) відносно $w_j^{[m]}$ можна записати у вигляді

$$\begin{aligned} w_j^{[m]}(k+1) =& w_j^{[m]} + \eta_j^{[m]}(k+1) e_j^{[m]}(k+1)\gamma_j^{[m]}\hat{y}_j^{[m]}(k+1) \\ &\times\left(1-\hat{y}_j^{[m]}(k+1)\right)x^{[m]}(k+1) = \\ =& w_j^{[m]}(k) + \eta_j^{[m]}(k+1) e_j^{[m]}(k+1)\gamma_j^{[m]} J_j^{[m]}(k+1), \end{aligned} \tag{2.7}$$

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

Мінімізувати критерій (2.6) відносно $\gamma_j^{[m]}$ можна за допомогою алгоритму Крушке-Мовеланна [42]

$$\begin{aligned} \gamma_j^{[m]}(k+1) =& \gamma_j^{[m]}(k) + \eta_j^{[m]}(k+1) e_j^{[m]}(k+1)\hat{y}_j^{[m]}(k+1) \\ &\times\left(1-\hat{y}_j^{[m]}(k+1)\right)u_j^{[m]}(k+1). \end{aligned} \tag{2.8}$$

Поєднуючи (2.7) та (2.6), отримаємо алгоритм навчання для j-ого нейрону m-ого каскаду

$$\frac{w_j^{[m]}(k+1)}{\gamma_j^{[m]}(k+1)} = \frac{w_j^{[m]}(k)}{\gamma_j^{[m]}(k+)} + \eta_j^{[m]}(k+1)e_j^{[m]}(k+1)\hat{y}_j^{[m]}(k+1) \times \left(1 - \hat{y}_j^{[m]}(k+1)\right)\left(\frac{\gamma_j^{[m]}x^{[m]}(k+1)}{u_j^{[m]}(k+1)}\right), \tag{2.9}$$

або, вводячи нові змінні, у більш компактній формі

$$\begin{aligned}\tilde{w}_j^{[m]}(k+1) &= \tilde{w}_j^{[m]}(k) + \eta_j^{[m]}(k+1)e_j^{[m]}(k+1)\hat{y}_j^{[m]}(k+1)\tilde{x}^{[m]}(k+1) \\ &= \tilde{w}_j^{[m]}(k) + \eta_j^{[m]}(k+1)e_j^{[m]}(k+1)\tilde{J}_j^{[m]}(k+1).\end{aligned} \tag{2.10}$$

Використовуючи регуляризуючий параметр (momentum term) [19, 58, 61], можна удосконалити процесс корегування синаптичних вагових коефіцієнтів під час навчання. Тоді, замість критерію (2.6) слід використовувати функцію

$$\begin{aligned}E_j^{[m]}(k) =& \frac{\eta}{2}\left(e_j^{[m]}(k)\right)^2 \\ &+ \frac{1-\eta}{2}\left\|\tilde{w}_j^{[m]}(k) - \tilde{w}_j^{[m]}(k-1)\right\|^2, 0 < \eta \leq 1.\end{aligned} \tag{2.11}$$

Тоді алгоритм навчання приймає вигляд

$$\begin{aligned}\tilde{w}_j^{[m]}(k+1) =& \tilde{w}_j^{[m]}(k) \\ &+ \eta_j^{[m]}(k+1)\Bigg(\eta e_j^{[m]}(k+1)\tilde{J}_j^{[m]}(k+1) \\ &+ (1-\eta)\left(\tilde{w}_j^{[m]}(k) - \tilde{w}_j^{[m]}(k+1)\right)\Bigg),\end{aligned} \tag{2.12}$$

що є модифікацією процедури Сільви-Альмейди [58].

Доцільно вдосконалити алгоритм, використовуючи підхід, запропонований у [13], тоді алгоритм (2.12) набуває слідкуючих та фільтруючих властивостей. Таким чином, кінцева модифікація алгоритму набуваю вигляду

$$\begin{cases} \tilde{w}_j^{[m]}(k+1) = & \tilde{w}_j^{[m]}(k) + \dfrac{\eta e_j^{[m]}(k+1)\tilde{J}_j^{[m]}(k+1)}{r_j^{[m]}(k+1)} \\ & + \dfrac{(1-\eta)\Big(\tilde{w}_j^{[m]}(k) - \tilde{w}_j^{[m]}(k-1)\Big)}{r_j^{[m]}(k+1)}, \\ r_j^{[m]}(k+1) = & r_j^{[m]}(k) + \left\|\tilde{J}_j^{[m]}(k+1)\right\|^2 - \left\|\tilde{J}_j^{[m]}(k-s)\right\|^2, \end{cases} \tag{2.13}$$

де s – розмір «ковзного» вікна.

Цікаво, що при $s = 1$ та $\eta = 1$ отримуємо нелінійну версію загальновідомого алгоритму Качмажа-Уідроу-Хоффа [36, 68]:

$$\tilde{w}_j^{[m]}(k+1) = \tilde{w}_j^{[m]}(k) + \frac{e_j^{[m]}(k+1)\tilde{J}_j^{[m]}(k+1)}{\left\|\tilde{J}_j^{[m]}(k+1)\right\|^2}, \tag{2.14}$$

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

2.3. Навчання нео-фаззі нейронів у оптимізованій каскадній нейронній мережі

Низька швидкість навчання персептронів Розенблатта у поєднанні з труднощами інтерпретації результатів (властиві всім ШНС в цілому) спонукає шукати альтернативні підходи до синтезу еволюційних нейронних мереж. Як зазначається у [35], нейро-фаззі системи відомі високою інтерпретовистю і прозорістю, а також високими апроксимаційними властивостями, та є основою гібридних систем штучного інтелекту. У [15, 41] розглядаються гібридні каскадні системи штучного інтелекту побудовані на нео-фаззі нейронах [48, 73], що дозволяє їм суттєво підвищити швидкість корегування синаптичних вагових коефіцієнтів. Нео-фаззі нейрон (NFN), що його архітектуру наведено на рис. 2.3, – це нелінійна система, що реалізує нечітке висновування

$$\hat{y} = \sum_{i=1}^{n} f_i(x_i), \tag{2.15}$$

де x_i – i-й вхідний сигнал $(i = 1, 2, \ldots, n)$,

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

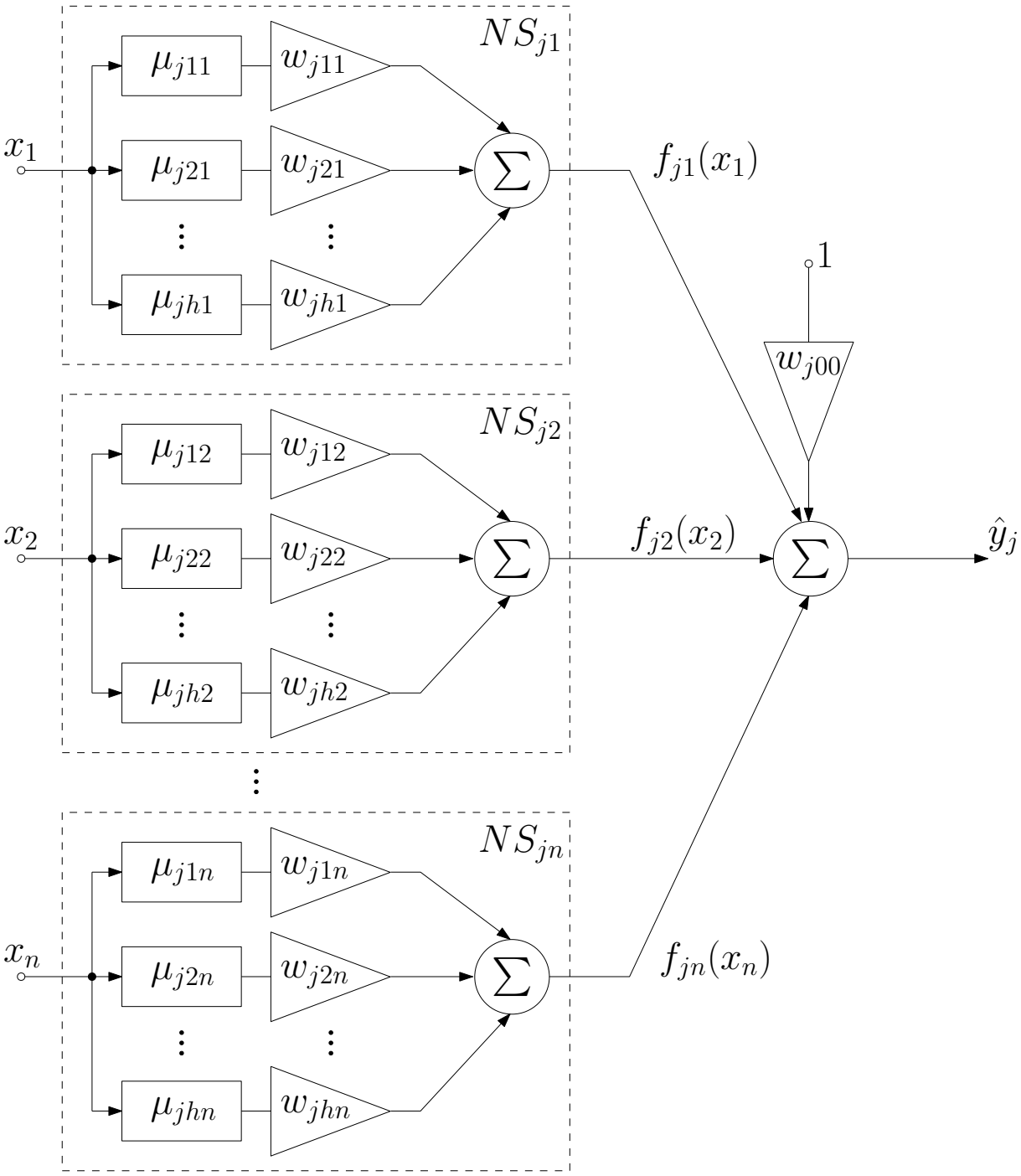

Рис. 2.3. Архітектура нео-фаззі нейрону

Структурними елементами нео-фаззі нейрона є нелінійні синапси NS_i, які

трансформують вхідні сигнали в наступний спосіб:

$$f_i(x_i) = \sum_{l=1}^{h} w_{li}\mu_{li}(x_i), \tag{2.16}$$

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

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

Таким чином, нелінійний синапс NS_i реалізує нечітке висновування

$$\text{IF } x_i \text{ IS } X_{li} \text{ THEN THE OUTPUT IS } w_{li}, \tag{2.17}$$

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 [35].

Запишемо вихідні сигнали для нейронів першого каскаду у наступному вигляді:

$$\begin{cases} \hat{y}_j^{[1]}(k) = \sum\limits_{i=1}^{n} f_{ji}^{[1]}(x(k)) = \sum\limits_{i=1}^{n}\sum\limits_{l=1}^{h} w_{jli}^{[1]}\mu_{jli}^{[1]}(x_i(k)), \\ \text{IF } x_i \text{ IS } X_{li} \text{ THEN THE OUTPUT IS } w_{li} \end{cases} \tag{2.18}$$

j-й нео-фаззі нейрон другого каскаду зображено на рис. 2.4 згідно топології нейронної мережі, зображеної на рис. 2.1).

Автори нео-фаззі нейрона [48, 73] в якості фунцій належності використовували традиційні трикутні структури, які задовільняють умові розбиття Руспіні:

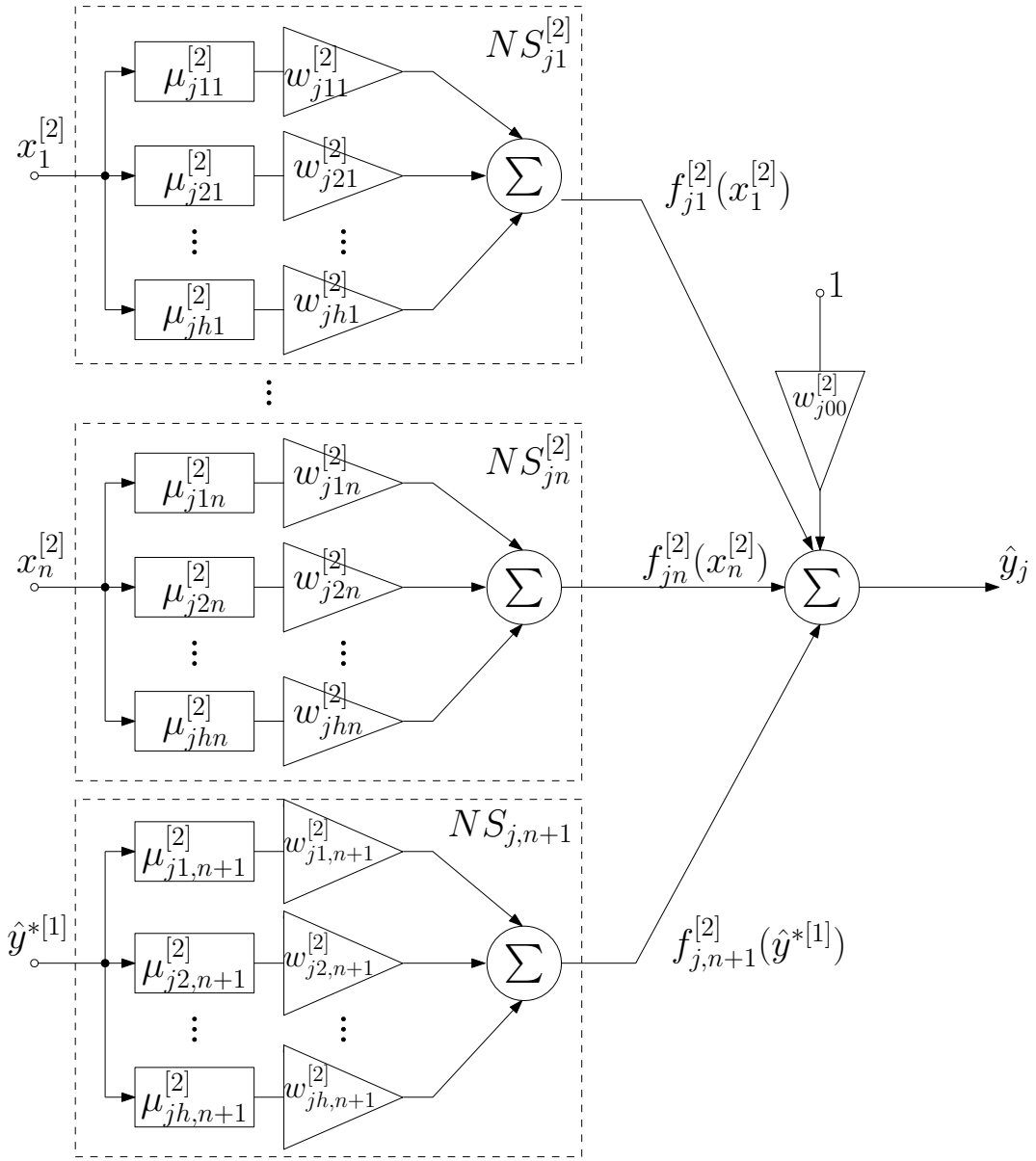

Рис. 2.4. Нео-фаззі нейрон другого каскаду пропонованої каскадної системи

$$\mu_{jli}^{[1]}(x_i) = \begin{cases} \dfrac{x_i - c_{j,l-1,i}^{[1]}}{c_{jli}^{[1]} - c_{j,l-1,i}^{[1]}} \text{ if } x_i \in \left[c_{j,l-1,i}^{[1]}, c_{jli}^{[1]}\right], \\ \dfrac{c_{j,l+1,i}^{[1]} - x_i}{c_{j,l+1,i}^{[1]} - c_{jli}^{[1]}} \text{ if } x_i \in \left[c_{jli}^{[1]}, c_{j,l+1,i}^{[1]}\right], \\ 0 \text{ інакше}, \end{cases} \tag{2.19}$$

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:

$$\mu_{jli}^{[1]}(x_i) + \mu_{j,l+1,i}^{[1]}(x_i) = 1, \tag{2.20}$$

$$f_{jl}^{[1]}(x_i) = w_{jli}^{[1]}\mu_{jli}^{[1]}(x_i) + w_{j,l+1,i}^{[1]}\mu_{j,l+1,i}^{[1]}(x_i). \tag{2.21}$$

Аппроксимуючі властивості системи можна поліпшити використовуючи кубічні сплайни [15] замість трикутних функцій належності:

$$\mu_{jli}^{[1]}(x_i) = \begin{cases} \dfrac{1}{4}\left(2 + 3\dfrac{2x_i - c_{jli}^{[1]} - c_{j,l-1,i}^{[1]}}{c_{jli1}^{[1]} - c_{j,l-1,i}^{[1]}} - \left(\dfrac{2x_i - c_{jli}^{[1]} - c_{j,l-1,i}^{[1]}}{c_{jli}^{[1]} - c_{j,l-1,i}^{[1]}}\right)^3\right), \\ \text{if } x \in \left[c_{j,l-1,i}^{[1]}, c_{jli}^{[1]}\right], \\ \dfrac{1}{4}\left(2 - 3\dfrac{2x_i - c_{j,l+1,i}^{[1]} - c_{jli}^{[1]}}{c_{j,l+1,i}^{[1]} - c_{jli}^{[1]}} + \left(\dfrac{2x_i - c_{j,l+1,i}^{[1]} - c_{jli}^{[1]}}{c_{j,l+1,i}^{[1]} - c_{jli}^{[1]}}\right)^3\right), \\ \text{if } x \in \left[c_{jli}^{[1]}, c_{j,l+1,i}^{[1]}\right], \\ 0, \text{ інакше}, \end{cases} \tag{2.22}$$

або B-сплайни [41]:

$$\mu_{jli}^{g[1]} = \begin{cases} \left.\begin{array}{l} 1, \text{ if } x_i \in \left[c_{jli}^{[1]}, c_{j,l+1,i}^{[1]}\right], \\ 0, \text{ otherwise} \end{array}\right\} \text{для } g = 1, \\ \dfrac{x_i - c_{jli}^{[1]}}{c_{j,l+g-1,i}^{[1]} - c_{jli}^{[1]}}\mu_{jli}^{g-1,[1]}(x_i) + \dfrac{c_{j,l+g,i}^{[1]} - x_i}{c_{j,l+g,i}^{[1]} - c_{j,l+g,i}^{[1]}}\mu_{j,l+1,i}^{g-1,[1]}(x_i), \\ \text{для } g > 1, \end{cases} \tag{2.23}$$

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

Нескладно помітити, що при $g = 2$ отримуємо трикутні функції належності (2.19).

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 (2.18) визначаємо вихідні сигнали інших каскадів. Так, для другого каскаду можемо записати вихідні сигнали у формі

$$\hat{y}_j^{[2]} = \sum_{i=1}^{n}\sum_{l=1}^{h} w_{jli}^{[2]}\mu_{jli}^{[2]}(x_i) + \sum_{l=1}^{h} w_{j,l,n+1}^{[2]}\mu_{j,l,n+1}^{[2]}\left(\hat{y}^{*[1]}\right), \tag{2.24}$$

вихідні сигнали для нейронів m-ого каскаду

$$\hat{y}_j^{[m]} = \sum_{i=1}^{n}\sum_{l=1}^{h} w_{jli}^{[m]}\mu_{jli}^{[m]}(x_i) + \sum_{p=n+1}^{n+m-1}\sum_{l=1}^{h} w_{jlp}^{[m]}\mu_{jlp}^{[m]}\left(\hat{y}^{*[p-n]}\right). \tag{2.25}$$

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

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

$$\begin{aligned}\mu_j^{[m]}(k) = \Big(&\mu_{j11}^{[m]}(x_1(k)),\ldots,\mu_{jh1}^{[m]}(x_1(k)),\mu_{j12}^{[m]}(x_2(k)),\\ &\ldots,\mu_{jh2}^{[m]}(x_2(k)),\ldots,\mu_{jli}^{[m]}(x_i(k)),\ldots,\mu_{jhn}^{[m]}(x_n(k)),\\ &\ldots,\mu_{j1,n+1}^{[m]}\left(\hat{y}^{*[1]}(k)\right),\ldots,\mu_{jh,n+m-1}^{[m]}\left(\hat{y}^{*[m-1]}(k)\right)\Big)^T\end{aligned} \tag{2.26}$$

та відповідний вектор синаптичних вагових коефіцієнтів

$$w_j^{[m]} = \Big(w_{j11}^{[m]}, \ldots, w_{jh1}^{[m]}, w_{j12}^{[m]}, \ldots, w_{jh2}^{[m]}, \ldots, w_{jli}^{[m]}, \ldots, w_{jhn}^{[m]}, w_{j1,n+1}^{[m]}, \ldots, w_{jh,n+m-1}^{[m]}, \Big)^T. \tag{2.27}$$

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

$$\hat{y}_j^{[m]}(k) = w_j^{[m]T} \mu_j^{[m]}(k). \tag{2.28}$$

У такому разі критерій навчання (2.6) приймає вигляд

$$E_j^{[m]}(k) = \frac{1}{2}\big(e_j^m(k)\big)^2 = \frac{1}{2}\Big(y(k) - w_j^{[m]T} \mu_j^{[m]}(k)\Big), \tag{2.29}$$

а мінімізувати його можна використавши модифікацію процедури [18] для «плаваючого» вікна

$$\begin{cases} w_j^{[m]}(k+1) = w_j^{[m]}(k) + \dfrac{e_j^{[m]}(k+1)\mu_j^{[m]}(k+1)}{r_j^{[m]}(k+1)}, \\ r_j^{[m]}(k+1) = r_j^{[m]}(k) + \left\|\mu_j^{[m]}(k+1)\right\|^2 - \left\|\mu_j^{[m]}(k-s)\right\|^2, + \end{cases} \tag{2.30}$$

або для випадку, коли $s = 1$,

$$w_j^{[m]}(k+1) = w_j^{[m]}(k) + \frac{e_j^{[m]}(k+1)\mu_j^{[m]}(k+1)}{\left\|\mu_j^{[m]}(k+1)\right\|^2}, \tag{2.31}$$

що збігається з одношаговим оптимальним алгоритмом Качмажа-Уідроу-Хоффа.

Вочевидь, замість (2.30) можна скористатися іншими алгоритмами, як-от експоненційно зважений рекурентний метод найменших квадратів (EWRLSM), що використовується у DENFIS [38], ETS [3] та FLEXFIS [4, 46]. Та варто зауважити, що EWRLSM може бути нестійким при малому коефіцієнті забування.

При використанні критерія навчання з регуляризуючим параметром (momentum term) (2.6) замість (2.29) отримуємо остаточний метод навчання нео-фаззі нейрона

$$\begin{cases} w_j^{[m]}(k+1) = & w_j^{[m]}(k) + \dfrac{\eta e_j^{[m]}(k+1)\mu_j^{[m]}(k+1)}{r_j^{[m]}(k+1)} \\ & + \dfrac{(1-\eta)\Big(w_j^{[m]}(k) - w_j^{[m]}(k-1)\Big)}{r_j^{[m]}(k+1)}, \\ r_j^{[m]}(k+1) = & r_j^{[m]}(k) + \left\|\mu_j^{[m]}(k+1)|\right\|^2 - \left\|\mu_j^{[m]}(k-s)\right\|^2. \end{cases} \tag{2.32}$$

Варто зробити наголос, що оскільки вихідні сигнали нео-фаззі нейрона лінійно залежать від його синаптичних вагових коефіцієнтів, можна використовувати будь-які методи адаптивної лінійної ідентифікації [45] (наприклад, рекурентний метод найменших квадратів, робастні методи, методи, що ігнорують застарілі данні, тощо), що дозволяє обробляти нестаціонарні сигнали в онлайн режимі.

2.4. Розширенні 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

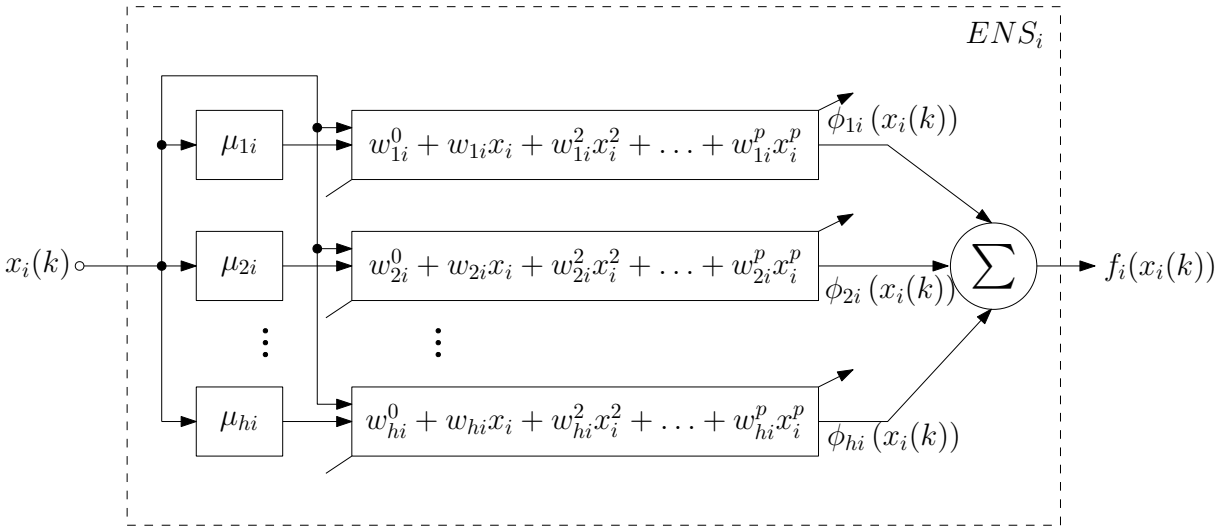

Рис. 2.5. Синапс розширеного нео-фаззі нейрону

Такаґі-Суґено нульового порядку [65, 66]. 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. 2.5 та рис. 2.6 відповідно.

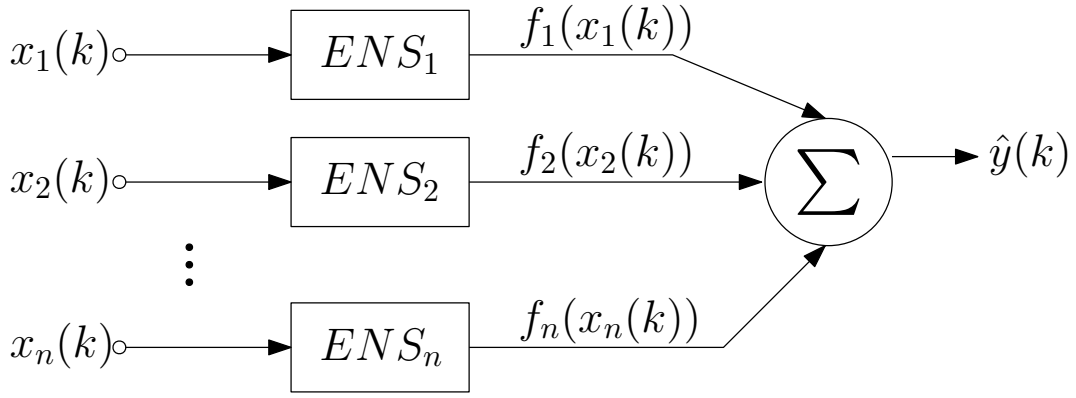

Рис. 2.6. Розширений нео-фаззі нейрон

Вводячі нові змінні

$$\phi_{li}(x_i) = \mu_{li}(x_i)\big(w_{li}^0 + w_{li}^1x_i + w_{li}^2x_i^2 + \cdots + w_{li}^px_i^p\big), \tag{2.33}$$

$$
\begin{aligned}
f_i(x_i) &= \sum_{l=1}^{h} \mu_{li}(x_i)\left(w_{li}^0 + w_{li}^1 x_i + w_{li}^2 x_i^2 + \cdots + w_{li}^p x_i^p\right) \\
&= w_{li}^0 \mu_{li}(x_i) + w_{li}^1 x_i \mu_{1i}(x_i) + \cdots + w_{li}^p x_i^p \mu_{1i}(x_i) \\
&+ w_{2i}^0 \mu_{2i}(x_i) + \cdots + w_{2i}^p x_i^p \mu_{2i}(x_i) + \cdots + w_{hi}^p x_i^p \mu_{hi}(x_i),
\end{aligned} \tag{2.34}
$$

$$
w_i = \left(w_{1i}^0, w_{1i}^1, \ldots, w_{1i}^p, w_{2i}^0, \ldots, w_{2i}^p, \ldots, w_{hi}^p\right)^T, \tag{2.35}
$$

$$
\begin{aligned}
\tilde{\mu}_i(x_i) = \Bigg(& \mu_{1i}(x_i), x_i(\mu_{1i}(x_i), \ldots, x_i^p(\mu_{1i}(x_i), \\
& \mu_{2i}(x_i), \ldots, x_i^p \mu_{2i}(x_i), \ldots, x_i^p \mu_{hi}(x_i)\Bigg)^T,
\end{aligned} \tag{2.36}
$$

можна представити вихідні сигнали розширеного нео-фаззі нейрона у вигляді

$$
f_i(x_i) = w_i^T \tilde{\mu}_i(x_i), \tag{2.37}
$$

$$
\begin{aligned}
\hat{y} &= \sum_{i=1}^{n} f_i(x_i) \\
&= \sum_{i=1}^{n} w_i^T \tilde{\mu}(x_i) \\
&= \tilde{w}^T \tilde{\mu}(x).
\end{aligned} \tag{2.38}
$$

де

$$
\tilde{w}^T = \left(w_1^T, \ldots, w_i^T, \ldots, w_n^T\right)^T, \tag{2.39}
$$

$$
\tilde{\mu}(x) = \left(\tilde{\mu}_1^T(x_1), \ldots, \tilde{\mu}_i^T(x_i), \ldots, \tilde{\mu}_n^T(x_n)\right)^T, \tag{2.40}
$$

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

$$\begin{gathered}\text{IF } x_i \text{ IS } X_{li} \text{ THEN THE OUTPUT IS} \\ w_{li}^0 + w_{li}^1 x_i + \cdots + w_{li}^p x_p,\ l = 1, 2, \ldots, h,\end{gathered} \tag{2.41}$$

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

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

$$\hat{y}^{[1]}(k) = \tilde{w}^{[1]T}(k-1)\tilde{\mu}^{[1]}(x(k)), \tag{2.42}$$

що відрізняється від виразу (2.28) 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 (2.30) та (2.31) для j-ого нейрону m-ого каскаду приймають вигляд

$$\begin{cases}\tilde{w}_j^{[m]}(k+1) = \tilde{w}_j^{[m]}(k) + \dfrac{e_j^{[m]}(k+1)\tilde{\mu}_j^{[m]}(k+1)}{\tilde{r}_j^{[m]}(k+1)}, \\ \tilde{r}_j^{[m]}(k+1) = \tilde{r}_j^{[m]}(k) + \left\|\tilde{\mu}_j^{[m]}(k+1)\right\|^2 - \left\|\tilde{\mu}_j^{[m]}(k-s)\right\|^2\end{cases} \tag{2.43}$$

та

$$\tilde{w}_j^{[m]}(k+1) = \tilde{w}_j^{[m]}(k) + \frac{e_j^{[m]}(k+1)\tilde{\mu}_j^{[m]}(k+1)}{\left\|\tilde{\mu}_j^{[m]}(k+1)\right\|^2} \tag{2.44}$$

відповідно.

2.5. Оптимізація пулу нео-фаззі нейронів

Вихідні сигнали, згенеровані нейронами пулу кожного з каскадів, можна об'єднати у окремому вузлі-нейроні $GN^{[m]}$, з точністю $\hat{y}^{*[m]}(k)$, не меншою

від точності будь-якого нейрону пулу $\hat{y}_j^{[m]}(k)$. Це завдання можна вирішити за допомогою підходу ансамблей нейронних мереж. Хоча відомі алгоритми не призначені для роботи в онлайн-режимі, варто розглянути методи адаптивного узагальнюючого прогнозування [51, 62].

Введемо вектор вхідних сигналів для m-ого каскаду:

$$\hat{y}^{[m]}(k) = \left(\hat{y}_1^{[m]}(k), \hat{y}_2^{[m]}(k), \ldots, \hat{y}_q^{[m]}(k)\right)^T; \tag{2.45}$$

тоді отпимальний вихідний сигнал, що його генерує нейрон $GN^{[m]}$ (що, власне, є адаптивним лінійним асоціатором [23, 33]), можна записати у формі

$$\hat{y}^{*[m]}(k) = \sum_{j=1}^{1} c_j^{[m]} \hat{y}_j^{[m]}(k) = c^{[m]T} \hat{y}^{[m]}(k) \tag{2.46}$$

з обмеженнями на незміщенність

$$\sum_{j=1}^{q} c_j^{[m]} = E^T c^{[m]} = 1, \tag{2.47}$$

де $c^{[m]} = \left(c_1^{[m]}, c_2^{[m]}, \ldots, c_q^{[m]}\right)^T$ та $E = (1, 1, \ldots, 1)^T$ – $(q \times 1)$-вектори.

Введемо критерій навчання на «ковзному» вікні

$$\begin{aligned} E^{[m]}(k) =& \frac{1}{2} \sum_{\tau=k-s+1}^{k} \left(y(\tau) - \hat{y}^{*[m]}(\tau)\right)^2 \\ =& \frac{1}{2} \sum_{\tau=k-s+1}^{k} \left(y(\tau) - c^{[m]T} \hat{y}^{[m]}(\tau)\right)^2, \end{aligned} \tag{2.48}$$

зважаючи на обмеженяя (2.47), функція Лаґранжа матиме вигляд

$$L^{[m]}(k) = E^{[m]}(k) - \lambda\left(1 - E^T c^{[m]}\right), \tag{2.49}$$

де λ – невизначений Лаґранжів множник.

Мінімізуючи (2.49) відносно $c^{[m]}$, отримуємо

$$\begin{cases} \hat{y}^{*[m]}(k+1) = \dfrac{\hat{y}^{[m]T}(k+1)P^{[m]}(k+1)E}{E^T P^{[m]}(k+1)E}, \\ P^{[m]}(k+1) = \left(\sum\limits_{\tau=k-s+2}^{k+1} \hat{y}^{[m]}(\tau)\hat{y}^{[m]T}(\tau) \right)^{-1} \end{cases} \tag{2.50}$$

або у рекурентній формі

$$\begin{cases} \tilde{P}^{[m]}(k+1) = P^{[m]}(k) - \dfrac{P^{[m]}(k)\hat{y}^{[m]}(k+1)\hat{y}^{[m]T}(k+1)P^{[m]}(k)}{1+\hat{y}^{[m]T}(k+1)P^{[m]}(k)\hat{y}^{[m]}(k+1)}, \\ P^{[m]}(k+1) = \tilde{P}^{[m]}(k+1) + \\ \qquad + \dfrac{\tilde{P}^{[m]}(k+1)\hat{y}(k-s+1)\hat{y}^{[m]T}(k-s+1)\tilde{P}^{[m]}(k+1)}{1-\hat{y}^{[m]T}(k-s+1)\tilde{P}^{[m]}(k+1)\hat{y}^{[m]}(k-s+1)}, \\ \hat{y}^{*[m]}(k+1) = \dfrac{\hat{y}^{[m]T}(k+1)P^{[m]}(k+1)E}{E^T P^{[m]}(k+1)E}. \end{cases} \tag{2.51}$$

У випадку, коли $s = 1$ (2.50) та (2.51) приймають доволі простий вигляд:

$$\begin{aligned} \hat{y}^{*[m]}(k+1) &= \frac{\hat{y}^{[m]T}(k+1)\hat{y}^{[m]}(k+1)}{E^T\hat{y}^{[m]}(k+1)} = \\ &= \frac{\left\|\hat{y}^{[m]}(k+1)\right\|^2}{E^T\hat{y}^{[m]}(k+1)} = \\ &= \frac{\sum\limits_{j=1}^{q}\left(\hat{y}^{[m]}(k+1)\right)^2}{\sum\limits_{j=1}^{q}\hat{y}^{[m]}(k+1)}. \end{aligned} \tag{2.52}$$

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.

Висновки до розділу 2

1. Розглянуті існуючі гібрідні системи обчислювального інтелекту, що еволюціонують, та визначені потенційні модифікації, що їх варто привнести аби такі системи можна було застосувати у режимі послідовного надходження даних на обробку.
2. Зсинтезована варіація каскадної системи, що еволюціонує, побудована на персептронах Розенблатта, для послідовного обробляння вхідних сигналів, що дозволило сформувати вимоги до вузлів шуканої гібридної системи.
3. Запропонована архітектура та методи навчання гібридної каскадної системи, що еволюціонує, заснованої на нео-фаззі нейронах. Пропонованій системі притаманні усі переваги нео-фаззі нейронів (інтерпритуємість та прозорість одночасно з вискокими апроксимаційними властивостями), а також, зрештою, вона забезпечує модель адекватної складності для кожного поставленого завдання.
4. Запропонована архітектура та методи навчання гібридної каскадної нейронної мережі, що еволюціонує, з оптимізацією пулу нейронів у кожному каскаді, що реалізують оптимальний за точністю прогноз нелінійних стохастичних і хаотичних сигналів у онлайн режимі. Варто зазначити, що оптимізіція пулу нейронів дуже доречна саме у разі застосування системи для аналізу даних в онлайн режимі, адже використання узагальнюючих нейронів дозволяє визначати оптимальний нейрон на кожному етапі функціонування системи, який з високою вірогідністю може змінюватися у випадку послідовного обробляння сигналів нестаціонарних об'єктів.
5. Запропонований розширений нео-фаззі нейрон, який дозволяє реалізовувати нечітке висновування за Такаґі-Суґено довільного порядку, що має покращені апроксимуючі властивості. Зсинтезована архітекутра гібридної системи, що ґрунтується на розширених нео-фаззі нейро-

нах.

РОЗДІЛ 3
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 [23, 26, 33, 37], 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 помилок (багатошарові персептрони, рекурентні нейронні мережі, адаптивні нейромережеві системи нечіткого виведення – ANFIS [69]) або методі найменших квадратів (радіально-базисні та функціонально пов'язані нейронні мережі) та скористатися процедурами на основі локальних критеріїв та «короткої» пам'яті типу алгоритма Качмажа-Уідроу-Хоффа. 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 спеціалі-

зованих 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 $R^n \to R^g$ у режимі реального часу.

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

Для вирішеня 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 [11, 12, 15–17, 27, 34, 41, 74]. Однак, слід зазначити, що каскадно-кореляційна мережа у формі, що її запропонували С. Фальман і К. Лєб'єр [27], 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 модифікаціїї до архітектури каскадно-кореляційної мережі CasCorLA:

1. замість елементарних персептронів Розенблата використовувати нео-фаззі нейрони (доцільність такого рішення було детально показано у розділі 2),
2. кількість нейронів у кожному каскаді відтепер має дорівнювати розмірності вектору вихідного сигналу системи.

Схему пропонованої архітектури наведено на рис. 3.1.

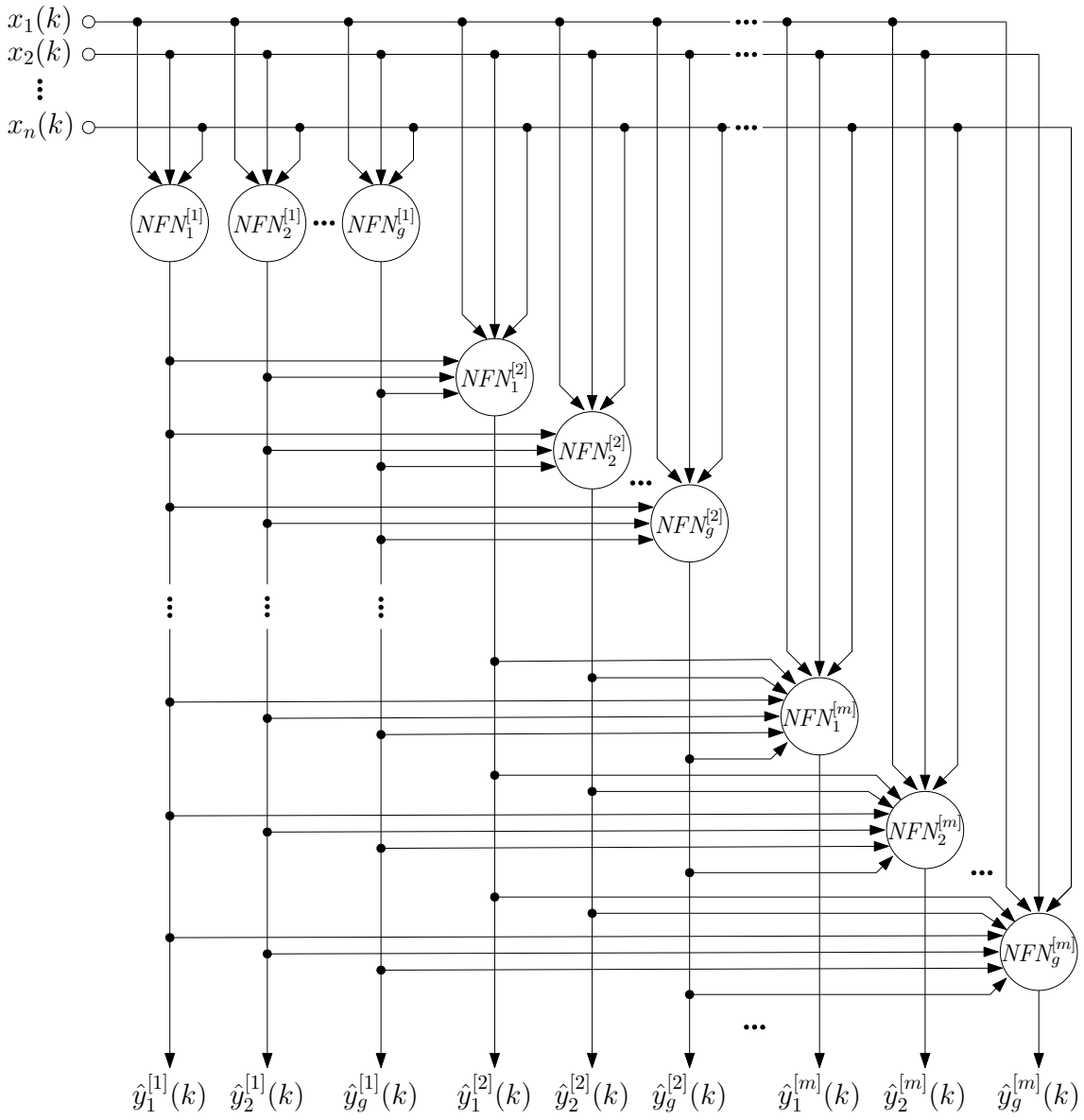

Рис. 3.1. Архітектура гібридної MIMO системи, побудованої на нео-фаззі нейронах

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

$$\hat{y}(k) = \left(\hat{y}_1^{*[m]}(k), \hat{y}_2^{*[m]}(k), \ldots, \hat{y}_g^{*[m]}(k)\right)^T, \tag{3.1}$$

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

Для кожного з нео-фаззі нейронів системи в якості функцій належності мо-

жна використовувати трикутні конструкції:

$$\mu_{jli}^{[1]}(x_i) = \begin{cases} \dfrac{x_i - c_{d,l-1,i}^{[1]j}}{c_{dli}^{[1]j} - c_{d,l-1,i}^{[1]j}} \text{ якщо } x_i \in \left[c_{d,l-1,i}^{[1]j}, c_{dli}^{[1]j}\right], \\ \dfrac{c_{d,l+1,i}^{[1]j} - x_i}{c_{d,l+1,i}^{[1]j} - c_{dli}^{[1]j}} \text{ якщо } x_i \in \left[c_{dli}^{[1]j}, c_{d,l+1,i}^{[1]j}\right], \\ 0 \text{ у протилежному випадку}, \end{cases} \tag{3.2}$$

кубічні сплайни:

$$\mu_{jli}^{[1]}(x_i) = \begin{cases} \dfrac{1}{4}\left(2 + 3\dfrac{2x_i - c_{dli}^{[1]j} - c_{d,l-1,i}^{[1]j}}{c_{dli1}^{[1]j} - c_{d,l-1,i}^{[1]j}} - \left(\dfrac{2x_i - c_{dli}^{[1]j} - c_{d,l-1,i}^{[1]j}}{c_{dli}^{[1]j} - c_{d,l-1,i}^{[1]j}}\right)^3\right), \\ \text{якщо } x \in \left[c_{d,l-1,i}^{[1]j}, c_{dli}^{[1]j}\right], \\ \dfrac{1}{4}\left(2 - 3\dfrac{2x_i - c_{d,l+1,i}^{[1]j} - c_{dli}^{[1]j}}{c_{d,l+1,i}^{[1]j} - c_{dli}^{[1]j}} + \left(\dfrac{2x_i - c_{d,l+1,i}^{[1]j} - c_{dli}^{[1]j}}{c_{d,l+1,i}^{[1]j} - c_{dli}^{[1]j}}\right)^3\right), \\ \text{якщо } x \in \left[c_{dli}^{[1]j}, c_{d,l+1,i}^{[1]j}\right], \\ 0 \text{ у протилежному випадку}, \end{cases} \tag{3.3}$$

або B-сплайни:

$$\mu_{jli}^{g[1]} = \begin{cases} \left.\begin{array}{l} 1 \text{ якщо } x_i \in \left[c_{dli}^{[1]j}, c_{d,l+1,i}^{[1]j}\right], \\ 0 \text{ у протилежному випадку} \end{array}\right\} \text{ якщо } g = 1, \\ \dfrac{x_i - c_{dli}^{[1]j}}{c_{d,l+g-1,i}^{[1]j} - c_{dli}^{[1]j}}\mu_{dli}^{g-1,[1]j}(x_i) + \dfrac{c_{d,l+g,i}^{[1]j} - x_i}{c_{d,l+g,i}^{[1]j} - c_{d,l+g,i}^{[1]j}}\mu_{d,l+1,i}^{g-1,[1]j}(x_i), \\ \text{якщо } g > 1, \end{cases} \tag{3.4}$$

де $\mu_{dli}^{g[1]j}(x_i)$ – l-й сплайн g-ого порядку. Варто зауважити, що всі ці конструкції задовільняють умовам одиничного розбиття Руспіні.

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

$$\begin{cases} \hat{y}_d^{[1]j}(k) = \sum\limits_{i=1}^{n} f_{di}^{[1]j}(x_i(k)) = \sum\limits_{i=1}^{n}\sum\limits_{l=1}^{h} w_{dli}^{[1]j}\mu_{dli}^{[1]j}(x_i(k)), \\ \text{ЯКЩО } x_i(k) \in X_{li}^{j} \text{ , ТОДІ ВИХІД } w_{dli}^{[1]j}. \end{cases} \tag{3.5}$$

вихідні сигнали нео-фаззі нейронів другого каскаду:

$$\begin{aligned} \hat{y}_d^{[2]j} = &\sum_{i=1}^{n}\sum_{l=1}^{h} w_{dli}^{[2]j}\mu_{dli}^{[2]j}(x_i) + \\ &\sum_{d=1}^{g}\sum_{l=1}^{h} w_{dl,n+1}^{[2]j}\mu_{dl,n+1}^{[2]j}\left(\hat{y}_d^{*[1]}\right) \ \forall \ d = 1, 2, \ldots, g \end{aligned} \tag{3.6}$$

вихідні сигнали m-ого каскаду:

$$\begin{aligned} \hat{y}_d^{[2]j} = &\sum_{i=1}^{n}\sum_{l=1}^{h} w_{dli}^{[2]j}\mu_{dli}^{[2]j}(x_i) + \\ &\sum_{d=1}^{g}\sum_{p=n+1}^{n+m-1}\sum_{l=1}^{h} w_{dlp}^{[m]j}\mu_{dlp}^{[m]j}\left(\hat{y}_d^{*[p-n]}\right) \ \forall \ d = 1, 2, \ldots, g \end{aligned} \tag{3.7}$$

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

$$\begin{aligned} \mu_d^{[m]j}(k) = \Big(&\mu_{d11}^{[m]j}(x_1(k)), \ldots, \mu_{dh1}^{[m]j}(x_1(k)), \mu_{d12}^{[m]j}(x_2(k)), \\ &\ldots, \mu_{dh2}^{[m]j}(x_2(k)), \ldots, \mu_{dli}^{[m]j}(x_i(k)), \ldots, \mu_{dhn}^{[m]j}(x_n(k)), \\ &\ldots, \mu_{d1,n+1}^{[m]j}\left(\hat{y}^{*[1]}(k)\right), \ldots, \mu_{dh,n+m-1}^{[m]j}\left(\hat{y}^{*[m-1]}(k)\right)\Big)^T \end{aligned} \tag{3.8}$$

та відповідний йому вектор синаптичних вагових коефіцієнтів

$$\begin{aligned} w_d^{[m]j} = \Big(&w_{d11}^{[m]j}, \ldots, w_{dh1}^{[m]j}, w_{d12}^{[m]j}, \ldots, w_{dh2}^{[m]j}, \ldots, w_{dli}^{[m]j}, \\ &\ldots, w_{dhn}^{[m]j}, w_{d1,n+1}^{[m]j}, \ldots, w_{dh,n+m-1}^{[m]j}\Big)^T, \end{aligned} \tag{3.9}$$

щоб записати вихідний сигнал системи у компактній формі:

$$\hat{y}_d^{[m]j}(k) = \left(w_d^{[m]j}\right)^T \mu_d^{[m]j}(k). \tag{3.10}$$

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 у першому розділі. Так корегувати вагові кофіцієнти можна за допомогою експоненційно зваженого рекурентного методу найменших квадратів:

$$\begin{cases} w_d^{[m]j}(k+1) = w_d^{[m]j}(k) + \\ \dfrac{P_d^{[m]j}(k)\left(y^d(k+1) - \left(w_d^{[m]j(k)}\right)^T \mu_d^{[m]j}(k+1)\right)}{\alpha + \left(\mu_d^{[m]j}(k+1)\right)^T P_d^{[m]j}(k)\mu_d^{[m]j}(k+1)}\mu_d^{[m]j}(k+1), \\ P_d^{[m]j}(k+1) = \dfrac{1}{\alpha}\left(P_d^{[m]j}(k) - \dfrac{P_d^{[m]j}(k)\mu_d^{[m]j}(k+1)\left(\mu_d^{[m]j}(k+1)\right)^T P_d^{[m]j}(k)}{\alpha + \left(\mu_d^{[m]j}(k+1)\right)^T P_d^{[m]j}(k)\mu_d^{[m]j}(k+1)}\right), \end{cases} \tag{3.11}$$

де $y^d(k+1), d = 1, 2, \ldots, g$ – зовнішній навчальний сигнал,

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

або градієнтного методу навчання, що, як зазначалося, відрізняється як згладжувальними, так і слідкуючими властивостями:

$$\begin{cases} w_d^{[m]j}(k+1) & = w_d^{[m]j}(k) + \dfrac{y^d(k+1) - \left(w_d^{[m]j}(k)\right)^T \mu_d^{[m]j}(k+1)}{r_d^{[m]j}(k+1)}\mu_d^{[m]j}(k+1), \\ r_d^{[m]j}(k+1) & = \alpha r_d^{[m]j} + \left\|\mu_d^{[m]j}(k+1)\right\|^2, 0 \leq \alpha \leq 1. \end{cases} \tag{3.12}$$

3.1.1. 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 пропоноувалося 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 сингали, 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 zg 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. 3.1.

3.1.1.1. 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 у розділі 2.5.

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

$$\hat{y}^{*[m]}(k) = \left(\hat{y}_1^{[m]}(k), \hat{y}_2^{[m]}(k), \ldots, \hat{y}_q^{[m]}(k)\right)^T; \tag{3.13}$$

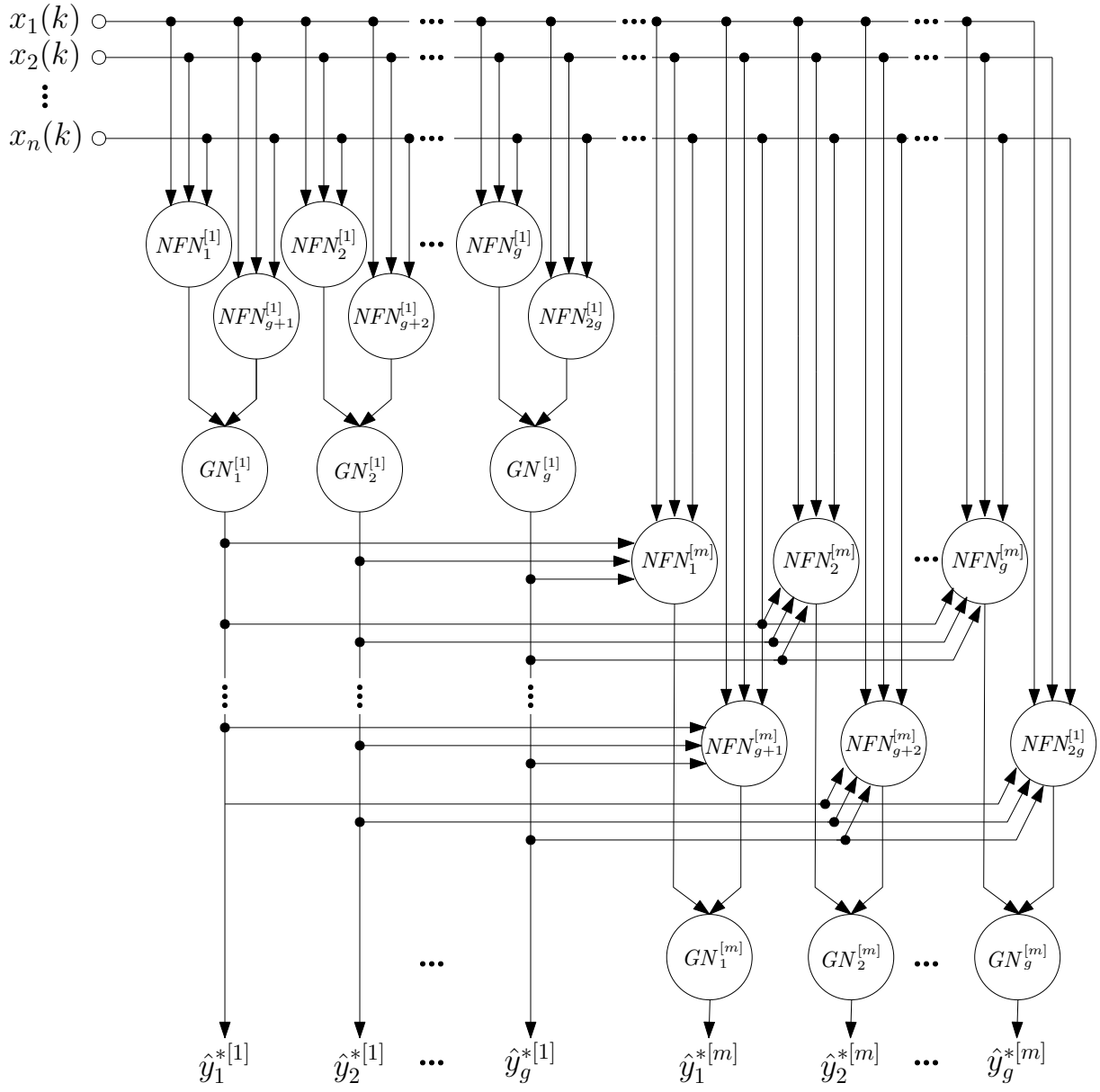

Рис. 3.2. Архітектура гібридної оптимізованої MIMO системи, побудованої на нео-фаззі нейронах

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

$$\left(\hat{y}_1^{[1]}(k), \hat{y}_{g+1}^{[1]}(k), \ldots, \hat{y}_{2g+1}^{[1]}(t), \ldots \hat{y}_{(z-1)(g+1)}^{[1]}(k)\right)^T \tag{3.14}$$

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

$$\left(\hat{y}_2^{[1]}(k), \hat{y}_{g+2}^{[1]}(k), \ldots, \hat{y}_{2g+2}^{[1]}(t), \ldots \hat{y}_{(z-1)(g+2)}^{[1]}(k)\right)^T \tag{3.15}$$

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

$$\left(\hat{y}_g^{[1]}(k), \hat{y}_{2g}^{[1]}(k), \ldots, \hat{y}_{(z-1)g}^{[1]}(k)\right)^T. \tag{3.16}$$

Нагадаємо, що точність вихідного сигналу узагальнюючих елементів має бути не гіршою точності будь-якого сингналу, що узагальнюється (подається на вхід до $GN_d^{[m]}$). Рекурентна форма методу навчання «на ковзному вікні» елементів $GN_d^{[m]}$ кожного каскаду має вигляд

$$\begin{cases} \tilde{P}_d^{[m]}(k+1) = P_d^{[m]}(k) - \dfrac{P_d^{[m]}(k)\hat{y}_d^{[m]}(k+1)\hat{y}_d^{[m]T}(k+1)P_d^{[m]}(k)}{1+\hat{y}_d^{[m]T}(k+1)P_d^{[m]}(k)\hat{y}_d^{[m]}(k+1)}, \\ P_d^{[m]}(k+1) = \tilde{P}_d^{[m]}(k+1) + \\ \qquad \dfrac{\tilde{P}_d^{[m]}(k+1)\hat{y}_d(k-s+1)\hat{y}_d^{[m]T}(k-s+1)\tilde{P}_d^{[m]}(k+1)}{1-\hat{y}_d^{[m]T}(k-s+1)\tilde{P}_d^{[m]}(k+1)\hat{y}_d^{[m]}(k-s+1)}, \\ \hat{y}_d^{*[m]}(k+1) = \dfrac{\hat{y}_d^{[m]T}(k+1)P_d^{[m]}(k+1)E}{E^T P_d^{[m]}(k+1)E}, \end{cases} \tag{3.17}$$

а у випадку, коли $s = 1$:

$$\begin{aligned} \hat{y}_d^{*[m]}(k+1) &= \frac{\hat{y}_d^{[m]T}(k+1)\hat{y}_d^{[m]}(k+1)}{E^T\hat{y}_d^{[m]}(k+1)} \\ &= \frac{\left\|\hat{y}_d^{[m]}(k+1)\right\|^2}{E^T\hat{y}_d^{[m]}(k+1)} \\ &= \frac{\sum\limits_{j=1}^{q}\left(\hat{y}_d^{[m]}(k+1)\right)^2}{\sum\limits_{j=1}^{q}\hat{y}_d^{[m]}(k+1)}. \end{aligned} \tag{3.18}$$

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

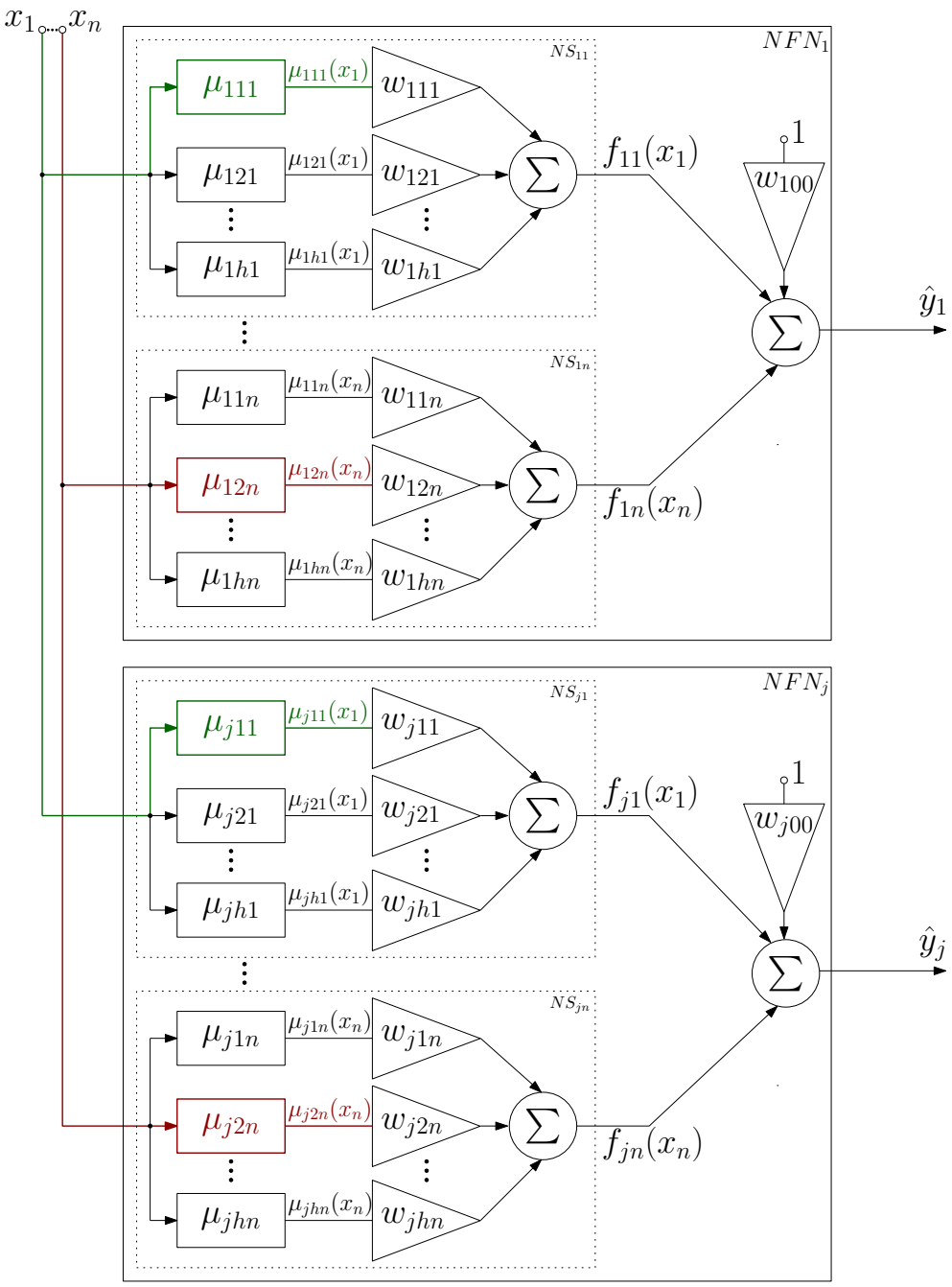

Рис. 3.3. Ілюстрація надмірності MIMO системи, побудованої на нео-фаззі нейронах

Архітектура багатовимірної каскадної системи, яка ґрунтується на звичайних нео-фаззі нейронах, що її описано у підрозділі 3.1, 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 $\hat{y}_d^{[1]j}(k), d = 1, 2, \ldots, g$. У результаті компоненти вихідного вектора

$$\hat{y}^{[1]j}(k) = \left(\hat{y}_1^{[1]j}(k), \hat{y}_2^{[1]j}(k), \ldots, \hat{y}_g^{[1]j}(k)\right)^T \tag{3.19}$$

обчислюються незалежно один від одного, хоча при цьому

$$\mu_{1il}(x_i(k)) = \mu_{2il}(x_i(k)) = \mu_{jil}(x_i(k)) = \mu_{nil}(x_i(k)). \tag{3.20}$$

Надмірність архітектури, що її наведено на рис. 3.1, проілюстрована на рис. 3.3, де зеленим кольором позначені неодноразово обчислювані тотожні значення функцій належності μ_{111} та μ_{j11}, червоним кольором – тотожні μ_{12n} та μ_{j2n}. Уникнути цього можна, якщо ввести до розгляду багатовимірний нео-фаззі нейрон, що є модифікацією систем, запропонованих у [15, 48].

3.2.1. Багатовимірний нео-фаззі нейрон. Вузлами багатовимірного нео-фаззі нейрону MNFN (схема наведена на рис. 3.4) 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

$$\mu^{[1]j}(k) = \left(\mu_{11}^{[1]j}(x_1(k)), \mu_{21}^{[1]j}(x_1(k)), \ldots, \mu_{h1}^{[1]j}(x_1(k)), \ldots, \mu_{hn}^{[1]j}(x_n(k))\right)^T \tag{3.21}$$

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

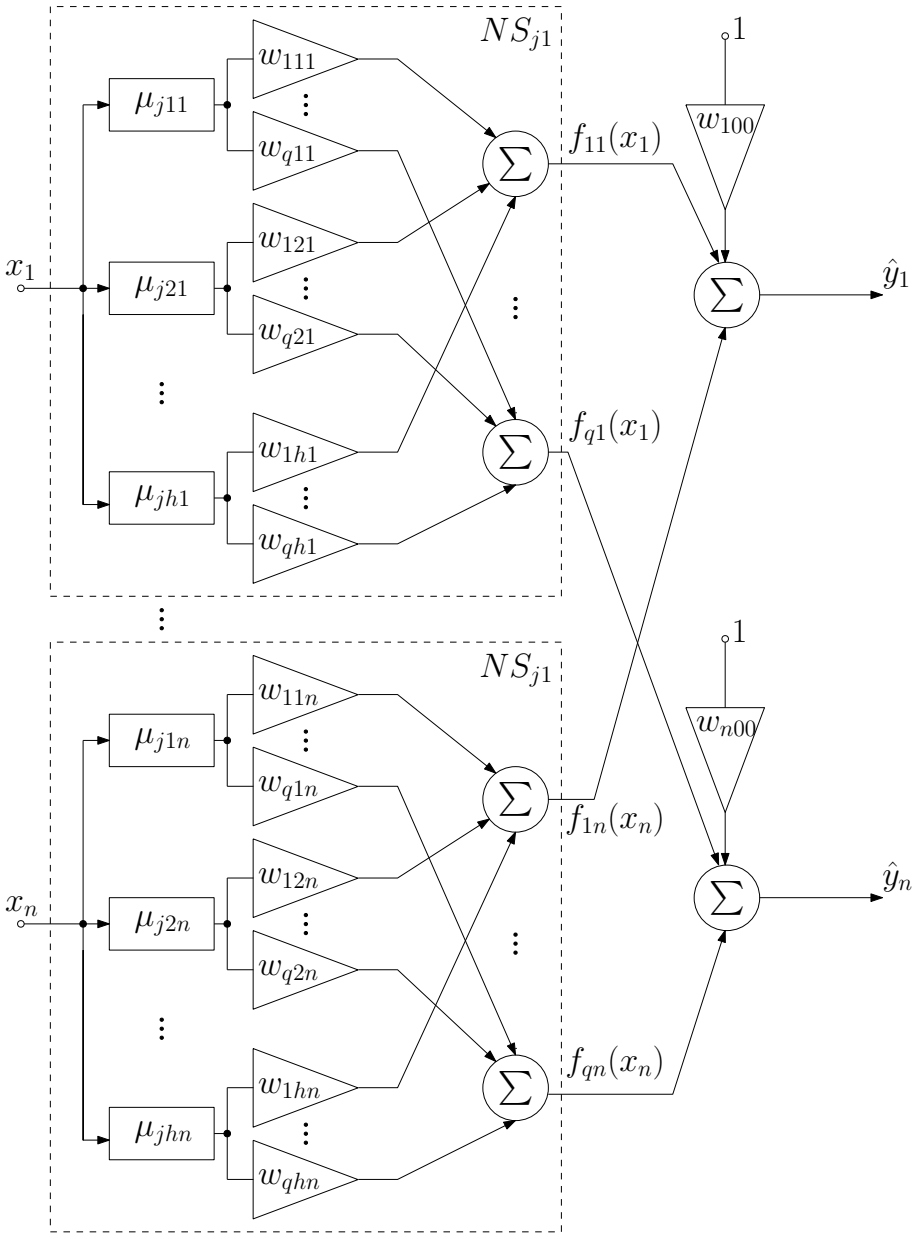

Рис. 3.4. Багатовимірний нео-фаззі нейрон

$$W^{[1]j} = \begin{pmatrix} w_{111}^{[1]j} & w_{112}^{[1]j} & \dots & w_{1li}^{[1]j} & \dots & w_{1hn}^{[1]j} \\ w_{211}^{[1]j} & w_{212}^{[1]j} & \dots & w_{2li}^{[1]j} & \dots & w_{2hn}^{[1]j} \\ \vdots & \vdots & & \vdots & & \vdots \\ w_{g11}^{[1]j} & w_{g12}^{[1]j} & \dots & w_{gli}^{[1]j} & \dots & w_{ghn}^{[1]j} \end{pmatrix} \tag{3.22}$$

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

$$\hat{y}^{[1]j}(k) = W^{[1]j}\mu^{[1]j}(k). \tag{3.23}$$

Навчаняя багатовимірного нео-фаззі нейрону можна реалізувати за допомогою матричної модифікації експоненційно-зваженого рекурентного методу найменших квадратів (3.11) у формі

$$\begin{cases} W^{[1]j}(k+1) & = W^{[1]j}(k)+ \\ & +\dfrac{\big(y(k+1)-W^{[1]j}(k)\mu^{[1]j}(k+1)\big)\big(\mu^{[1]j}(k+1)\big)^T P^{[1]j}(k)}{\alpha+\big(\mu^{[1]j(k+1)}\big)^T P^{[1]j}(k)\mu^{[1]j}(k+1)}, \\ P^{[1]j}(k+1) & = \dfrac{1}{\alpha}\left(P^{[1]j}(k)-\dfrac{P^{[1]j}(k)\mu^{[1]j}(k+1)\big(\mu^{[1]j}(k+1)\big)^T P^{[1]j}(k)}{\alpha+\big(\mu^{[1]j}(k+1)\big)^T P^{[1]j}(k)\mu^{[1]j}(k+1)}\right), \\ & 0<\alpha\le 1 \end{cases} \tag{3.24}$$

або багатовимірного варіанту методу (3.12)

$$\begin{cases} W^{[1]j}(k+1) & = W^{[1]j}(k)+\dfrac{y(k+1)-W^{[1]j}(k)\mu^{[1]j}(k+1)}{r^{[1]j}(k+1)}\times \\ & \times\big(\mu^{[1]j}(k+1)\big)^T, \\ r^{[1]j}(k+1) & = \alpha r^{[1]j}(k)+\big\|\mu^{[1]j}(k+1)\big\|^2, \\ & 0\le\alpha\le 1, \end{cases} \tag{3.25}$$

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

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.

3.2.2. Метод визначення локально оптимального вихідного сигналу пулу багатовимірних нео-фаззі нейронів каскадної системи, що еволюціонує. У цьому підрозділі запропоновано узагальнюючий ней-

рон $GMN^{[m]}$ та рекурентний метод його навчання, щоб він об'єднував усі вихідні сигнали нейронів $MNFN^{[m]}$ пулу каскаду у вихідний сигнал

$$\hat{y}^{*[m]}(k) = \left(\hat{y}_1^{*[m]}(k), \hat{y}_2^{*[m]}(k), \ldots, \hat{y}_g^{*[m]}(k)\right)^T \tag{3.26}$$

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

Розв'язати це завдання можна, знову скориставшись апаратом невизначених множників Лагранжа та адаптивного багатовимірного узагальненого прогнозування [48].

Введемо до розгляду вихідний сигнал нейрону $GMN^{[m]}$ у вигляді

$$\hat{y}^{*[m]}(k) = \sum_{j=1}^{q} c_j^{[m]} \hat{y}_j^{[m]}(k) = \hat{y}^{[m]}(k) c^{[m]}, \tag{3.27}$$

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 узагальнення, що відповідають умовам незміщенності

$$\sum_{j=1}^{q} c_j^{[m]} = E^T c^{[m]} = 1, \tag{3.28}$$

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

Введемо критерій навчання

$$\begin{aligned} E^{[m]}(k) &= \sum_{\tau=1}^{k} \left\| y(\tau) - \hat{y}^{[m]}(\tau) c^{[m]} \right\|^2 \\ &= Tr\left(\left(Y(k) - \hat{Y}^{[m]}(k) I \otimes c^{[m]} \right)^T \left(Y(k) - \hat{Y}^{[m]}(k) I \otimes c^{[m]} \right) \right) \end{aligned} \tag{3.29}$$

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

$$\hat{Y}^{[m]}(k) = \begin{pmatrix} \hat{y}_1^{[m]T}(1) & \hat{y}_2^{[m]T}(1) & \dots & \hat{y}_q^{[m]T}(1) \\ \hat{y}_1^{[m]T}(2) & \hat{y}_2^{[m]T}(2) & \dots & \hat{y}_q^{[m]T}(2) \\ \vdots & \vdots & & \vdots \\ \hat{y}_1^{[m]T}(k) & \hat{y}_2^{[m]T}(k) & \dots & \hat{y}_q^{[m]T}(k) \end{pmatrix}, \tag{3.30}$$

I – одинична $(g \times g)$ матриця,

$\otimes$ – символ тензорного добутку.

З урахуванням обмежень (3.28) запишемо функцію Лагранжа

$$\begin{aligned} L^{[m]}(k) &= E^{[m]}(k) + \lambda\left(E^T c^{[m]} - 1\right) \\ &= \sum_{\tau=1}^{k} \left\| y(\tau) - \hat{y}^{[m]}(\tau) c^{[m]} \right\|^2 + \lambda\left(E^T c^{[m]} - 1\right) \\ &= Tr\left(\left(Y(k) - \hat{Y}^{[m]}(k) I \otimes c^{[m]}\right)^T \left(Y(k) - \hat{Y}^{[m]}(k) I \otimes c^{[m]}\right)\right) \\ &+ \lambda\left(E^T c^{[m]} - 1\right) \\ &= Tr\left(V^{[m]T}(k) V^{[m]}(k)\right) + \lambda\left(E^T c^{[m]} - 1\right), \end{aligned} \tag{3.31}$$

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

$$\begin{cases} \nabla_{c^{[m]}} L^{[m]}(k) = \overrightarrow{0}, \\ \dfrac{\partial L^{[m]}(k)}{\partial \lambda} = 0 \end{cases} \tag{3.32}$$

призводить до очевидного результату

$$\begin{cases} c^{[m]} = \left(R^{[m]}(k)\right)^{-1} E\left(E^T \left(R^{[m]}(k)\right)^{-1}\right)^{-1} \\ \lambda = -2E^T \left(R^{[m]}(k)\right)^{-1} E, \end{cases} \tag{3.33}$$

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 $R^{n+(m-1)g} \to R^g$.

Висновки до розділу 3.1

1. Розглянута задача апроксимації та екстраполяції багатовимірних часових рядів за умови апріорної і поточної структурної та параметричної невизначеності; проаналізовані існуючі гібрідні системи обчислювального інтелекту, що використовуються для вирішення задач прогнозування та індентифікаціїї багатовимірних даних у пакетному режимі; сформовані вимоги та обмеження до шуканої гібридної системи, здатної реалізувати нелінійне відображення $R^n \to R^g$ у режимі реального часу.
2. Зсинтезовано каскадну архітектуру системи, що ґрунтується на нео-фаззі нейронах, здатну реалізувати нелінійне відображення $R^n \to R^g$ у режимі послідовного обробляння даних.
3. Запропоновано архітектуру багатовимірного нео-фаззі нейрона та метод його навчання, що забезпечують підвищену швидкість налаштування синаптичних ваг та додаткові згладжуючі властивості.
4. Запропоновано архітектуру та рекурентний метод навчання багатовимірного узагальнюючого елементу, що в режимі реального часу реалізує оптимальне об'єднання багатовимірних вихідних сигналів нейронів пулу каскаду.
5. Запропоновано MIMO архітектуру та методи навчання гібридної каскадної нейронної мережі з оптимізацією пулу багатовимірних нейронів у кожному каскаді, що реалізують оптимальний за точністю прогноз нелінійних стохастичних і хаотичних сигналів у онлайн режимі.

РОЗДІЛ 4
КАСКАДНА НЕЙРОННА МЕРЕЖА, ЩО ЕВОЛЮЦІОНУЄ, ДЛЯ ПОСЛІДОВНОГО НЕЧІТКОГО КЛАСТЕРУВАННЯ ПОТОКІВ ДАНИХ

У цьоому розділі описані архітектура та методи навчання пропонованої 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 описні два підходи, переваги яких поєднує у собі пропонована система: нечітке та ієрархічне кластерування.

4.1. Труднощі та особливості відомих методів кластерування даних

Завдання кластерування (класифікації 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, пропована 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 тренувальною вибірки розміру N [26].

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 у режимі реального часу).

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 інший. Крім того, ієрархічне кластерування є статичним, тобто образи віднесені до певного кластеру на ранніх стадіях не можуть бути пізніше належними іншому, що унеможливлює створення модифікацій методів для послідовного кластерування, на відміну від розподільного кластерування. Методи ієрархічного кластерування здебільшого мають обчислювальну складність принаймні $\mathcal{O}\left(N^2\right)$, що робить їх використання недоцільним для велких наборів даних.

4.1.1. Нечітке послідовне кластерування. 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 [1, 8, 28, 30, 50, 72], а для його вирішення широко використовується апарат обчислювального інтелекту [9-12] і, насамперед, нейро-фаззі підхід [13]. 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 $x(1), x(2), \ldots, x(1), ;x(N)$, 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 ($J = 1, 2, \ldots, M$). 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 (Dynamik Data Mining, Data Stream Mining) [2, 6, 20, 32, 44, 57, 70] 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 [40] 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 перети-

наються. 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 [7, 9, 22, 24, 31, 43, 53, 54, 59, 60, 63, 64]. Такі гібридні системи володіють обширною функціональністю завдяки використанню спеціальних алгоритмів налаштування, що ґрунтуються на процедурах оптимізації прийнятої цільової функції, але потребують попередньо заданої кількості класетрів та фіксованого значення фаззіфікатору.

4.2. 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 (Partition Entorpy Index) - ще один критерій дійсності нечіткого кластерування, запропонований (Bezdek, 1974a, 1981), що залежить лише від значень фунції належності

$$PE = -\frac{1}{N}\sum_{l=1}^{M}\sum_{i=1}^{N} u_{li}\log_a(u_{li}). \tag{4.1}$$

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 (4.1).

Фукуяма та Суґено запропонували індекс дійсності нечіткого кластерування, залежний як від рівнів належності так і від самих вхідних даних:

$$FS = \sum_{i=1}^{N}\sum_{l=1}^{M} u_{li}^{\beta}\big(\|x_i - z_l\|^2 - \|z_l - z\|^2\big), \tag{4.2}$$

де z та z_l – середнє арифметичне усієї виборки та образів віднесених до кластеру M_l відповідо. З визначення (4.2) видно, що малі значення FS говорять про компактні добре визначені кластери.

Нечітка множина i-ого образу визначається як

$$\tilde{A}_l = \sum_{i=1}^{N} \frac{u_{li}}{x_i}, l = 1, 2, \ldots, M. \tag{4.3}$$

Ступінь, в якій A_l є підмножиною A_p визначається наступним чином

$$\begin{cases} S\big(\tilde{A}_l, \tilde{A}_p\big) = \dfrac{U\big(\tilde{A}_l \cap \tilde{A}_p\big)}{U\big(\tilde{A}_l\big)}, \\ U\big(\tilde{A}_j\big) = \sum\limits_{i=1}^{N} u_{ji}. \end{cases} \tag{4.4}$$

Зважаючи на (4.4), можна запропонувати такі варіанти обчислення міри подібності:

$$N_1\big(\tilde{A}_l, \tilde{A}_p\big) = \frac{S\big(\tilde{A}_l, \tilde{A}_p\big) + S\big(\tilde{A}_p, \tilde{A}_l\big)}{2}, \tag{4.5a}$$

$$N_2\big(\tilde{A}_l, \tilde{A}_p\big) = \min\big(S\big(\tilde{A}_l, \tilde{A}_p\big), S\big(\tilde{A}_p, \tilde{A}_l\big)\big), \tag{4.5b}$$

$$N_3\big(\tilde{A}_l, \tilde{A}_p\big) = S\big(\tilde{A}_l \cup \tilde{A}_p, \tilde{A}_p \cap \tilde{A}_l\big). \tag{4.5c}$$

Тоді індекс дійсності кластерування, що грунтується на нечіткій подібності, можна визначити як

$$FSim = \max_{1\le l\le M} \max_{1\le p\le M, p\ne l} N\big(\tilde{A}_l, \tilde{A}_p\big), \tag{4.6}$$

де міру нечіткої подібності $N\big(\tilde{A}_l, \tilde{A}_p\big)$ можна знайти за будь-яким виразом (4.5).

4.3. Архітектура каскадної мережі, що еволюціонує, для нечіткого кластерування

До нульовго шару системи послідовно передаються дані у формі векторного сигналу $x(k) = (x_1(k), x_2(k), \ldots, x_1(k))^T$, де $k = 1, 2, \ldots, N, N+1, \ldots$ — індекс поточного дискретного часу. Вхідні сигнали надходять до всіх вузлів системи $N_j^{[m]}$, де $j = 1, 2, \ldots, q$ - кількість вузлів у пулі-ансамблі, $m = 1, 2, \ldots$ — номер каскаду. Вузол кожного каскаду призначений для онлайн кластерування потоку данних і відрізняється від вузлів-сусідів використаним алгоритмом 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 загальную якість кластеризації пула, враховуючи прийняту кількість кластерів $m+1$. Таким чином, система розв'язує задачу кластерування нестаціонарного потоку даних в умовах невизначенності щодо кількості кластерів, а також їх вигляду і рівню взаємного перекриття. І, нарешті, вихідний вузол системи XB^*, порівнюючи якість кластеризації 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, що надоходить до системи, може оброблятися у паралельному режимі вузлами системи $N_j^{[m]}$[2, 6].

4.4. Адаптивне навчання вузлів каскадної нейро-фаззі системи, що еволюціонує

В основі алгоритмів навчання вузлів системи лежать алгоритми нечіткого кластерування, засновані на цільових функціях, такі, що вирішують задачу їх оптимізації при деяких апріорних припущеннях. Найбільш поширеним є

ймовірнісний підхід, заснований на мінімізації цільової функції

$$E\Big(u_{jl}^{[m]}(k),\, c_{jl}^{[m]}\Big) = \sum_{k=1}^{N}\sum_{l=1}^{m+1}\Big(u_{jl}^{[m]}(k)\Big)^{\beta}\Big\|x(k) - c_{jl}^{[m]}\Big\|^2 \tag{4.7}$$

при обмеженнях

$$\sum_{l=1}^{m+1}(k) = 1, \quad 0 \le \sum_{k=1}^{N} u_{jl}^{[m]}(k) \le N \tag{4.8}$$

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,

$k = \overline{1, N}$ — номер образу (N — кількість образів у вхідній виборці, що, у рамках класичного подходу Бездека, вважається незмінною та такою, що задана апріорі).

Вводячи функцію Лагранжа

$$\begin{aligned} L\Big(u_{jl}^{[m]}(k),\, c_{jl}^{[m]},\, \lambda_j^{[m]}(k)\Big) = \sum_{k=1}^{N}\sum_{l=1}^{m+1}\Big(u_{jl}^{[m]}(k)\Big)^{\beta}\Big\|x(k) - c_{jl}^{[m]}\Big\|^2 + \\ + \sum_{k=1}^{N}\lambda_j^{[m]}(k)\Bigg(\sum_{l=1}^{m+1}u_{jl}^{[m]}(k) - 1\Bigg) \end{aligned} \tag{4.9}$$

(тут $\lambda_j^{[m]}(k)$ — невизначений множник Лагранжа) та вирішивши систему рівнянь Каруша-Куна-Таккера, нескладно отримати шукане рішення у вигляді

$$\begin{cases} u_{jl}^{[m]}(k) = \dfrac{\left(\left\|x(k) - c_{jl}^{[m]}\right\|^2\right)^{\frac{1}{1-\beta}}}{\sum\limits_{l=1}^{m+1}\left(\left\|x(k) - c_{jl}^{[m]}\right\|^2\right)^{\frac{1}{1-\beta}}}, \\ c_{jl}^{[m]} = \dfrac{\sum\limits_{k=1}^{N}\left(u_{jl}^{[m]}(k)\right)^{\beta} x(k)}{\sum\limits_{k=1}^{N}\left(u_{jl}^{[m]}(k)\right)^{\beta}}, \\ \lambda_j^{[m]}(k) = -\left(\left(\sum\limits_{l=1}^{m+1}\beta\left\|x(k) - c_{jl}^{[m]}\right\|^2\right)^{\frac{1}{1-\beta}}\right)^{\frac{1}{1-\beta}}, \end{cases} \tag{4.10}$$

що при $\beta = 2$ збігається з алгоритмом нечітких с-середніх Бездека (FCM) [29] і приймає форму

$$\begin{cases} u_{jl}^{[m]}(k) = \dfrac{\left\|x(k) - c_{jl}^{[m]}\right\|^{-2}}{\sum\limits_{l=1}^{m+1}\left\|x(k) - c_{jl}^{[m]}\right\|^{-2}}, \\ c_{jl}^{[m]} = \dfrac{\sum\limits_{k=1}^{N}\left(u_{jl}^{[m]}(k)^2 x(k)\right)}{\sum\limits_{k=1}^{N}\left(u_{jl}^{[m]}(k)\right)^2}. \end{cases} \tag{4.11}$$

Тут варто відзначити, що вибір фаззіфікатора $\beta = 2$ в (4.11) не дає жодних переваг порівняно з довільним значенням β у (4.10), 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 [52, 56, 67].

Для послідовної обробки потоку даних, що надходять в online режимі, у [10, 14] були запропоновані рекурентні алгоритми, в основі яких лежить процедура нелінійного програмування Ерроу-Гурвіца-Удзави [5]. Так, пакетному алгоритмові (4.10) відповідає вираз

$$\begin{cases} u_{jl}^{[m]}(k+1) = \dfrac{\left\| x(k+1) - c_{jl}^{[m]}(k) \right\|^{\frac{1}{1-\beta}}}{\sum\limits_{l=1}^{m+1} \left\| x(k+1) - c_{jl}^{[m]}(k) \right\|^{\frac{1}{1-\beta}}}, \\ c_{jl}^{[m]}(k+1) = c_{jl}^{[m]}(k) + \eta(k+1)\Big(u_{jl}^{[m]}(k+1)\Big)^{\beta_j}\Big(x(k+1) - c_{jl}^{[m]}(k)\Big), \end{cases} \tag{4.12}$$

(тут $\eta(k+1)$ — параметр кроку навчання), що є узагальненням алгоритму навчання Чанга-Лі [21] і при $\beta = 2$ близьке до ґрадієнтної процедури Парка-Деера [53].

$$\begin{cases} u_{jl}^{[m]}(k+1) = \dfrac{\left\| x(k+1) - c_{jl}^{[m]}(k) \right\|^{-2}}{\sum\limits_{l=1}^{m+1} \left\| x(k+1) - c_{jl}^{[m]}(k) \right\|^{-2}}, \\ c_{jl}^{[m]}(k+1) = c_{jl}^{[m]}(k) + \eta(k+1)\Big(u_{jl}^{[m]}(k+1)\Big)^{2}\Big(x(k+1) - c_{jl}^{[m]}(k)\Big). \end{cases} \tag{4.13}$$

Варто зауважити, що, розглянувши співвідношення (4.12) з позицій навчання Кохоненової самоорганізованої мапи (SOM) [40], можна помітити, що множник $\left(u_{jl}^{[m]}\right)^{\beta_j}$ відповідає функції сусідства в правилі навчання на основі принципу «переможцю дістається більше», маючи при цьому дзвонуватий вигляд.

Вочевидь, у випадку, коли $\beta_j = 1$ та $u_{jl}^{[m]}(k) \in [0,1]$, процедура (4.12) збігається з чітким алгоритмом c-середніх (HCM), коли ж $\beta_j = 0$, маємо стандартне правило навчання Когонена «переможцю дістається все» [40]

$$c_{jl}^{[m]}(k+1) = c_{jl}^{[m]}(k) + \eta(k+1)\Big(x(k+1) - c_{jl}^{[m]}(k)\Big), \tag{4.14}$$

запропоноване Каш'япом та Блейдоном [39] у шістдесятих роках минулого століття. Легко побачити, що процедура (4.14) оптимізує цільову функцію

$$E\left(c_{jl}^{[m]}\right) = \sum_{k=1}^{N} \left\| x(k) - c_{jl}^{[m]} \right\|^2, \quad \sum_{l=1}^{m+1} N_l = N, \tag{4.15}$$

мінімум якої збігається із середнім арифметичним

$$c_{jl}^{[m]} = \frac{1}{N} \sum_{k=1}^{N_l} x(k), \tag{4.16}$$

де N_l — кількість векторів, віднесених до l-го кластеру у процесі конкуренції.

Якщо записати (4.16) у рекурентній формі, отримаємо оптимальний алгоритм самонавчання Ципкіна [25]

$$c_{jl}^{[m]}(k+1) = c_{jl}^{[m]}(k) + \frac{1}{N_l(k+1)}\left(x(k+1) - c_{jl}^{[m]}(k)\right), \tag{4.17}$$

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 алгоритм навчання (4.12) вузла можна розглядати як правило самонавчання нечіткої модифікації самоорганізовної мапи Когонена, архітектура якої наведена на рис

TODO:рис

Тут $N_{jl}^{[m]K}$ — стандартні нейрони Когонена, пов'язані між собою латеральними зв'язками, що налаштовуються згідно "переможцю дістається більше"правилу навчання на основі другого співвідношення (4.12). Вузли $N_{jl}^{[m]u}$ обчислюють рівні належності згідно першому співвідношенню (4.12). Вузли $N_j^{[m]}$ кожного з каскадів відрізняються тільки фаззіфікатором алгоритму самонавчання, а вузол кожного наступного каскаду містить додатково один нейрон Когонена і один елемент для розрахунку рівнів належності.

4.5. Керування каскадами самонавчанної нейро-фаззі системи, що еволюціонує

Якість кластерування кожого вузла системи може бути оцінена за допомогою будь-якого з індексів, що використовуються у задачах нечіткого кластерування [72]. Одим 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

$$PC_j^{[m]} = \frac{1}{N}\sum_{k=1}^{N}\sum_{l=1}^{m+1}\left(u_{jl}^{[m]}(k)\right)^2. \tag{4.18}$$

Цей коефіцієнт має ясний фізичний зміст: щокраще 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 online режимі

$$PC_j^{[m]}(k+1) = PC_j^{[m]}(k) + \frac{1}{k+1}\left(\sum_{l=1}^{m+1}\left(u_{jl}^{[m]}(k+1)\right)^2 - PC_j^{[m]}(k)\right). \tag{4.19}$$

Розрахунок коефіцієнту розбиття проводиться для кожного вузла системи разом з налаштуванням їх параметрів, тобто співвідношення (4.12) та (4.19) 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 [71], який для фіксованої вибірки з N спостережень може бути записаний у вигляді

$$XB_j^{[m]} = \frac{\left(\sum\limits_{k=1}^{N}\sum\limits_{l=1}^{m+1}\left(u_{jl}^{[m]}(k)\right)^2\left\|x(k)-c_{jl}^{[m]}\right\|^2\right)\Big/N}{\min\limits_{l\neq q}\left\|c_{jl}^{[m]}-c_{jq}^{[m]}\right\|^2} = \frac{NXB_j^{[m]}}{DXB_j^{[m]}} \tag{4.20}$$

Вираз (4.20) також можна записати у рекурентній формі

$$XB_j^{[m]}(k+1) = \frac{NXB_j^{[m]}(k+1)}{DXB_j^{[m]}(k+1)} = \frac{NXB_j^{[m]}(k)+\frac{1}{k+1}\left(\sum\limits_{l=1}^{m+1}\left(u_{jl}^{[m]}(k+1)\right)^2\left\|x(k+1)-c_{jl}^{[m]}(k+1)\right\|^2-NXB_j^{[m]}(k)\right)}{\min\limits_{l\neq q}\left\|c_{jl}^{[m]}(k+1)-c_{jq}^{[m]}(k+1)\right\|^2}, \tag{4.21}$$

при цьому рекурентні вирази (4.19) та (4.21) реалізуються одночасно.

Індекс Ксі-Бені є по суті співвідношенням відхилення всередині кластерів $NXB_j^{[m]}$ до величини поділу кластерів $DXB_j^{[m]}$. Оптимальному числу кластерів у каскаді відповідає мінімальне значення (4.20) та (4.21). 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 фаззифікатору, ефективність роботи кожного каскаду доцільно оцінювати за допомогою розширеного індексу Ксі-Бені EXB [72].

$$EXB_j^{[m]} = \frac{\left(\sum\limits_{k=1}^{N}\sum\limits_{l=1}^{m+1}\left(u_{jl}^{[m]}(k)\right)^{\beta[m]}\left\|x(k)-c_{jl}^{[m]}\right\|^2\right)\Big/N}{\min\limits_{l\neq q}\left\|c_{jl}^{[m]}-c_{jq}^{[m]}\right\|^2} = \frac{NEXB_j^{[m]}}{DEXB_j^{[m]}} \tag{4.22}$$

або його рекурентої форми

$$XB_j^{[m]}(k+1) = \frac{NXB_j^{[m]}(k+1)}{DXB_j^{[m]}(k+1)} = \frac{NXB_j^{[m]}(k)+\frac{1}{k+1}\left(\sum\limits_{l=1}^{m+1}\left(u_{jl}^{[m]}(k+1)\right)^{\beta[m]}\left\|x(k+1)-c_{jl}^{[m]}(k+1)\right\|^2-NXB_j^{[m]}(k)\right)}{\min\limits_{l\neq q}\left\|c_{jl}^{[m]}(k+1)-c_{jq}^{[m]}(k+1)\right\|^2}, \tag{4.23}$$

де $\beta^{[m]}$ — 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 обробку в онлайн режимі.

Висновки до розділу 4

1. Розглянуто завдання нечіткого кластерування у режимі послідовного надходження даних до системи.
2. Запропоновано метод визначення локально оптимальної кількості кластерів і значення параметру фаззіфікації для послідовного кластерування потоків даних.
3. Запропоновано архітектуру і метод самонавчання каскадної нейрофаззі системи, що еволюціонує, для послідовного кластерування потоків даних з автоматичним визначенням оптимальної кількості кластерів. Кожен вузол кожного каскаду системи вирішує завдання кла-

стерування незалежно від інших, що дозволяє організувати паралельну обробку інформації в каскадах, тобто підвищити швидкодію цього процесу. Система не містить жодних порогових параметрів, що задаються суб'єктивно, а процес оцінювання якості її функціонування визначається шляхом відшукання оптимального значення певного індексу дійсності розбиття даних на кластери (їх поточна оцінка також проводиться в режимі реального часу). Відмінною особливістю пропонованої системи є те, що вона самостійно визначає і поточне значення фаззіфікатору, і оптимальну кількість кластерів на кожному етапі обробляння даних.

РОЗДІЛ 5
МОДЕЛЮВАННЯ ТА ПРАКТИЧНЕ ЗАСТОСУВАННЯ РОЗРОБЛЕНИХ МЕТОДІВ ТА АРХІТЕКТУР

5.1. Моделювання самонавчанної нейро-фаззі системи, що еволюціонує

5.1.1. Придумати назву1. Одна з основних переваг, притаманних пропонованій самонавчанній нейро-фаззі системі, що еволюціонує, полягає в автоматичному визначенні оптимальної кількості кластерів та значення фаззифікатору на кожному етапі обробляння даних. Першу серію експериментів було проведено на штучно зсинтезованих наборах даних з різним ступенем розмитості та перекриття класів аби дослідити вплив значення параметру фаззіфікації на якість кластерування в режимі реального часу відповідно до обраного критерію дійсності.

$$= 6$$

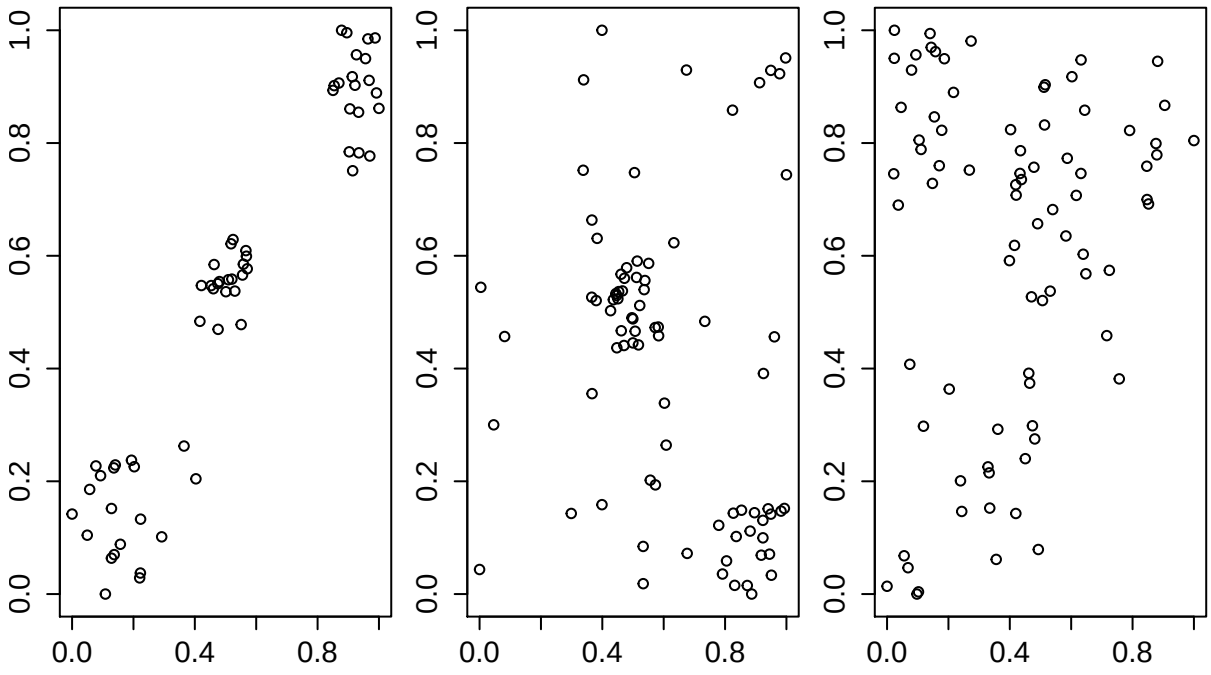

Рис. 5.1. Штучно сгенеровані набори даних

Кожен з наборів даних, що їх наведено на рис. 5.1, містить 80 спостережень з 2 ознаками (для очності) у кожному спостереженні. Тестові дані були сгенеровані таким чином, аби у першому наборі класи були чітко розподілені (crisp dataset), у другому наборі кластерні границі були дещо розмиті (fuzzy dataset), у третьому випадку класи сильно перетиналися (extra fuzzy dataset). Логічно припустити, що система, яка тестується, обере менше значення параметру фаззіфікації для першого датасету та більше для останнього, де границі класів спостережень є більш розмитими.

Спостереження надходили до нейро-фаззі мережі у послідовному режимі, вагові коефіцієнти нейронів були проініціалізовані, використовуючи пакетну модифікацію обраного алгоритму кластерування на датасеті з довільних двадцятьох спостережень відповідного набору даних (адже система, як і класичний fuzzy c-means, чутлива до ініціалзації). Локально оптимальні кількість кластерів та значення параметру фаззіфікації обумовлювалися максимальним середнім значенням рекурентних коефіцієнту розбиття PC (4.19) та Ксі-Бені індексу (4.21): $\max \dfrac{PC_j^{[m]} + 1 - XB_j^{[m]}}{2}$ (у данному випадку використовувалося від'ємне значення Ксі-Бені індексу $1 - XB(k)$, оскільки щоменше $XB_j^{[m]}$, то ліпшим є розбиття даних на кластери).

Для першого набору даних (crsip dataset), як і передбачалося, оптимальним виявися другий каскад ($m = 3$) з трьома кластерами і нейроном-переможцем із найменшим значенням параметру фаззіфікації $\beta = 2$ (рис. 5.3). Така конфігурація є оптимальною відповідно до обох використовуваних індексів валідності – найменше значення Ксі-Бені індексу $XB_j^{[m]}$ та найбільший коефіцієнт розбиття $PC_j^{[m]}$:

$$PC_1^{[2]} = 0.9009951,$$
$$XB_1^{[2]} = 0.03349166.$$

Лише одне спостереження у цьому датасеті (його позначено багряним квадратом) не належить жодному кластерові з ступінем більшим від 0.6. Індекси

Каскад 1 ($m = 2$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	0.91758	0.7446	0.64787	0.59236
Індекс Ксі-Бені	0.052129	0.061034	0.092235	0.1294
Каскад 2 ($m = 3$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	0.92643	0.6609	0.50214	0.43305
Індекс Ксі-Бені	0.027232	0.06872	0.17281	0.26914
Каскад 3 ($m = 4$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	0.87218	0.5256	0.37605	0.31993
Індекс Ксі-Бені	0.15687	0.4153	0.84699	1.1765
Каскад 4 ($m = 5$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	0.73909	0.45445	0.32428	0.27063
Індекс Ксі-Бені	0.12985	0.30637	0.68584	1.0551

Таблиця 5.1

Індекси валідності (датасет 1)

валідності нейронів системи наведені у таблиці 5.1.

Для набору даних з середньою вираженістю класів найліпшим виявився нейрон другого каскаду ($m = 3$) і фаззіфікатором $\beta = 3$ (таблиця 5.2).

Як показано на рис. 5.2, декілька спостережень у центрі (позначені багряними квадратами) можна віднести до 2 кластерів з відносно високим ступінем належності, проте більшість спостережнь можна чітко розкластеризувати, що ілюструється високим значенням коефіцієнту розбиття, та дуже низьким Ксі-Бені індексом:

$$PC_2^{[2]} = 0.9727868,$$
$$XB_2^{[2]} = 0.087474.$$

Для набору з найменш чіткими межами класів (таблиця 5.3), система обрала нейроном-переможцем вузол третього каскаду ($m = 4$) з високим параметром фаззіфікації $\beta = 4$:

$$PC_3^{[3]} = 0.335525,$$
$$XB_3^{[3]} = 0.2128333.$$

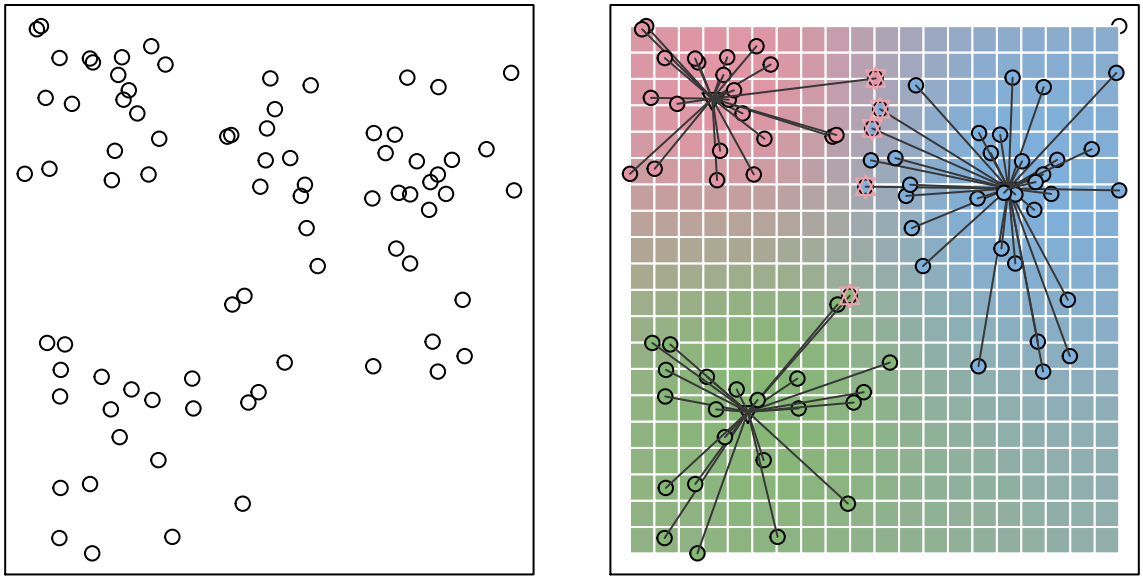

Рис. 5.2. Набір даних з нечіткими межами класів (fuzzy dataset)

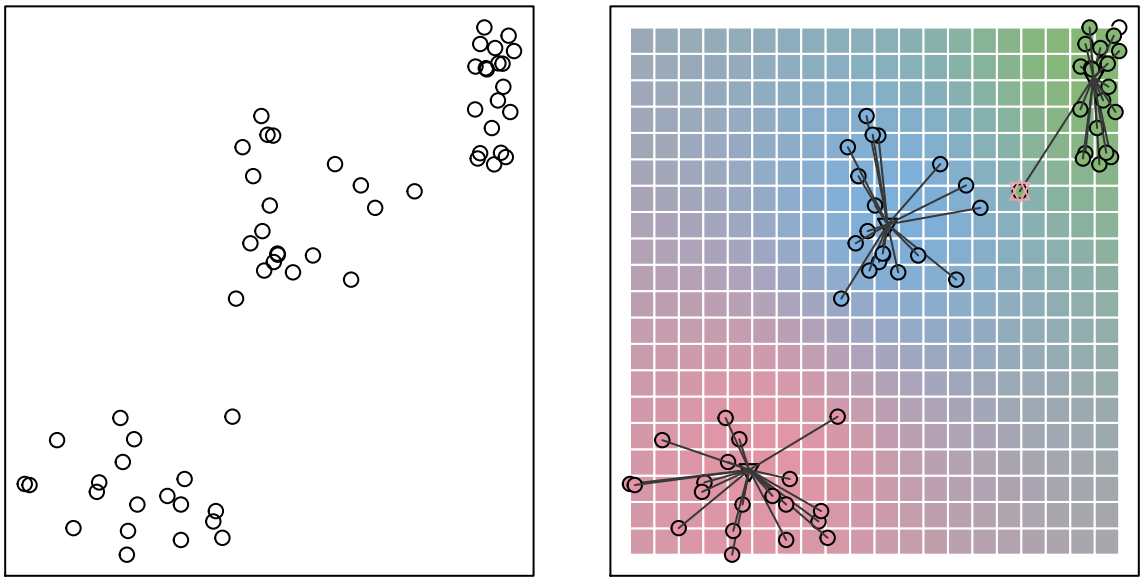

Рис. 5.3. Набір даних з чітко вираженими класами (Crisp dataset)

Каскад 1 ($m = 2$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	0.78414	0.58928	0.53853	0.52239
Індекс Ксі-Бені	0.16668	0.30834	0.3745	0.38723
Каскад 2 ($m = 3$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	50084	0.71164	0.97275	0.4191
Індекс Ксі-Бені	0.009751	0.031235	0.087474	0.1323
Каскад 3 ($m = 4$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	0.91888	0.47532	0.32777	0.28912
Індекс Ксі-Бені	0.052563	0.1757	0.27516	0.33766
Каскад 4 ($m = 5$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	0.85618	0.34327	0.24778	0.22445
Індекс Ксі-Бені	0.048316	0.19887	0.34307	0.41228
Каскад 5 ($m = 6$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	0.81295	0.30709	0.21636	0.19214
Індекс Ксі-Бені	0.060896	0.19702	0.31393	0.38668

Таблиця 5.2

Індекси валідності (датасет 2)

На рис. 5.4 спостереження, для яких ступінь належності до будь-якого кластеру не перевищує 0.6, позначені багряними квадратами. Як і очікувалося, для цього набору даних кількість таких спостережень значно вища від попередніх датасетів з більш компактними та «чіткими» класами.

Для очності у всіх наведених рисунках кольором позначені не тільки розкластеровані спостереження і центри кластерів, а й задній план (фон) малюнків, що дозволяє візуально визначити, до якого кластеру система віднесла б нові спостереження. Не дивно, що, тоді як для перших двох датасетів важко визначити домінуючий колір, оскільки кластери їх спостережень більш менш компактні та явно виражені, для останнього набору даних домінуючий колір – сірий, сформований кольорами усіх кластерів, що ілюструє великий ступінь перекриття класів і, відповідно, високе значення оптимального параметру фаззіфікації β, що обрала система.

Ця низка експериментів проілюструвала як важливо вірно визначати параметр фаззіфікації, оптимальне значення якого у випадку обробляння даних

Каскад 1 ($m = 2$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	0.85094	0.71415	0.61734	0.57085
Індекс Ксі-Бені	0.10584	0.11462	0.13797	0.16101
Каскад 2 ($m = 3$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	0.61668	0.42848	0.37779	0.35884
Індекс Ксі-Бені	0.1754	0.20364	0.22364	0.23995
Каскад 3 ($m = 4$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	0.33458	0.44082	0.79405	0.29615
Індекс Ксі-Бені	0.20989	0.129	0.051039	0.26282
Каскад 4 ($m = 5$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	0.50244	0.33067	0.26029	0.23318
Індекс Ксі-Бені	0.37268	0.61417	0.79695	0.93626
Каскад 5 ($m = 6$)	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
Коефіцієнт розбиття	0.53279	0.29731	0.22648	0.19858
Індекс Ксі-Бені	0.27407	0.47298	0.60569	0.70716

Таблиця 5.3

Індекси валідності (датасет 3)

у послідовному режимі з високою вирогідністю змінюється у часі, а саме здатність визначати оптимальне значення цього параметру в онлайн режимі є відмінною особливістю попропонованої самонавчанної нейро-системи.

5.1.2. Придумати назву2. Наступна низка експериментів була проведена на наборі даних «Іриси Фішера» (Fisher's Iris data set).

Це багатовимірний датасет для задачі класифікації, на прикладі якого англійський статистик та біолог Рональд Фішер в 1936 році продемонстрував роботу розробленого ним методу дискримінантного аналізу. Іноді його також називають «Ірисами Андерсона» (через те, що дані були зібрані американським ботаніком Едгаром Андерсоном). Цей набір даних став класичним і часто використовується в літературі для ілюстрації роботи різних статистичних алгоритмів.

Проте цей датасет рідко використовується у кластерному аналізі, адже межі класів «Verginica» та «Versicolor» не можна чітко визначити, ґрунту-

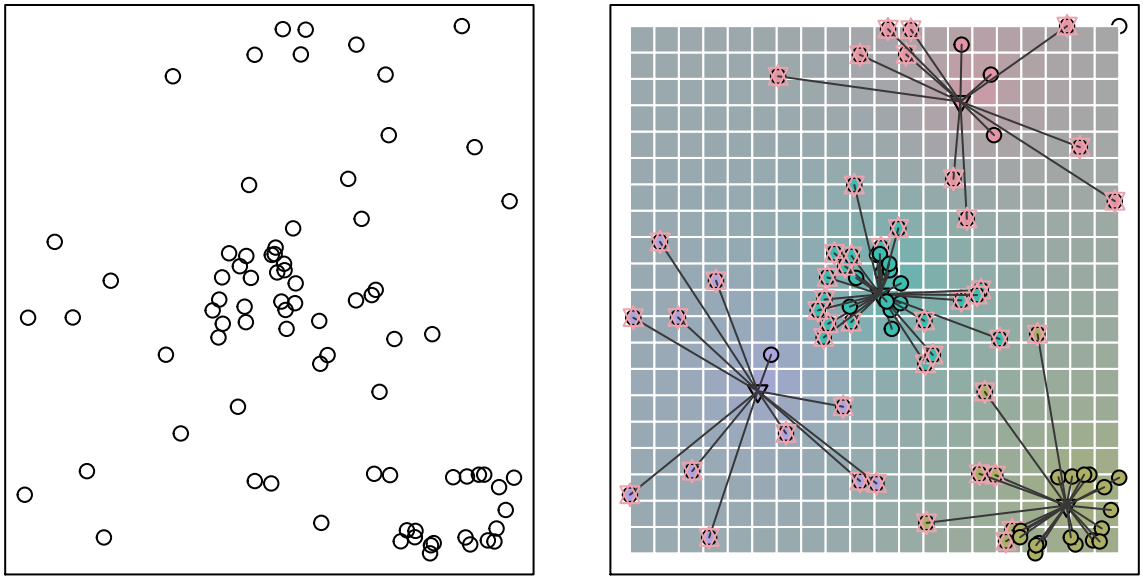

Рис. 5.4. Набір даних з класами, що перетинаються (extra fuzzy dataset)

ючись на даних, що їх використовував Фішер (що легко продемонструвати за допомогою ієрархічного кластерування, рис. 5.5). Саме цим і цікавий для нас цей набір даних: коли класичні методи чіткого кластерного аналізу не справляються з задачею, може стати у нагоді система, що реалізує нечітке кластерування зі змінним параметром фаззифікації та кількістю кластерів. Для більшості методів кластерного аналізу, зокрема для методу нечітких середніх (fuzzy c-means), необхідно заздалегідь задати кількість кластерів, і очевидним рішенням є прийняти $m = 3$, адже маємо три класи: Iris Verginica, Iris Versicolor та Iris Setosa (рис. 5.6).

	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
avg	0.8313073	0.8741245	0.8709475	0.8888124
min	0.7859722	0.7533766	0.6615745	0.7656498
max	0.8534013	0.9166667	0.935051	0.9604701

Таблиця 5.4

Точність кластерування при $m = 3$

Точність кластерування за допомогою методу нечітких середніх за таких умов ($m = 3$, $\beta = 2$) рідко перевищує 83% (таблиця 5.4). (Оскільки для обра-

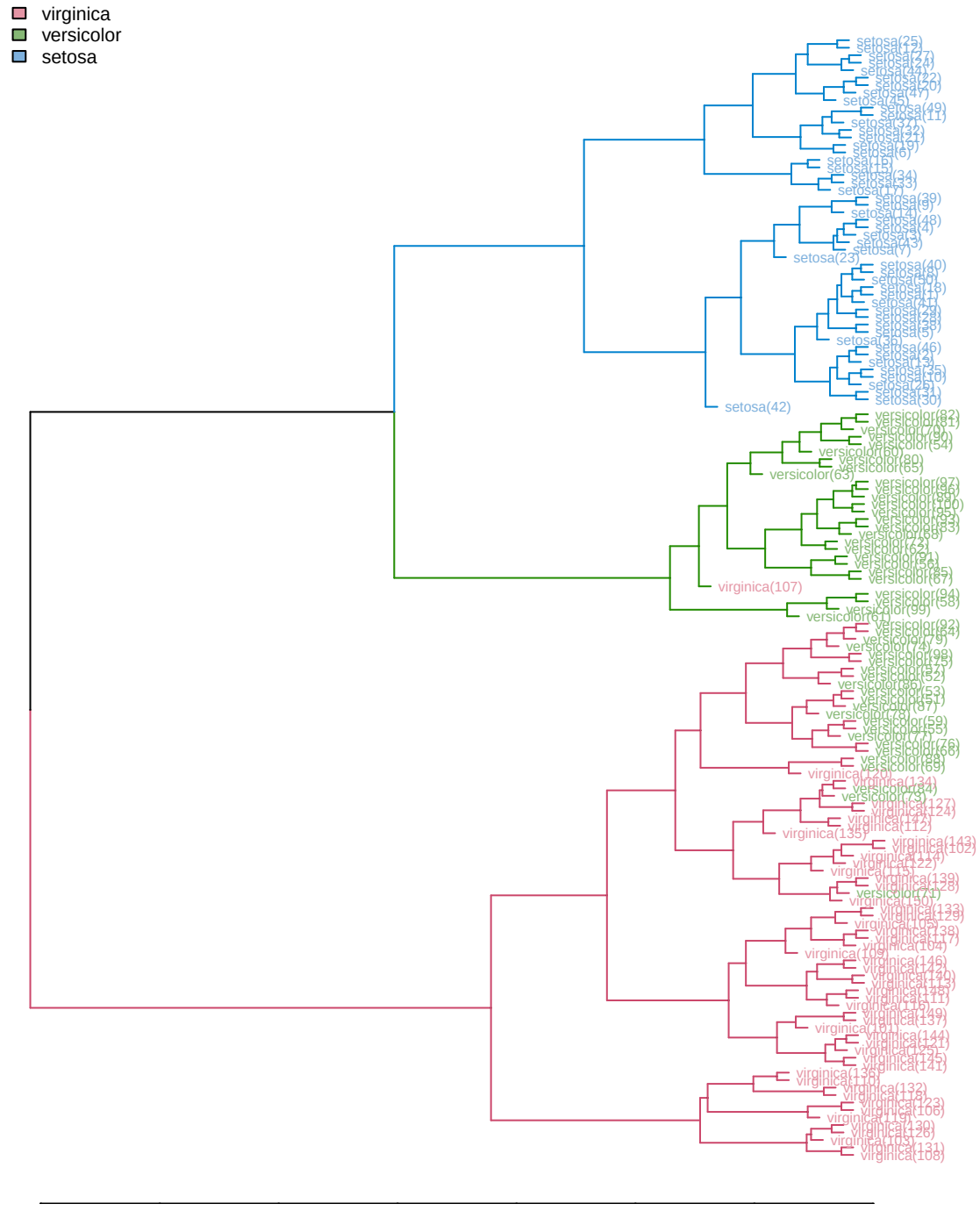

Рис. 5.5. Ієрархічне класерування датасету «Іриси Фішера»

ного датасету існують мітки з вірною класифікацією, ефективність кластеризації вимірювалася у відсотках точності щодо еталонного розбиття після дефаззіфікації.) Проте, якщо не обмежувати пропоновану систему у кількості кластерів (система ініціалізується інтервалом допустимих значень m (кількість кластерів) та параметру фаззифікації β), вельми цікавими є результати кластерування нейронів кожного з каскадів.

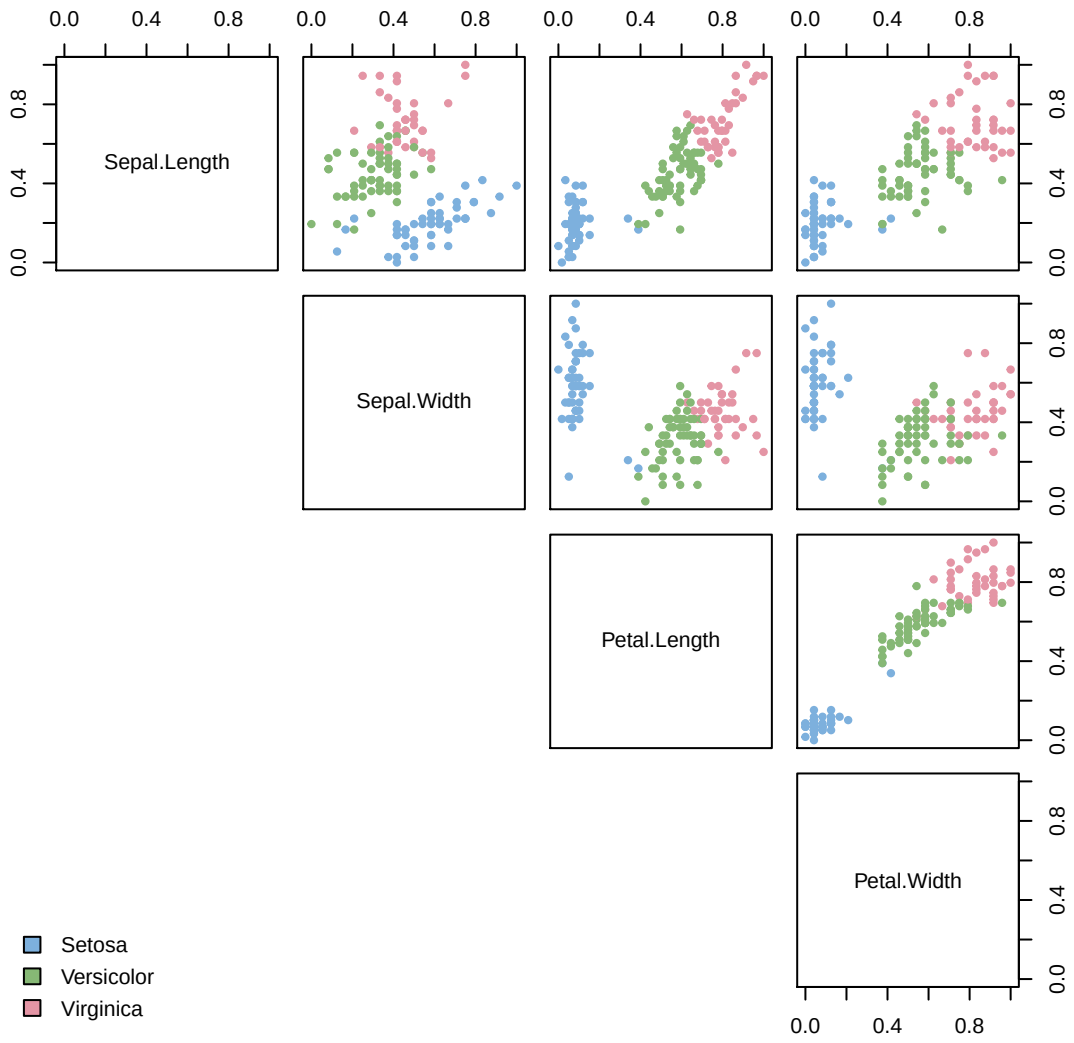

Рис. 5.6. Розкластерований датасет «Іриси Фішера» при $m = 3$, $\beta = 2$ (Точність кластерування – 96%)

У таблицяі 5.5 наведена точність розбиття даних, коли $m \gg 3$ кластерів відповідно. Варто зазначити, що нейрони у пулі кожного каскаду реалізують метод нечітких середніх зі змінним значення фазифікатору, а отже є чутливими до довільно ініціалізовних цетрах кластерів, тому у таблицях наведені середня, мінімальна та максимальна точності кластерування (після дефаззіфікації).

$m = 7$	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
avg	0.8972948	0.9150268	0.9242503	0.9178207
min	0.8536056	0.8461905	0.8723182	0.8600289
max	0.9621849	0.9736172	0.9810146	0.9663462
$m = 8$	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
avg	0.9065560	0.9296311	0.9243606	0.9248976
min	0.8217056	0.8562179	0.8577202	0.8590278
max	0.9474588	0.9789402	0.9848214	0.9747899
$m = 9$	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
avg	0.9258154	0.9282887	0.9308971	0.9229753
min	0.8689921	0.8270525	0.8684641	0.8556390
max	0.9849170	0.9806397	0.9664112	0.9748284
$m = 10$	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
avg	0.9213191	0.9285106	0.9332528	0.9282907
min	0.8663370	0.8722271	0.8652272	0.8766667
max	0.9663420	0.9838095	0.9723656	0.9756335
$m = 11$	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
avg	0.9295315	0.9408977	0.9317242	0.9295800
min	0.8520268	0.8964924	0.8890781	0.8788656
max	0.9716166	0.9848485	0.9704892	0.9798627
$m = 12$	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
avg	0.9349407	0.9433244	0.9337934	0.9306632
min	0.8815133	0.8949802	0.8798160	0.8486111
max	0.9795274	0.9783497	0.9630952	0.9772727
$m = 13$	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
avg	0.9420998	0.9398127	0.9375204	0.9357708
min	0.8823175	0.8614025	0.8882479	0.8828348
max	0.9807518	0.9788034	0.9753452	0.9748873

Таблиця 5.5

Точність кластерування для $m \in [7, 13]$, $\beta \in [2, 5]$

На рис 5.7 зображено залежність точності кластерування від кількості кластерів. Цікаво, що при, здавалося б, очевидному рішенні обрати кількість кластерів рівною трьом, отримуємо чиненайгіршу точнічть кластерування (при $\beta = 2$) після дефаззіфікації щодо еталонного розбиття (Для порівнян-

ня на рис. 5.8 та рис. 5.9 наведені розбиття, що їх запропонували нейроні-переможці деяких каскадів, де $m \gg 3$).

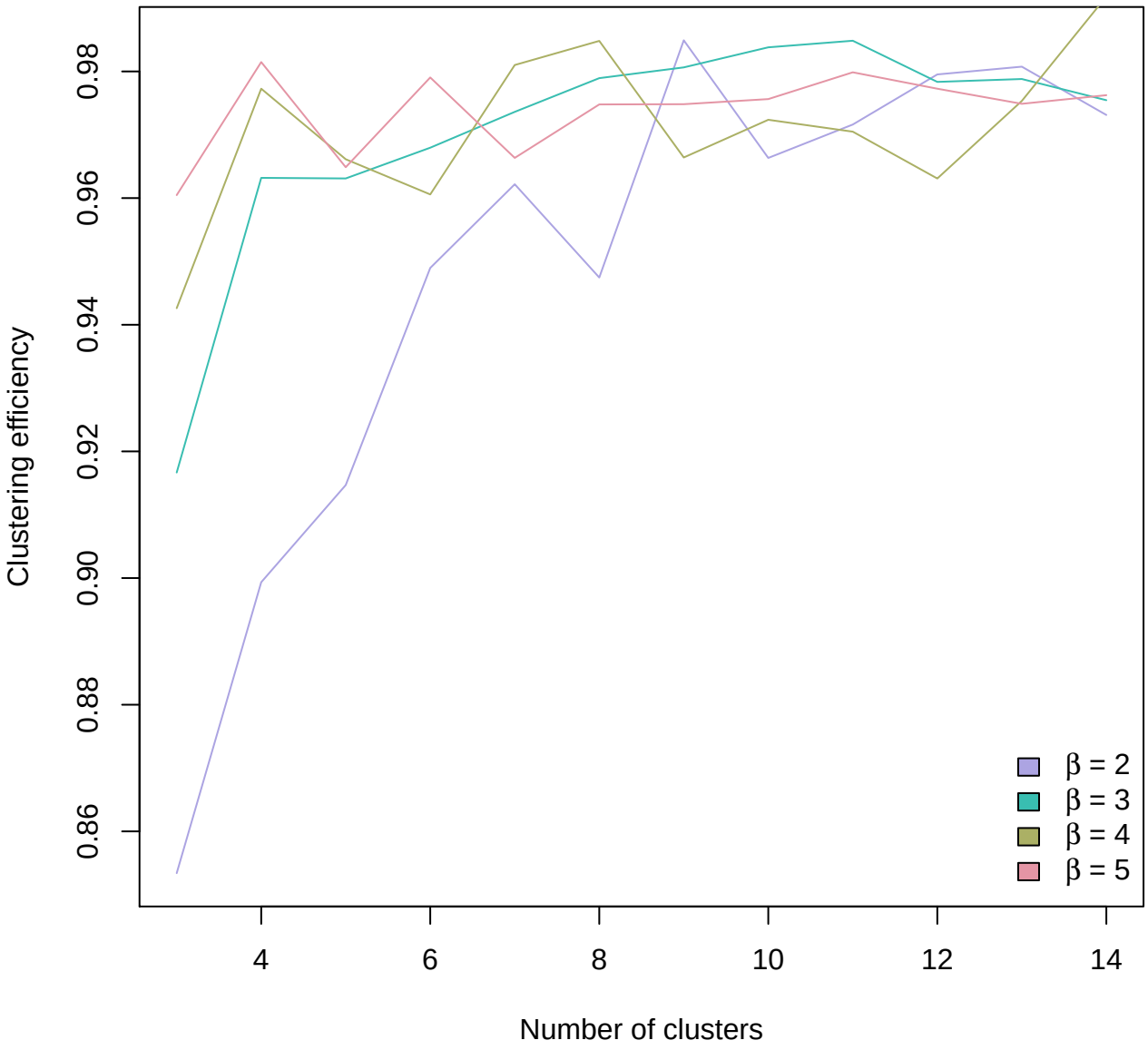

Рис. 5.7. Точність кластерування від кількості кластерів та параметру фаззіфікації

Цьому легко знайти пояснення, адже метод нечітких k-середніх (а саме цей метод у цьому експерименті реалізовують вузли пулів кожного каскаду) добре розпізнає кластери лише гіперсферичної форми. Проте кластер довільної (негіперсферичної) форми, можна розбити на декілька гіперсферичних

підкластерів, що й відбувається у каскадах, де $m > 3$, що пропонують розбиття на дрібні кластери. На рисунках 5.10 та 5.11 наведені розбиття деяких каскадів, де кількість кластерів більша від кількості класів еталонної вибірки; тут можна побачити, що декілька кластерів, що після дефазифікації будуть віднесені до одного класу, наприклад, Iris Virginica розташовані поруч один з одним, тобто є складовими більшого кластеру негіперсферичної форми.

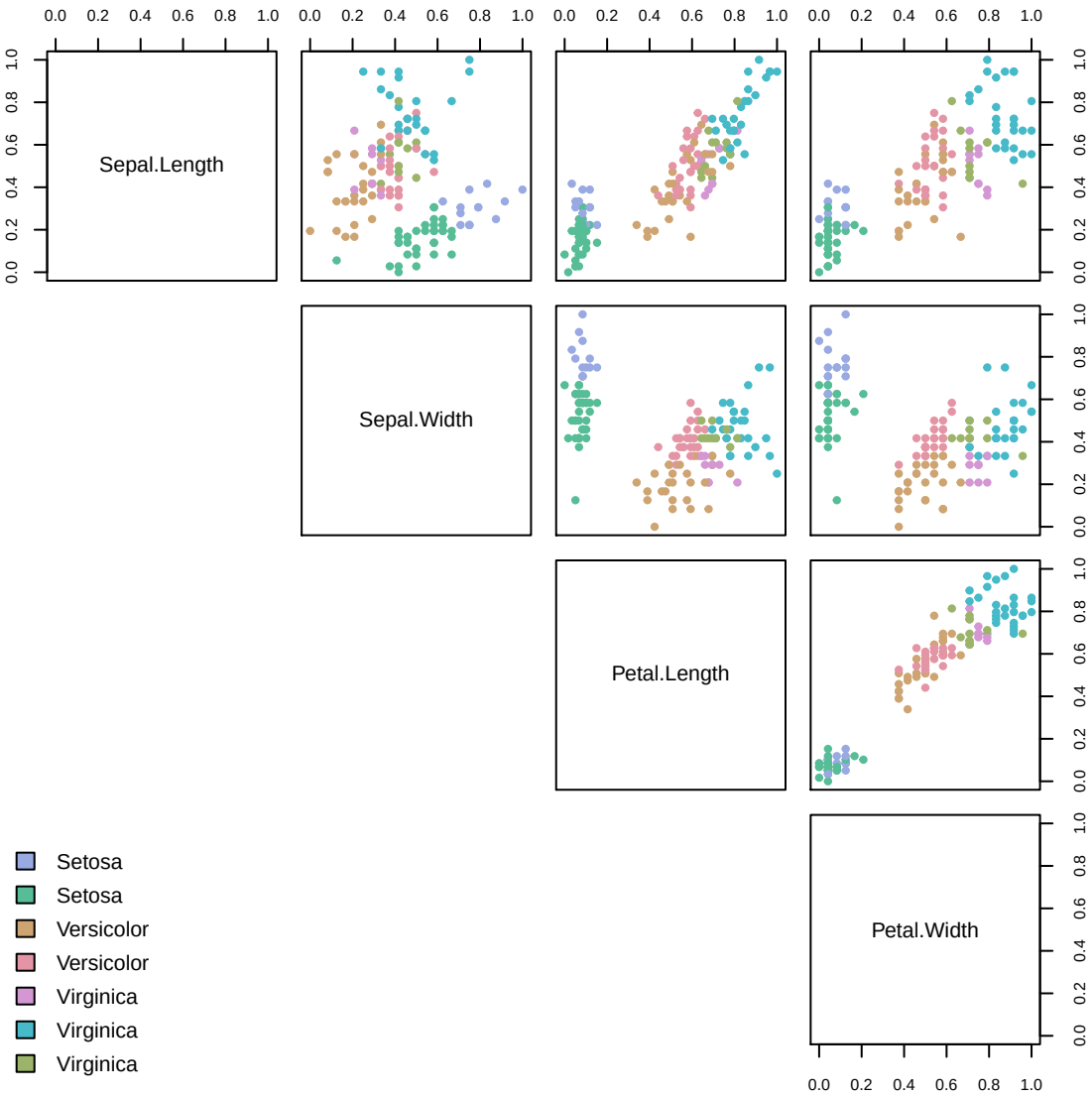

Рис. 5.8. Розкластерований датасет «Іриси Фішера» при $m = 7$, $\beta = 5$ (Точність кластерування $\approx 93\%$)

Таким чином, видається доречним, навіть у випадку, коли відоме еталонне розбиття датасету, дозволити системі обрати кінцеву кількість кластерів

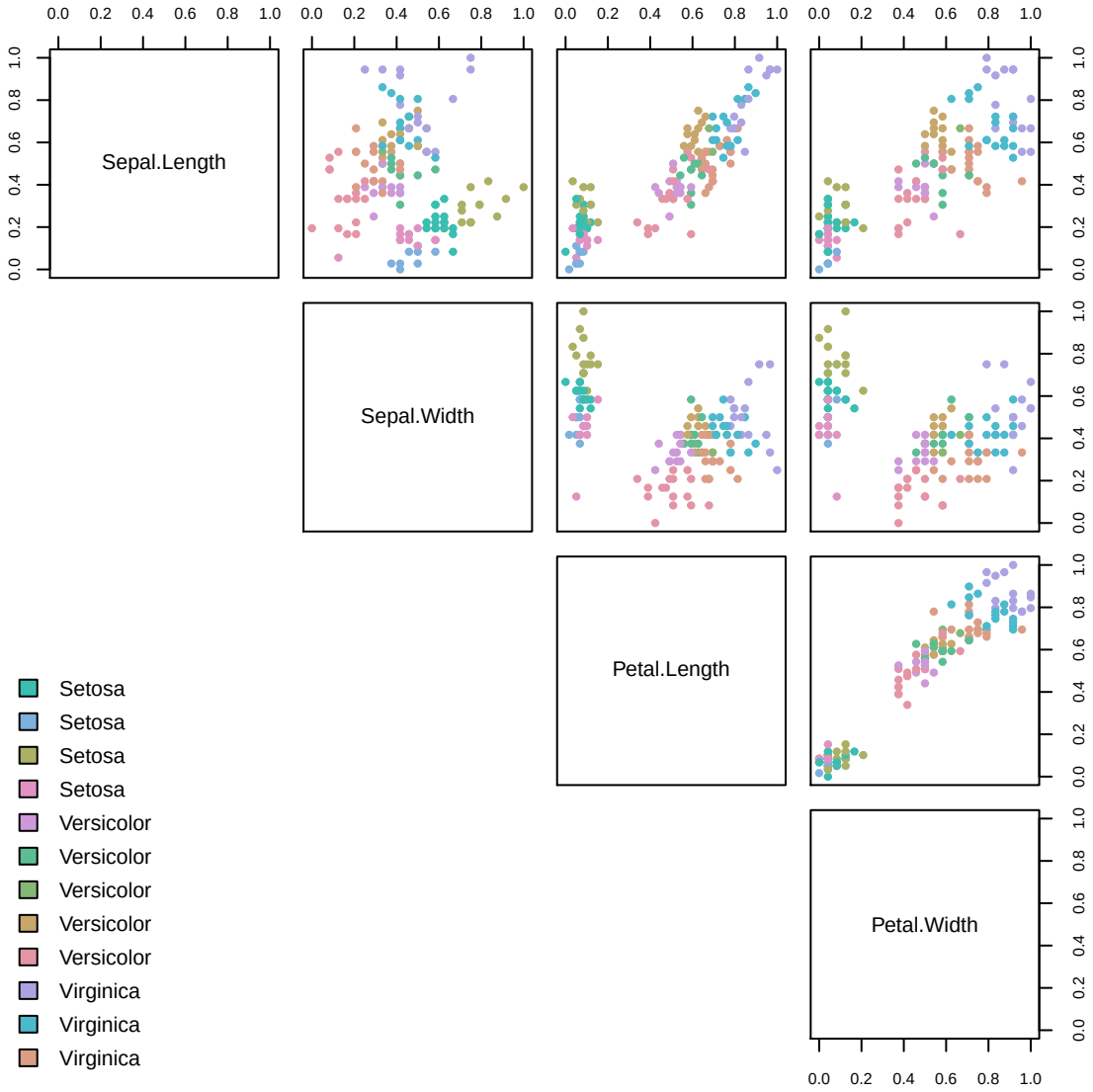

Рис. 5.9. Розкластерований датасет «Іриси Фішера» при $m = 12$, $\beta = 4$ (Точність кластерування $\approx 96\%$)

	$\beta = 2$	$\beta = 3$	$\beta = 4$	$\beta = 5$
avg	0.9369168	0.9448829	0.9383179	0.9403416
min	0.8847819	0.8953380	0.8787879	0.9069805
max	0.9731262	0.9754579	0.9918301	0.9762515

Таблиця 5.6

Точність кластерування при $m = 14$

самостійно, особливо у випадку, коли вузли системи реалізують однаковий метод кластерування.

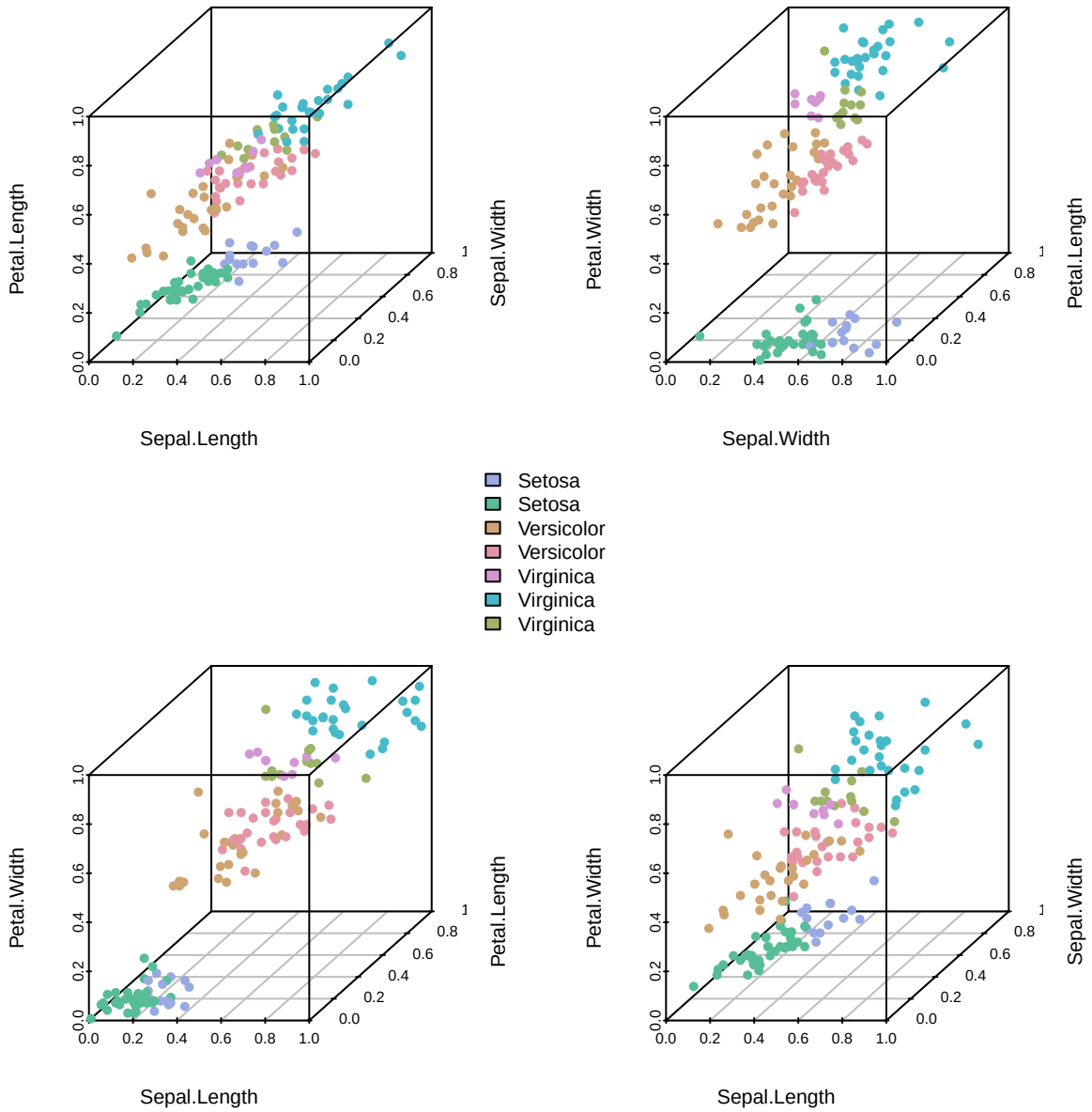

Рис. 5.10. Розкластерований датасет «Іриси Фішера» при $m = 7$, $\beta = 5$ (Точність кластерування $\approx 93\%$)

Варто зауважити, що у цьому випадку для визначення локально оптимального розбиття доцільно використовувати модифіковані індекси валідності, чи такі, що не залжать від відстані цетрів кластері, наприклад ті, що

ґрунтуються на щільності (density-based).

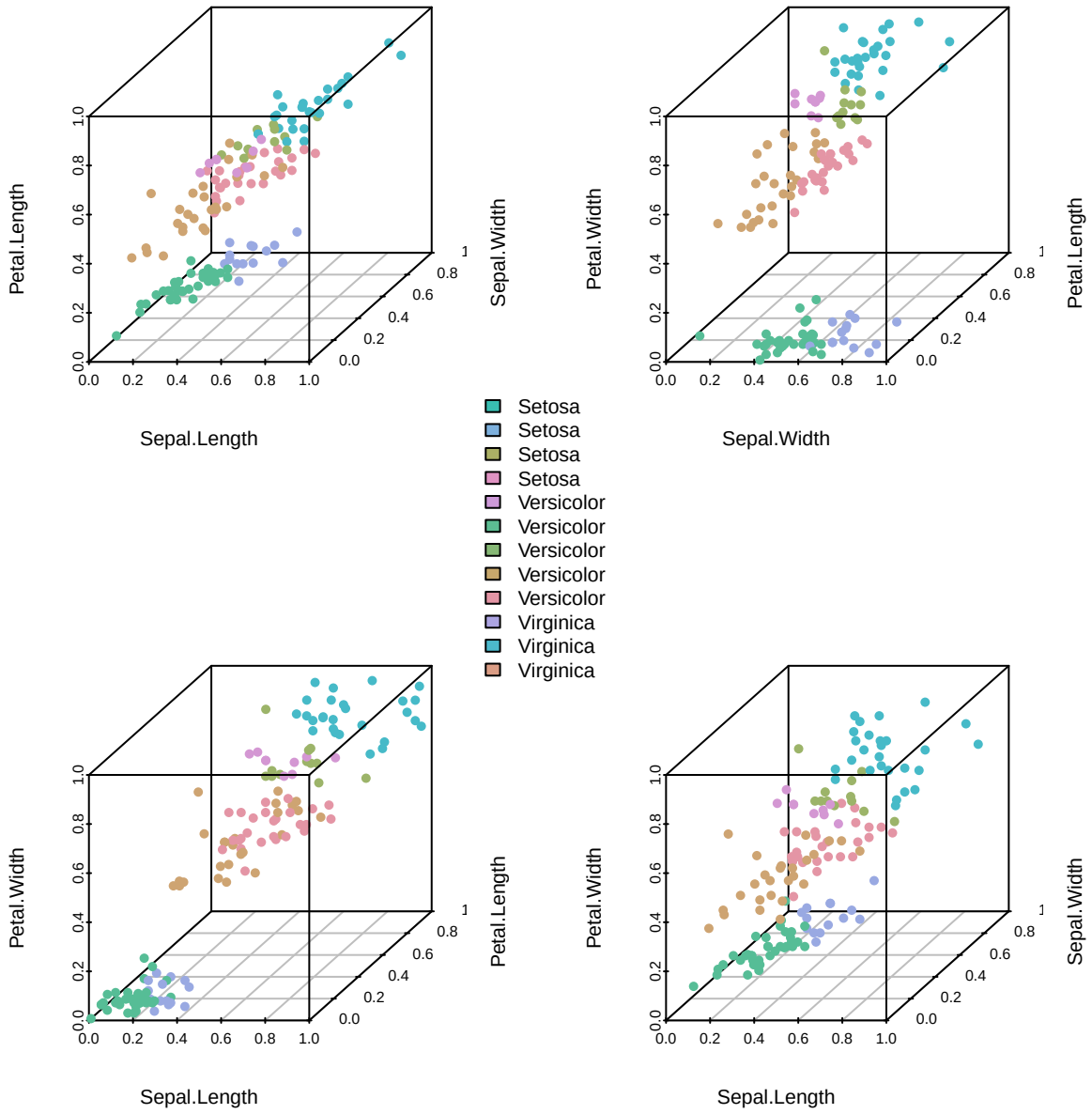

Рис. 5.11. Розкластерований датасет «Іриси Фішера» при $m = 12$, $\beta = 4$ (Точність кластерування $\approx 96\%$)

5.1.3. Придумати назву 3. Наступну серію експериментів було проведено на датасеті «Знання студентів про електричні машини постійного струму».

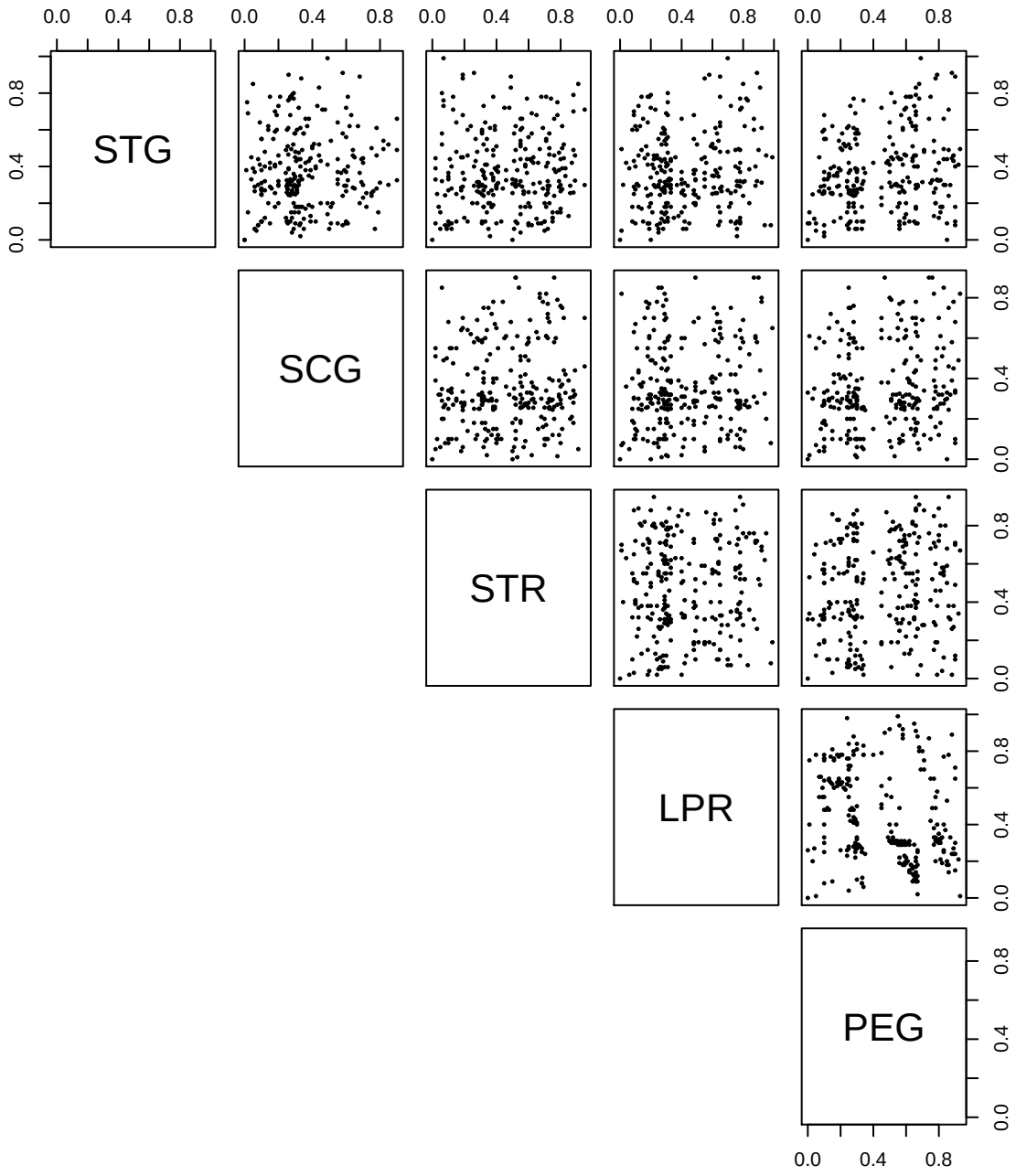

Рис. 5.12. Датасет «Знання студентів про електричні машини постійного струму»

Цей датасет було додано до UCI репозиторію у 2013 році, він містить 403 паттерни, кожен з п'ятьма атрибутами:

1. STG: кількість часу, що його витратив(витратила) студент(ка) на навчання цільового матеріалу,
2. SCG: Кількість повторюваннь навчання цільового матеріалу студен-

том(студенткою),

3. STR: Кількість часу, що його використав(використала) студент(ка) на навчання матеріалу, пов'язаного з цільовим матеріалом,
4. LPR: Оцінка, що її отримав(отримала) студент(ка) на іспиті з предмету, пов'язаного з цільовим предметом,
5. PEG: Оцінка, що її отримав(отримала) студент(ка) на іспиті з цільового предмету,

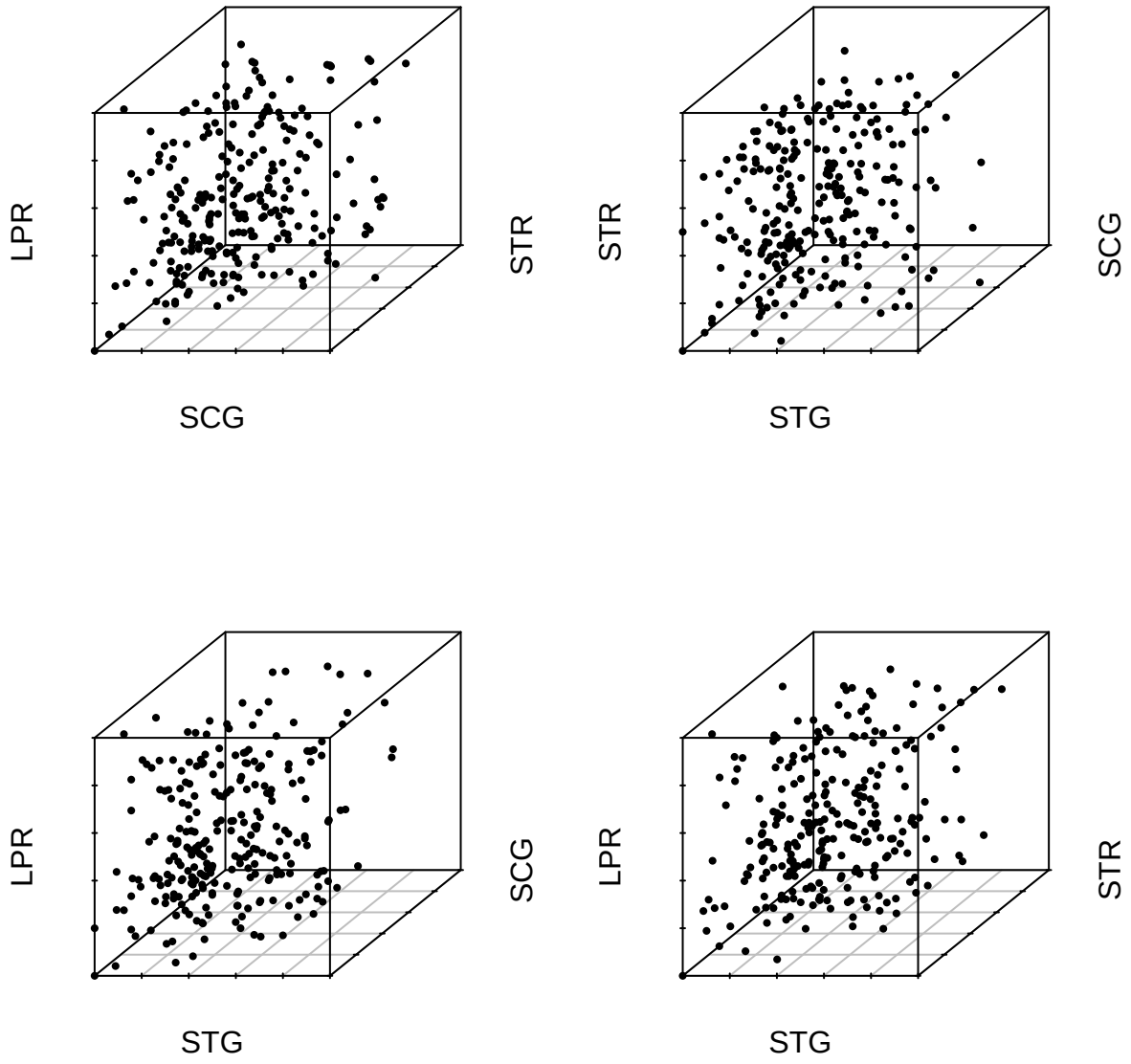

Рис. 5.13. Датасет «Знання студентів про електричні машини постійного струму»

Попарні графіки атрибутів наведено на рис. 5.12 та у тримірному просторі на рис. 5.13.

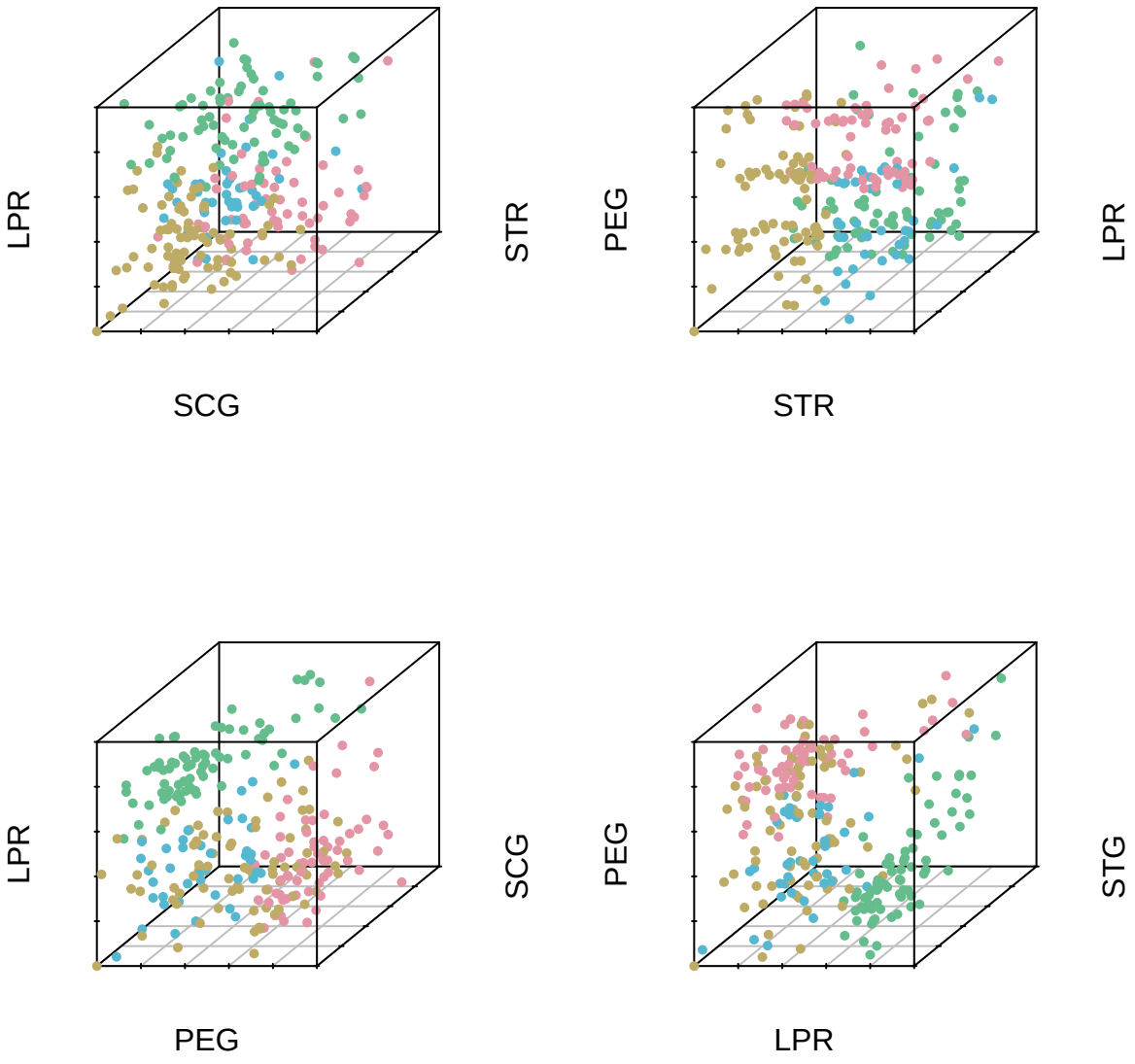

Рис. 5.14. Датасет «Знання студентів про електричні машини постійного струму»

Для цього експерименту нейрони-узагальнювачі керувалися рекурентним Ксі-Бені Індексом при визначанні локально-оптимального нейрона (з найлішпшим параметром фаззіфікації) та каскаду (з оптимальною кількістю кластерів).

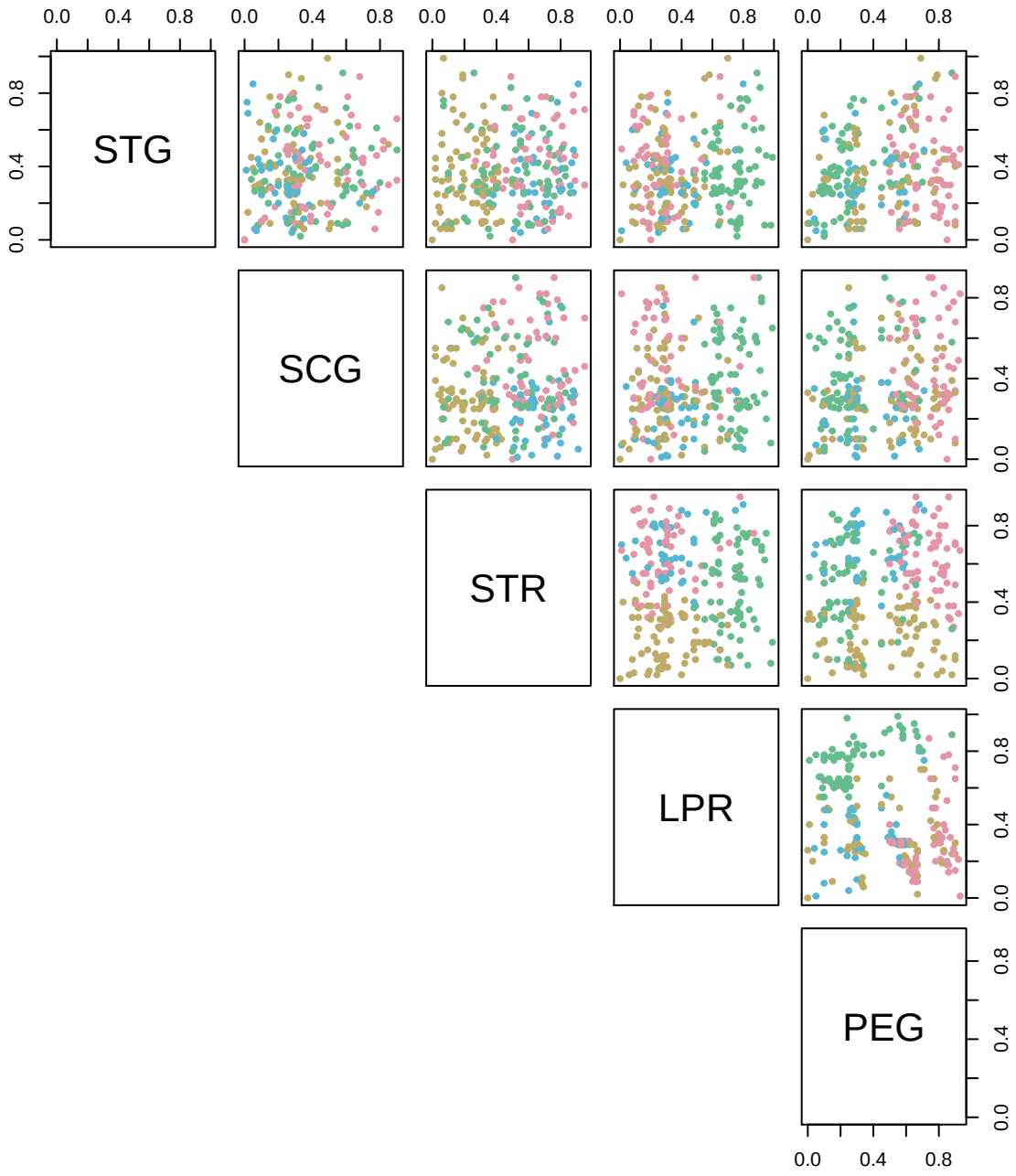

Рис. 5.15. Датасет «Знання студентів про електричні машини постійного струму»

Оптимальне розбиття, що його наведено на рис.5.14 та на рис.5.15, надав другий нейрон третього каскаду з $m = 4$, $\beta = 2$ з коефіцієнтом Ксі-Бені 0.38155.

5.1.4. Розв'язування практичних задач за допомогою розробленої самонавчанної гібридної каскадної системи, що еволюціонує. Проблема здорового харчування - одна з найактуальніших у наші дні. Повноцінне харчування передбачає споживання достатньої кількості білків, жирів, вуглеводів, вітамінів, макро- і мікроелементів для нормального функціонування організму в цілому. Багато хвороб шлунково-кишкового тракту «молодіють» - це гастрити, виразкова хвороба шлунка і різні порушення обміну речовин. Фізичне здоров'я, стан імунітету, довголіття, психічна гармонія - все це безпосередньо пов'язано з проблемою здорового харчування людини. Для студентів проблема харчування стоїть особливо гостро. У зв'язку з браком часу у студентів немає можливості дотримуватися правильного режиму прийомів їжі в кількості 3-4 разів. Також характерний в основному сидячий спосіб життя - гіподинамія. У поєднанні з незбалансованим раціоном харчування це згубно впливає на організм і його стан. Звісно, вирішення проблеми здорового харчування потребує комплексного підходу, проте інфомованість - невід'ємна складова правильного підбору раціону здорового харчування. Насьогодні нескладно знайти інформацію щодо рекомендованої денної кількості калорій, білків, жирів та вуглеводів, проте важко дати оцінку конкретому прийому їжі, наприклад, придбаному у їдальні, де немає етикеток з такою інформацією. Мобільний додаток «Spoon app» може стати у нагоді, коли користувач прагне бути проінформованим щодо поживності конкретної трапези, роблячи аналіз світлини тарілки з їжею. Вхідними даними мобільного додатку є світлина, що її користувач має зробити таким чином, аби тарілка знаходилася у центрі, а також тип тарілки (звичайна, глибока, дуже глибока) аби на виході додаток мав обґрунтовану кількість калорій та поживність порції, що була зображена на світлині.

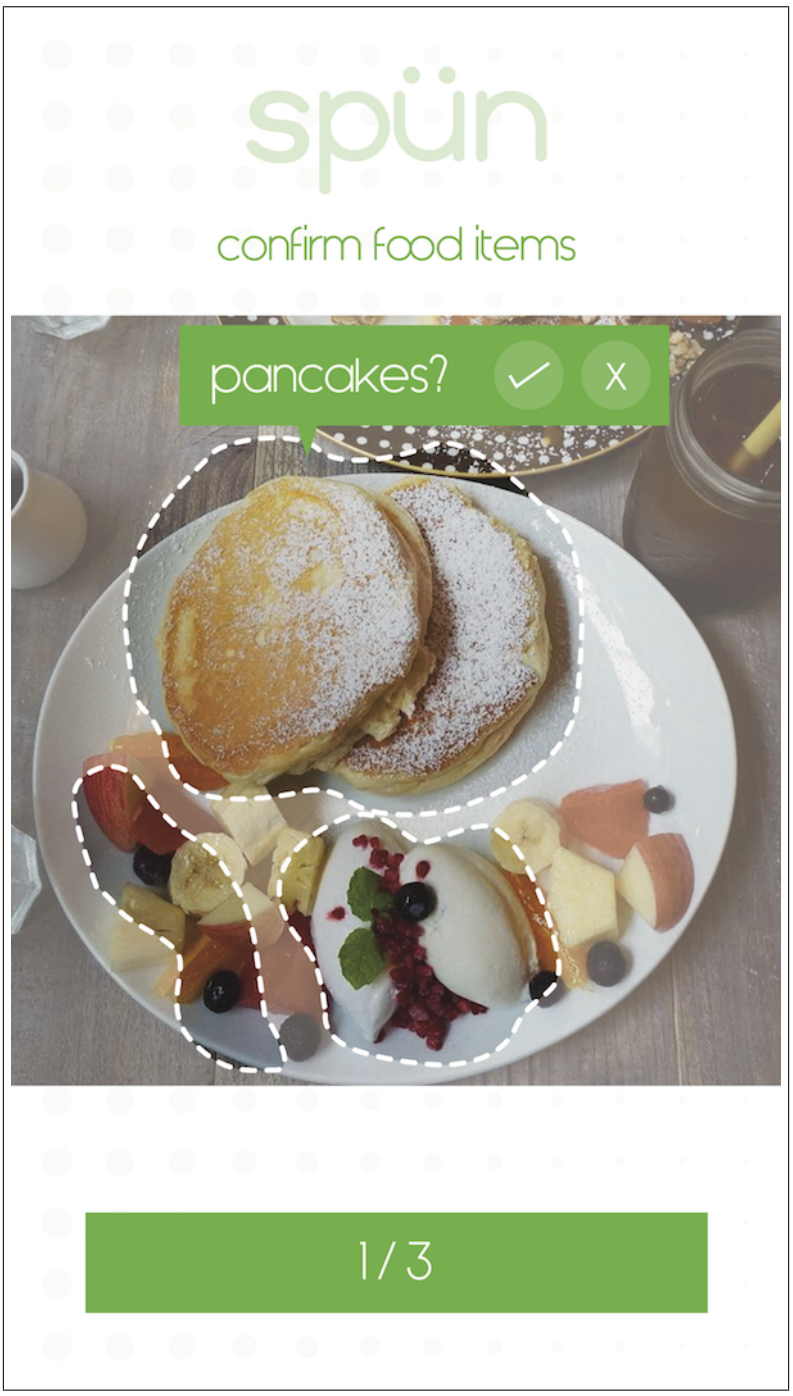

Рис. 5.16. Другий етап аналізу світлини з прийомом їжі мобільним додатком Spoon App (навчаняя з іпленням)

Аналіз світлини можна розбити на декілька етапів:

1. кластерування даних зображених на світлині (відбувається на стороні клієнту)
2. ідентифікація окремих складових прийому їжі: класифікація кожного зображення, після розбиття світлини на кластери на першому етапі

(відбувається на серверній стороні), визначення типу продукту за допомогою бази даних, що знаходиться на сервері, та подальше визначанні кількості калорій, співвідношення білкі, жирів та вуглеводів.

Рис. 5.17. Перший етап аналізу світлини з прийомом їжі мобільним додатком Spoon App (кластерування за умови невизначенності щодо кількості кластерів)

Другий етап у певному сенсі є навчанням з підкріпленням, адже користу-

вачеві пропонується підтвердити чи скорегувати кінцевий результат класифікації, як зображено на рис. 5.16. Проте більш цікавим нам видається саме перший етап аналізу світлини, і пропонована гібридна самонавчана система використовується саме на цьому етапі, адже вона краще від інших існуючих систем задовольняє умовам, що їх було висунуто на етапі формування технічних вимог до програмного забеспечення (Software Requirements Specifications):

— кластерування має проходити за умови невизначенності щодо кількості кластерів,

— оскільки кластерування відбувається на стороні кліенту, важливо мінімізувати обслювальну складність алгоритму, а отже перевага надається методам послідовного кластерування.

По завершенні аналізу на першому етапі мобільний додаток попронує розбиття світлини на m кластерів, як показано на рис. 5.17. Варто зауважити, що межі кластерів, що їх визначила самонавчана система, дещо відрізняються від тих, що зображені на рис. 5.17, для того, щоб користувачеві було зручніше візуально сприймати розбиття світлини на кластери, пункирні лініїї, що зображуть межі кластерів, на декілька міліметрів віддалені від меж дійсних кластерів, проте на сервер для подальшої класифікації відправляється світлина з розбиттям, що запропонувала система. На цьому етапі користувач може скорегувати розбиттся, перетягнувши пунктирну лінію меж кластерів, чи зовсім видалити пропонований кластер. Хоча навчання з підкріпленням у прямому сенсі, відбувається лише на етапі классифікації (база даних на сервері оновлюються, коли користувач корегує результат класифікації), а на цьому етапі маємо саме навчання без учителя, це все ж таки дає змогу у певному сенсі дати оцінку кластеруванню системи: вважаємо кластерування успішним, якщо користувач не робив жодних змін до пропонованого розбиття, та неуспішним, коли розбиття було скореговане. Після бета тестування мобільного додатку маємо наступні результати:

Цікавим видається перебіг подій, коли користувач видалив один кластер,

кількість та межі кластеров залишилися незмінними	608
межі кластеров не було дещо змінено користувачем, проте кількість кластерів залишилась незмінною	61
користувач змінив кількість кластерів та межі пропонованих кластерів	52
користувач видалив кластер(и), межі пропонованих кластерів, що лишилися, незмінні	29

Таблиця 5.7

Результати бета тестування першого етапу (кластерування за умови невизначенності щодо кількості кластерів) аналізу світлини мобільним додатком «Spoon App»

проте залишим межі інших кластерів незмінними, у такому випадку, як зазначалося вище, вважається що кластерування не було успішним. Проте подальший аналіз показав, що у 90% таких випадків користувач залишив на тарілці неїстівний предмет (виделку, ложку тощо), і хоча система вірно відвела йому окремий кластер, не має сенсу класифікувати його та визначати калорійність цього предмету, тому логічно, що користувач видалив його на цьому етапі. У 10% випадків, як показав аналіз світлин з початковим кластеруванням, що запропонувала самонавчана система, та світлин після корегування користувачем, користувач свідомо видаляв складову прийому їжі, зазвичай найменш корисну (тітечко, шоколад тощо). Тому видається доцільним ці 4% світлин також віднести до таких, що були вірно розкластеровані системою (яка, проте, не бере до уваги людський фактор).

5.2. Моделювання гібридної каскадної нейро-фаззі мережі з оптимізацією пулу нейронів

Для проведення наступних чисельних експериментів (за вийнятком експериментів з самонавчанною гібридною системою, що еволюціонує) було обрано такі критерії оцінки:

— MSE (Mean Squared Error, середньоквадратична похибка),

— SMAPE (Symmetric Mean Absolute Percentage Error, симетрична абсолютна процентна похибка)

Середньквадратична похибка похибка (MSE) обчислюється наступним чином следующим образом:

5.2.1. Моделювання розширенного нейро-фаззі нейрона. Для цього експерименту датасет було сгенеровано за формулою

$$\sin\big(k + \sin(2k)\big) \text{ для } k \in [1, 600], \tag{5.1}$$

(фазовий портрет наведено на рис. 5.18).

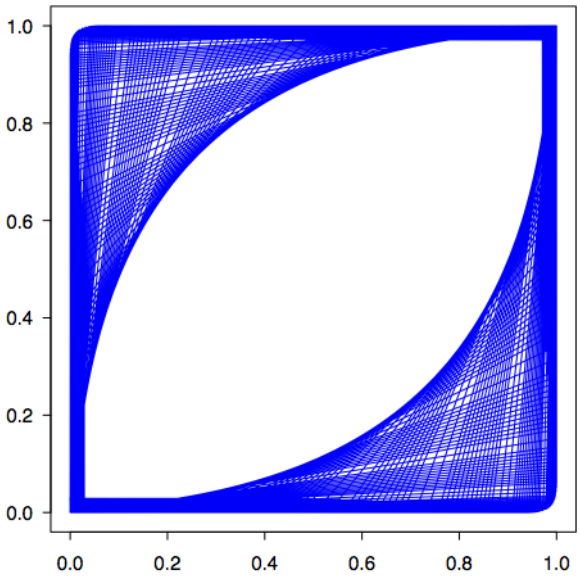

Рис. 5.18. Фазовий портрет штучно сгенерованого датасету

Результати експерименту наведено у таблиці нижче, а також проілюстровано залежність точності прогнозу від порядку висновування (рис. 5.19) та кількості фунцій належності (рис. 5.20).

Inference order 0		
Membership Functions	RMSE	SMAPE
2	0.0743020867216913	3.39992930699002
3	0.07593154138757	7.7.2130842903092
4	0.0816015155371995	3.74330006151476
5	0.0899940897089563	7.28423693259539
6	0.098548606433121	4.97414996889677
Inference order 1		
Membership Functions	RMSE	SMAPE
2	0.088863162670838	2.86925982823229
3	0.107342623622113	2.61031587883906
4	0.114302161682684	2.43845462677015
5	0.121556611017793	2.21187746150696
6	0.129922583600673	2.3504927587044
Inference order 2		
Membership Functions	RMSE	SMAPE
2	0.0949715863284214	3.10116527241627
3	0.127570672613453	3.21603084937958
4	0.121809301731276	2.49191939755954
5	0.134120568445479	2.74555892364861
6	0.146724724859333	2.39716069975922
Inference order 3		
Membership Functions	RMSE	SMAPE
2	0.100666657170784	3.26456771933306
3	0.130173960343431	2.12509843555904
4	0.154580667425542	2.17237774290269
5	0.139334351692536	3.04236918521622
6	0.150497843534837	2.43818195954604

Таблиця 5.8

Точність прогнозу розширенного нео фаззі нейрону на штучно сгенерованому датасеті

Отже, як видно з таблиці TODO table та на рис. 5.21, можна зробити висновок, що точність прогнозу розширеного нео-фаззі нейрону вища від точності звичайного нео-фаззі нейрону (ENFN з нульовим порядком виснову-

ванням).

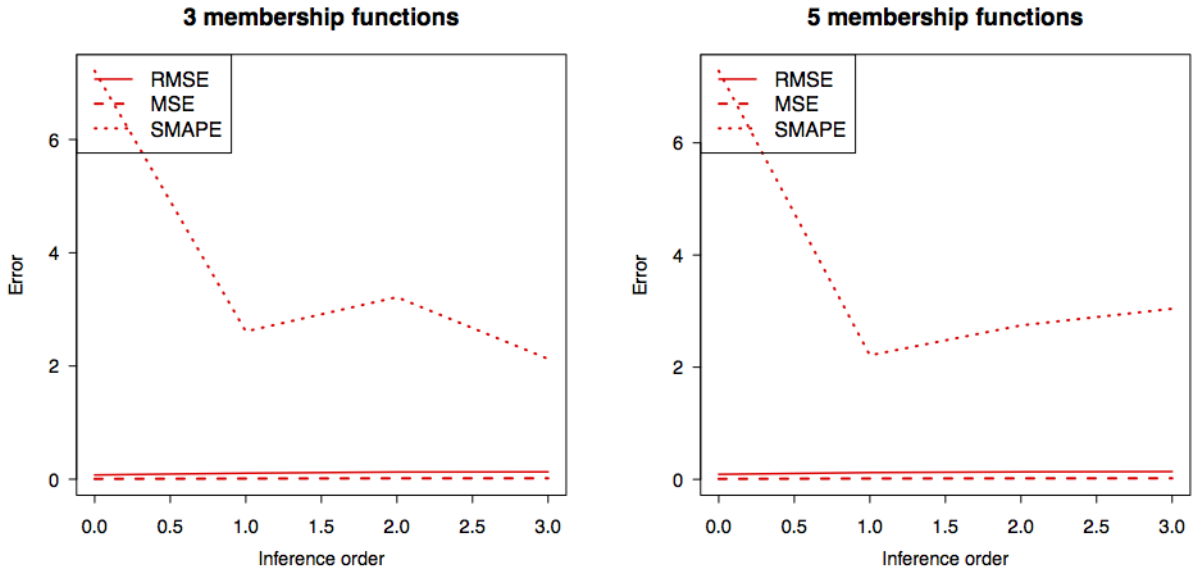

Рис. 5.19. Похибка прогнозу розширенного нео-фаззі нейрону від порядку висновування (для трьох та п'яти функцій належності)

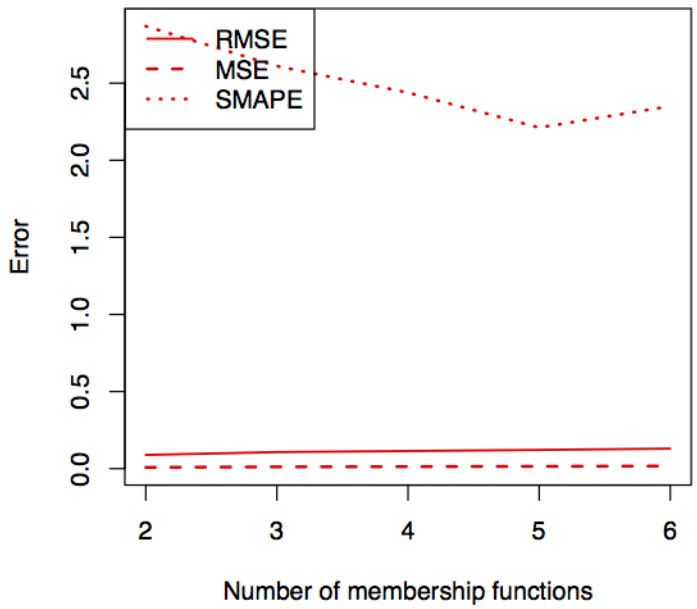

Рис. 5.20. Похибка прогнозу розширенного нео-фаззі нейрону від кількості фунцій належності (порядок висновування - 2)

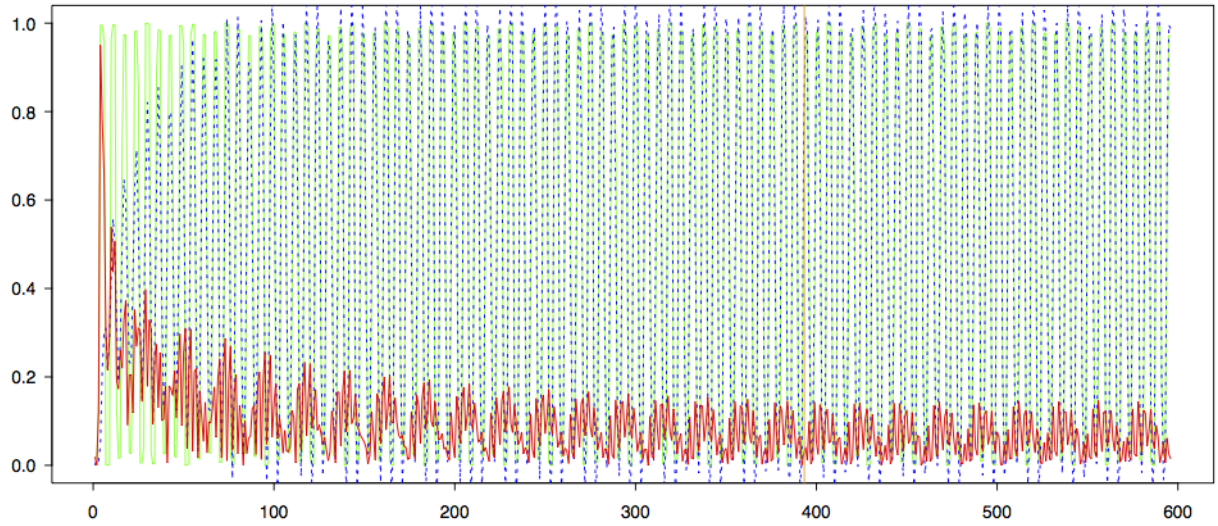

Рис. 5.21. Прогноз розширенного нео-фаззі нейрону з 3 трикутними функціями належності, що реалізує нечітке висновування 3-ого порядку (зелена лінія – шуканий сигнал, синя пунктирна лінія – проноз розширенного нео-фаззі нейрону, червона лінія – похибка; жовтобагряна вертикальна лінія позначає закінчення тренувальної частини датасету)

5.3. Моделювання гібридної каскадної нейро-фаззі мережі на розшиерниї нео-фаззі нейронах з оптимізацією пулу нейронів

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] Abonyi Janos, Feil Balazs. Cluster Analysis for Data Mining and System Identification. — Birkhäuser Basel, 2007. — ISBN: 3764379871.

[2] Adaptive Clustering for Multiple Evolving Streams / Bi-Ru Dai, Jen-Wei Huang, Mi-Yen Yeh, Ming-Syan Chen // IEEE Trans. Knowl. Data Eng. — 2006. — Vol. 18, no. 9. — P. 1166–1180. — URL: http://dx.doi.org/10.1109/TKDE.2006.137; http://doi.ieeecomputersociety.org/10.1109/TKDE.2006.137.

[3] Angelov Plamen P, Filev Dimitar P. An approach to online identification of Takagi-Sugeno fuzzy models // Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on. — 2004. — Vol. 34, no. 1. — P. 484–498.

[4] Angelov Plamen P., Lughofer Edwin. Data-driven evolving fuzzy systems using eTS and FLEXFIS: comparative analysis // Int. J. General Systems. — 2008. — Vol. 37, no. 1. — P. 45–67. — URL: http://dx.doi.org/10.1080/03081070701500059.

[5] Arrow Kenneth Joseph, Hurwicz Leonid, Uzawa Hirofumi. Studies in linear and non-linear programming. Stanford mathematical studies in the social sciences. — Stanford,Ca : Stanford university press, 1972. — ISBN: 0-8047-0562-3. — URL: http://opac.inria.fr/record=b1099343.

[6] Beringer Jürgen, Hüllermeier Eyke. Online clustering of parallel data streams // Data Knowl. Eng. — 2006. — Vol. 58, no. 2. — P. 180–204. — URL: http://dx.doi.org/10.1016/j.datak.2005.05.009.

[7] Beringer Jürgen, Hüllermeier Eyke. Online clustering of parallel data streams // Data Knowl. Eng. — 2006. — Vol. 58, no. 2. — P. 180–204. — URL: http://dx.doi.org/10.1016/j.datak.2005.05.009.

[8] Bezdek J.C. Pattern recognition with fuzzy objective function algorithms. — Kluwer Academic Publishers, 1981.

[9] Bodyanskiy Yevgeniy. Computational Intelligence Techniques for Data Analysis // Leipziger Informatik-Tage / Ed. by Klaus P. Jantke, Klaus-Peter Fähnrich, Wolfgang S. Wittig. — Vol. 72 of LNI. — GI, 2005. — P. 15–36. — URL: http://subs.emis.de/LNI/Proceedings/Proceedings72/article3735.html.

[10] Bodyanskiy Ye., Gorshkov Ye., Kolodyazhniy V. New recursive learning algorithms for fuzzy Kohonen clustering network // Proc. 17th Int. Workshop on Nonlinear Dynamics of Electronic Systems. — 2009. — P. 58–61.

[11] Bodyanskiy Yevgeniy, Grimm Paul, Teslenko Nataliya. Evolving cascaded neural network based on multidimensional Epanechnikov's kernels and its learning algorithm // Int. J. Information Technologies and Knowledge. — 2011. — Vol. 5, no. 1. — P. 25–30.

[12] Bodyanskiy Yevgeniy, Kharchenko Oleksandra, Vynokurova Olena. Hybrid cascade neural network based on wavelet-neuron // Information Theories and Application. — 2011. — Vol. 18, no. 4. — P. 335–343.

[13] Bodyanskiy Yevgeniy, Kokshenev Illya, Kolodyazhniy Vitaliy. An adaptive learning algorithm for a neo fuzzy neuron. // EUSFLAT Conf. — 2003. — P. 375–379.

[14] Bodyanskiy Ye., Kolodyazhniy V., Stephan A. Recursive fuzzy clustering algorithms // Proc. 10th East West Fuzzy Colloquium. — 2002. — P. 276–283.

[15] Bodyanskiy Yevgeniy, Viktorov Yevgen. The cascaded neo-fuzzy architecture using cubic-spline activation functions // Inf Theor Appl. — 2009. — Vol. 16, no. 3. — P. 245–259.

[16] Bodyanskiy Yevgeniy, Viktorov Yevgen. 110 9 – Intelligent Processing THE CASCADE NEO-FUZZY ARCHITECTURE AND ITS ONLINE LEARNING ALGORITHM. — 2013. — URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.386.7492.

[17] Bodyanskiy Yevgeniy, Viktorov Yevgen, Pliss Iryna. International Book Series "Information Science and Computing " 25 THE CASCADE GROWING NEURAL NETWORK USING

QUADRATIC NEURONS AND ITS LEARNING ALGORITHMS FOR ON-LINE INFORMATION PROCESSING. — 2013. — URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.386.9313.

[18] Bodyanskiy Ye V, Pliss IP, Solovyova T. Multistep optimal predictors of multidimensional non-stationary stochastic processes // Doklady AN USSR, series A. — 1986. — Vol. 12. — P. 47–49.

[19] Chen L H, Chang S. An adaptive learning algorithm for principal component analysis. // IEEE transactions on neural networks / a publication of the IEEE Neural Networks Council. — 1995. — Vol. 6, no. 5. — P. 1255–63.

[20] Chen Y., Tu L. Density-based clustering for real-time stream data // Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining. — 2007. — P. 133–142.

[21] Chung Fu Lai, Lee Tong. Fuzzy Competitive Learning // Neural Netw. — 1994. — Vol. 7, no. 3. — P. 539–551. — URL: http://dx.doi.org/10.1016/0893-6080(94)90111-2.

[22] Chung Fu-Lai, Lee Tong. Fuzzy competitive learning // Neural Networks. — 1994. — Vol. 7, no. 3. — P. 539–551. — URL: http://dx.doi.org/10.1016/0893-6080(94)90111-2.

[23] Cichocki A., Unbehauen R. Neural networks for optimization and signal processing // Journal of Signal Processing. — 1998. — Vol. 2. — P. 62–63.

[24] Crespo Fernando, Weber Richard. A methodology for dynamic data mining based on fuzzy clustering // Fuzzy Sets and Systems. — 2005. — Vol. 150, no. 2. — P. 267–284. — URL: http://dx.doi.org/10.1016/j.fss.2004.03.028.

[25] Cypkin Ja.Z., Kaplinskii A. I., Krasnenker A. S. Fundamentals of the theory of learning systems // Nauka. — 1970. — P. 252.

[26] Du K.L., Swamy M.N.S. Neural Networks and Statistical Learning. SpringerLink : Bücher. — Springer, 2013. — ISBN: 9781447155713. — URL: https://books.google.com.ua/books?id=wzK8BAAAQBAJ.

[27] Fahlman Scott E., Lebiere Christian. The cascade-correlation learning architecture // Advances in Neural Information Processing Systems 2. — Mor-

gan Kaufmann, 1990. — P. 524–532.

[28] Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition / Frank Höppner, Frank Klawonn, Rudolf Kruse, Thomas Runkler. — 1. Auflage edition. — John Wiley & Sons, 1999. — ISBN: 0471988642.

[29] Fuzzy models and algorithms for pattern recognition and image processing / James C. Bezdek, James Keller, Raghu Krisnapuram, Nikhil R. Pal. The Handbooks of fuzzy sets series. — New York : Springer, 2005. — ISBN: 0-387-24515-4. — URL: http://opac.inria.fr/record=b1102792.

[30] Gan Guojun, Ma Chaoqun, Wu Jianhong. Data Clustering: Theory, Algorithms, and Applications (ASA-SIAM Series on Statistics and Applied Probability). — SIAM, 2007. — ISBN: 0898716233.

[31] Han H., Qiao J. A self-organizing fuzzy neural network based on a growing-and-pruning algorithm // Fuzzy Systems, IEEE Transactions on. — 2010. — Vol. 18, no. 6. — P. 1129–1143.

[32] Han J, Kamber M. Mining Stream, Time-Series and Sequence Data // Data Mining: Concepts and Techniques. — 2006. — P. 467–489.

[33] Haykin S. Neural Networks: a Comprehensive Foundation. — New York, NY : Macmillan, 1994.

[34] Bodyanskiy Yevgeniy, Dolotov Artem, Pliss Iryna, Viktorov Yevgen. International Book Series "Information Science and Computing" 13 THE CASCADE ORTHOGONAL NEURAL NETWORK. — 2008.

[35] Jang Jyh-Shing Roger, Sun Chuen-Tsai, Mizutani Eiji. Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence. MATLAB curriculum series. — pub-PH:adr : Prentice-Hall, 1997. — P. xxvi + 614. — ISBN: 0-13-261066-3.

[36] Kaczmarz S. Approximate solution of systems of linear equations // Int. J. Control,. — 1993. — Vol. vol. 53. — P. 1269–1271.

[37] Kasabov Nikola. Evolving connectionist systems - the knowledge engineering approach (2. ed.). — Springer, 2007. — P. I–XXI, 1–457. — URL:

http://dx.doi.org/10.1007/978-1-84628-347-5.

[38] Kasabov Nikola K, Song Qun. DENFIS: dynamic evolving neural-fuzzy inference system and its application for time-series prediction // Fuzzy Systems, IEEE Transactions on. — 2002. — Vol. 10, no. 2. — P. 144–154.

[39] Kashyap R. L., Blaydon Colin C. Estimation of probability density and distribution functions // IEEE Transactions on Information Theory. — 1968. — Vol. 14, no. 4. — P. 549–556. — URL: http://dx.doi.org/10.1109/TIT.1968.1054184.

[40] Kohonen Teuvo. Self-organizing maps. — Springer Science & Business Media, 2001. — Vol. 30.

[41] Kolodyazhniy V, Bodyanskiy Ye. Cascaded multiresolution spline-based fuzzy neural network // Proc. Int. Symp. on Evolving Intelligent Systems. — 2010. — P. 26–29.

[42] Kruschke John K, Movellan Javier R. Benefits of gain: Speeded learning and minimal hidden layers in back-propagation networks // Systems, Man and Cybernetics, IEEE Transactions on. — 1991. — Vol. 21, no. 1. — P. 273–280.

[43] Leng Gang, Prasad Girijesh, McGinnity T. Martin. An on-line algorithm for creating self-organizing fuzzy neural networks // Neural Networks. — 2004. — Vol. 17, no. 10. — P. 1477–1493. — URL: http://dx.doi.org/10.1016/j.neunet.2004.07.009.

[44] Liu Ying-Ho. Stream mining on univariate uncertain data // Appl. Intell. — 2013. — Vol. 39, no. 2. — P. 315–344. — URL: http://dx.doi.org/10.1007/s10489-012-0415-3.

[45] Ljung Lennart. System Identification: Theory for the User. — Prentice-Hall, Inc., 1999.

[46] Lughofer Edwin. FLEXFIS: A Robust Incremental Learning Approach for Evolving Takagi-Sugeno Fuzzy Models // IEEE T. Fuzzy Systems. — 2008. — Vol. 16, no. 6. — P. 1393–1410. — URL: http://dx.doi.org/10.1109/TFUZZ.2008.925908.

[47] Lughofer Edwin. Evolving Fuzzy Systems - Methodologies, Advanced

Concepts and Applications. — Springer, 2011. — Vol. 266 of Studies in Fuzziness and Soft Computing. — P. 1–410. — ISBN: 978-3-642-18086-6. — URL: http://dx.doi.org/10.1007/978-3-642-18087-3; http://dx.doi.org/10.1007/978-3-642-18087-3.

[48] Miki T., (Japan) T. Yamakawa. Analog Implementation of Neo-Fuzzy Neuron and Its On-board Learning. — WSEAS, 1999. — URL: http://www.worldses.org/online/.

[49] Nechyba Michael C., Xu Yangsheng. Cascade Neural Networks with Node-Decoupled Extended Kalman Filtering. — 1997.

[50] Oliveira Jose Valente de, Pedrycz Witold. Advances in Fuzzy Clustering and Its Applications. — New York, NY, USA : John Wiley & Sons, Inc., 2007. — ISBN: 0470027606.

[51] Otto Peter, Bodyanskiy Yevgeniy, Kolodyazhniy Vitaliy. A new learning algorithm for a forecasting neuro-fuzzy network // Integrated Computer-Aided Engineering. — 2003. — Vol. 10, no. 4. — P. 399–409. — URL: http://content.iospress.com/articles/integrated-computer-aided-engineering/ica00163.

[52] Pakhira Malay K, Bandyopadhyay Sanghamitra, Maulik Ujjwal. Validity index for crisp and fuzzy clusters // Pattern recognition. — 2004. — Vol. 37, no. 3. — P. 487–501.

[53] Park Dong C., Dagher Issam. Gradient Based Fuzzy c_means (GBFCM) Algorithm // IEEE International Conference on Neural Networks (ICNN'94). — Vol. III. — Orlando, FL : IEEE, 1994. — jun. — P. 1626–1631. — Intelligent Computing Research Lab, Department of ECE, FL.

[54] Pedrycz Witold, Rai Partab. Collaborative clustering with the use of Fuzzy C-Means and its quantification // Fuzzy Sets and Systems. — 2008. — Vol. 159, no. 18. — P. 2399–2427. — URL: http://dx.doi.org/10.1016/j.fss.2007.12.030.

[55] Prechelt L. Investigation of the CasCor Family of Learning Algorithms // Neural Networks. — 1997. — Vol. 10, no. 5. — P. 885–896.

[56] Roubens Marc. Fuzzy clustering algorithms and their cluster validity // European Journal of Operational Research. — 1982. — Vol. 10. — P. 294–301.

[57] Silva Bruno, Marques Nuno. Neural Network-based Framework for Data Stream Mining. // STAIRS. — 2012. — P. 294–305.

[58] Silva F. M., Almeida L. B. Speeding up back-propagation // Advanced Neural Computers / Ed. by R. Eckmiller. — Amsterdam : Elsevier, 1990. — P. 151–158.

[59] Smoothly distributed fuzzy C-means: a new self-organizing map / R. D. Pascual Marqui, A. D. Pascual Montano, K. Kochi, J. M. Carazo // Pattern Recognition. — 2001. — Vol. 34, no. 12. — P. 2395–2402. — URL: http://www.sciencedirect.com/science/article/B6V14-43W07HY-B/2/8a783959537578469a5357887f06327b.

[60] Tsao E. C. K., Bezdek J. C., Pal N. R. Fuzzy Kohonen clustering networks // Pattern Recognition. — 1994. — Vol. 27, no. 5. — P. 757–764. — URL: http://www.sciencedirect.com/science/article/B6V14-48MPPXD-1YX/2/cf0e26b7498b3a4f4b7698975495f7f9.

[61] Veitch A. C., Holmes G. A Modified Quickprop Algorithm // Neural Computation. — 1991. — Vol. 3, no. 3. — P. 310–311.

[62] Vorobyov Sergiy A., Cichocki Andrzej, Bodyanskiy Yevgeniy V. Adaptive Noise Cancellation For Multi-Sensory Signals. — 2001. — URL: http://citeseer.ist.psu.edu/404753.html.

[63] Vuorimaa Petri. Fuzzy Self–Organizing Map // Fuzzy Sets and Systems. — 1994. — Vol. 66. — P. 223–231.

[64] Vuorimaa P. Use of the Fuzzy Self-Organizing Map in pattern recognition // Proceedings of the Third IEEE Conference on Fuzzy Systems. IEEE World Congress on Computational Intelligence / Signal Process. Lab. , Tampere Univ. of Technol. , Finland. — Vol. 2. — New York, NY, USA : IEEE, 1994. — P. 798–801.

[65] Wang Li-Xin. Adaptive fuzzy systems and control - design and stability analysis. — Prentice Hall, 1994. — P. I–XVII, 1–232. — ISBN: 978-0-13-099631-2.

[66] Wang Li-Xin, Mendel Jerry M. Fuzzy basis functions, universal approximation, and orthogonal least-squares learning // IEEE Transactions

on Neural Networks. — 1992. — Vol. 3, no. 5. — P. 807–814. — URL: http://dx.doi.org/10.1109/72.159070.

[67] Wang W, Zhang Y. On fuzzy cluster validity indices // Fuzzy Sets and Systems. — 2007. — oct. — Vol. 158, no. 19. — P. 2095–2117.

[68] Widrow B., Hoff M. E. Adaptive Switching Circuits // Proceedings WESCON. — 1960. — P. 96–104.

[69] Wolf M. ANFIS: Adaptive-Network-Based Fuzzy Inference System. — 1998.

[70] Wu Yi. Network Big Data: A Literature Survey on Stream Data Mining. — 2014. — Vol. 9. — P. 2427–2434. — URL: http://ojs.academypublisher.com/index.php/jsw/article/view/12381.

[71] Xie Xuanli Lisa, Beni Gerardo. A Validity Measure for Fuzzy Clustering // IEEE Trans. Pattern Anal. Mach. Intell. — 1991. — Vol. 13, no. 8. — P. 841–847. — URL: http://dx.doi.org/10.1109/34.85677.

[72] Xu Rui, Wunsch Donald C. Clustering algorithms in biomedical research: a review // Biomedical Engineering, IEEE Reviews in. — 2010. — Vol. 3. — P. 120–154.

[73] Yamakawa T., Tomoda S. A Neo Fuzzy Neuron and its Applications to System Identification and Prediction of the System Behaivour // Proc. 2nd Int. Conf. on Fuzzy Logic and Neural Networks (IIZUKA'92). — Iizuka, 1992. — P. 477–483.

[74] Ye Bodyanskiy, Vynokurova E, Teslenko N. Cascade GMDH-wavelet-neuro-fuzzy network // Proc. of the 4th Intern. Workshop on Inductive Modelling IWIM 2011. — 2011. — P. 22–30.