Міністерство освіти і науки України

Національний університет водного господарства та природокористування

Кафедра обчислювальної техніки

04-04-217

МЕТОДИЧНІ ВКАЗІВКИ

до лабораторних робіт із навчальної дисципліни
«Комп'ютерні системи штучного інтелекту»
(Частина 1. Способи подання інформації та пошук рішень)
для здобувачів вищої освіти другого (магістерського) рівня за спеціальності 123 «Комп'ютерна інженерія» денної та заочної форм навчання

Рекомендовано науково-методичною
комісією зі спеціальності
123 "Комп'ютерна інженерія"
Протокол № 1 від 10.09.2018 р.

Рівне – 2018

Методичні вказівки до лабораторних робіт з навчальної дисципліни "Комп’ютерні системи штучного інтелекту" (Частина 1. Способи подання інформації та пошук рішень) для здобувачів вищої освіти другого (магістерського) рівня за спеціальності 123 «Комп’ютерна інженерія» денної та заочної форм навчання / Шатний С. В. – Рівне : НУВГП. – 55 с.

Укладач: С. В. Шатний, старший викладач кафедри обчислювальної техніки.

Відповідальний за випуск – Б. Б. Круліковський, завідувач кафедри обчислювальної техніки.

ЗМІСТ

Вступ 4

1. Загальні методичні вказівки 5

2. Опис лабораторного обладнання 7

3. Лабораторна робота №1. 9

4. Лабораторна робота №2. 17

5. Лабораторна робота №3 31

6. Лабораторна робота №4 31

Рекомендована література 52

Вступ

Методичні вказівки до лабораторних робіт обов’язкової навчальної дисципліни «Комп’ютерні системи штучного інтелекту» складена відповідно до освітньо-професійної програми підготовки магістра спеціальності 123 «Комп’ютерна інженерія». Предметом вивчення навчальної дисципліни є формування у студентів теоретичних знань і розуміння принципів побудови та функціонування основних апаратних засобів обчислювальних машин та інформаційних систем, а також практичних навичок розробки, програмування інформаційно-технічних систем на основі методів та підходів штучного інтелекту. Опанування основних положень комп’ютерних систем штучного інтелекту передбачає наявність попередніх знань таких дисциплін, як "Комп’ютерні системи", "Програмування мікроконтролерів", "Технології проектування комп’ютерних систем", "Архітектура комп’ютера". На матеріалі даної дисципліни ґрунтується в подальшому написання кваліфікаційної магістерської роботи.

Перша частина лабораторного циклу (3 лабораторних робіт) присвячена дослідженню способів подачі даних та пошуку рішень в інтелектуальних системах штучного інтелекту. Для кожного типу задачі вказано порядок підготовки, перелік експериментів, методика досліджень, зміст інформації, що має бути у звіті з лабораторної роботи, а також наведено орієнтовний перелік контролюючих запитань для перевірки засвоєних знань.

Дослідження базових задач комп’ютерних систем штучного інтелекту готовить студентів до розуміння принципів роботи з розширеними задачами, що мають вивчатися в наступному циклі лабораторних робіт.

1. Загальні методичні вказівки

Перед початком першого заняття викладач проводить інструктаж з техніки безпеки в даній лабораторії і правилами протипожежної безпеки при роботі з електронно-обчислювальними машинами. Кожен студент повинен самостійно вивчити перераховані документи і розписом у спеціальному журналі кафедри засвідчити своє ознайомлення з правилами і заходами безпечного виконання робіт у лабораторії.

На першому занятті викладач повідомляє студентам план лабораторних занять на поточний семестр, рекомендує їм необхідну літературу, знайомить із прийнятою методикою підготовки, виконання, а також з порядком захисту звітів по виконаних лабораторних роботах.

Виконання кожної лабораторної роботи складається з трьох етапів:

1. Підготовка до лабораторної роботи, вивчення теоретичного матеріалу, підготовка заготовки звіту з описом назви, мети, досліджуваними алгоритмами та переліком запланованих досліджень роботи, заготовками таблиць для запису експериментальних даних. Перевірку готовності до виконання лабораторної роботи студенти проходять перед її початком. При цьому перевіряється знання досліджуваної задачі, класифікаційних ознак, вхідних та вихідних даних, основних параметрів та характеристик. В разі незадовільної підготовки студенти не допускаються до проведення досліджень. У процесі підготовки до лабораторної роботи студент повинен чітко усвідомити собі кінцеву мету лабораторного дослідження, виконувану функцію досліджуваного пристрою, характер зміни вхідних і вихідних сигналів.

2. Виконання роботи після отримання допуску починається із складання досліджуваної схеми алгоритму у відповідності із ходом роботи.

3. Оформлення звіту з лабораторної роботи в домашніх умовах. Звіт, крім попередніх даних, повинен містити результати виконання лабораторної роботи: таблиці істинності, часові діаграми досліджуваних процесів, графіки, аналіз і порівняння отриманих результатів з теоретичними, пояснення їх відмінностей (при наявності). Часові діаграми складаються таким чином, щоб вони відображали всі процеси в контрольних точках алгоритму, часові діаграми взаємопов'язаних процесів розташовані одна під іншою та синхронізовані в часі. Крім того, для полегшення розуміння на часових діаграмах вказуються причинно-наслідкові зв'язки у вигляді стрілочок в напрямку розповсюдження перехідних процесів.

Порядок, виконання досліджень у лабораторії:

1. Студент допускається до виконання чергової лабораторної роботи при наявності підготовленої до поточного заняття заготовки та при відсутності незданих звітів з попередніх робіт (2 і більше).

2. Після цього виконуються намічені дослідження, по закінченню яких результати пред'являються викладачеві і за його дозволом схема розбирається і робоче місце прибирається.

3. В протокол поточної роботи заносяться результати, що отримані студентом на занятті, підписуються викладачем, якщо дослідження виконані в повному обсязі. За отриманими даними оформляється остаточний звіт з роботи.

4. До наступної лабораторної роботи остаточно оформляється протокол і пред'являється викладачеві на наступному занятті для захисту.

5. Протоколи всіх робіт зберігаються у студента до виконання всього лабораторного циклу та використовуються для підготовки до екзамену.

Роботи № 1 - №4 виконуються на ПЕОМ із використанням спеціалізованого ПЗ MATLAB та компіляторів мови C++.

2. Опис лабораторного обладнання.

Для проведення лабораторних робіт використовуються персональні комп'ютери із встановленим програмним забезпеченням, а саме:

- MATLAB;
- NI LabView;
- CodeBlocks;

Додатково використовуються наступні програмні модулі:

- Fuzzy Toolbox;
- NNTool;
- MinGW C++;

Все перераховане програмне забезпечення використовується в навчальному та академічному режимах, а також із умовно-безкоштовною ліцензією, що відповідає ліцензійним вимогам до використання спеціалізованого програмного забезпечення.

Лабораторна робота № 1
Методи видобування асоціативних правил з великих масивів даних

Мета роботи: вивчити основні методи видобування асоціативних правил з великих масивів даних, навчитися використовувати спеціалізовані програмні засоби для видобування знань з масивів даних.

Теоретичні відомості

Операції з асоціативними правилами у пакеті MATLAB дозволяє виконувати модуль *ARMADA*. Він дозволяє створювати асоціативні правила по заданим користувачем даним в рамках середовища MATLAB.

Для запуску модулю *ARMADA* необхідно зробити папку, де знаходиться цей модулю, поточною, після чого в командному рядку середовища MATLAB написати наступну команду: armada

В результаті з'явиться головна інтерфейсна форма модулю, в якій необхідно обрати файл з вхідними даними та параметри для аналізу даних (рис. 1).

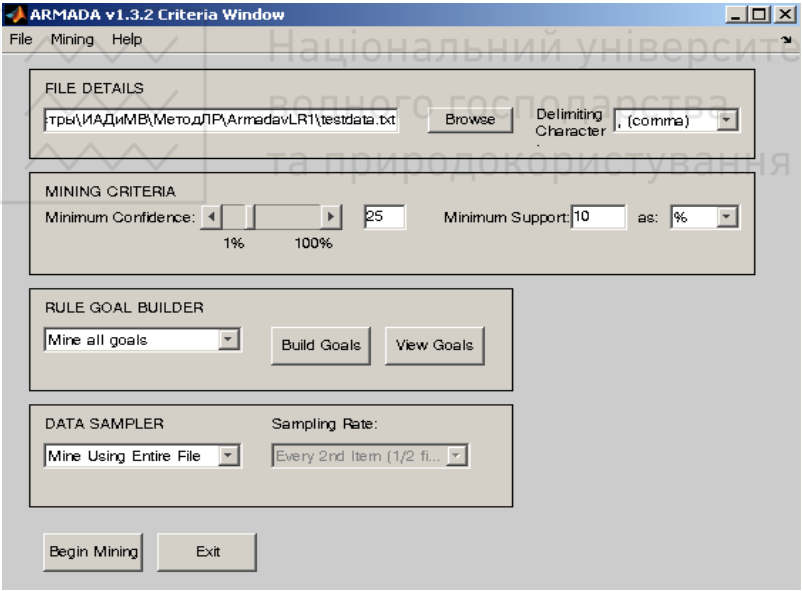

Рис. 1. Головна інтерфейсна форма модулю *ARMADA*

Для інтелектуального аналізу даних за допомогою асоціативних правил в полі FILE DETAILS необхідно ввести шлях до файлу з даними для аналізу. Файл з даними

для аналізу можна також обрати у віконному режимі, натиснувши кнопку Browse. В полі зі списком Delimiting Character необхідно вказати символ, що відокремлює дані одне від одного.

Компоненти головної форми Minimum Confidence Minimum Support призначені для введення параметрів minsupport та minconfidence, відповідно.

Для виконання аналізу даних та отримання вихідної інформації у вигляді асоціативних правил необхідно натиснути кнопку Begin Mining, після чого відбудеться процес аналізу даних, та з'явиться форма з результатами роботи програми (рис.2). Вихід з програми виконується за допомогою кнопки Exit.

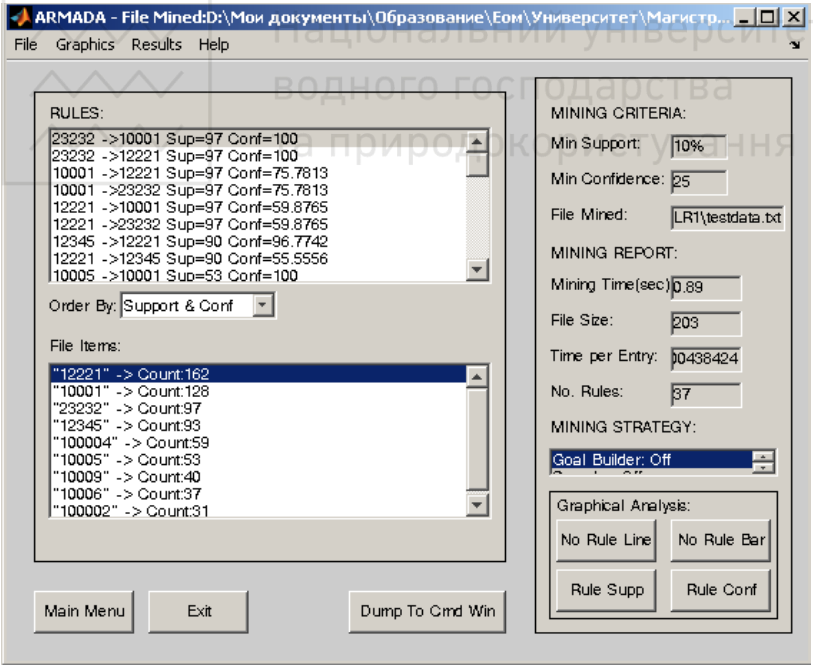

Рис. 2. Результати аналізу даних за допомогою модулю *ARMADA*

В області RULES наведено отримані асоціативні правила, в області File Items – елементи файлу для аналізу та кількість разів, які вони зустрічаються у файлі.

Компоненти в області MINING CRITERIA відображають параметри аналізу: значення параметрів minsupport та minconfidence, а також файл із даними для аналізу.

В області MINING REPORT відображається інформація про хід виконання аналізу:

– час, витрачений на виконання аналізу (Mining Time);

– розмір файлу (File Size);

– час, затрачений на один запис (Time per Entry);

– кількість отриманих асоціативних правил (No. Rules)

За допомогою головного меню отримані дані можна зберегти на диск (File –> Save) та отримати графіки, що відображають процес аналізу даних (Graphics).

Завдання до роботи

1. Ознайомитися з конспектом лекцій та рекомендованою літературою за темою роботи
2. Сформувати набір даних для обробки та аналізу.
3. Розробити за допомогою середовища Matlab програмне забезпечення для видобування асоціативних правил з великих масивів даних або вивчити рекомендоване програмне забезпечення (пакет Armada модулю Matlab) та здійснити обробку набору даних з метою виділення асоціативних правил.
4. Оформити звіт з роботи.
5. Відповісти на контрольні питання.

Зміст звіту

1. Тема та мета роботи.
2. Короткі теоретичні відомості.
3. Набір даних для обробки (якщо він великий – навести фрагмент).
4. Детальний опис процесу використання програмного забезпечення для обробки набору даних, що має бути проілюстрований зображеннями екранних форм з описом кожного їх елементу та коментарями до кожної дії.
5. Текст програми з коментарями.
6. Результати роботи програмного забезпечення (набір отриманих правил, інші характеристики).
7. Висновки, що містять відповіді на контрольні запитання, а також відображують результати виконання роботи та їх критичний аналіз.

Контрольні запитання

1. Що таке асоціативне правило?
2. Для чого призначені асоціативні правила?
3. Дати визначення понять підтримки та достовірності правила.
4. Яке призначення алгоритмів пошуку асоціативних правил?
5. На які підзадачі розбивається задача знаходження асоціативних правил?
6. Які методи використовуються для знаходження асоціативних правил?
7. Яким чином обираються значення параметрів minsupport та minconfidence?
8. Що таке числові асоціативні правила?
9. Поясніть поняття “узагальнене асоціативне правило”.
10. Що називається ієрархією елементів?

Лабораторна робота № 2
Дерева вирішальних правил

Мета роботи: вивчити основні методи побудови дерев вирішальних правил, навчитися використовувати спеціалізовані програмні засоби для побудови дерев вирішальних правил.

Теоретичні відомості

Дерева вирішальних правил у пакеті Matlab реалізовані у модулі Statistics Toolbox. Основними функціями для роботи з деревами є: treefit, treeprune, treedisp, treetest, treeval.

Таблиця 1

Функції пакету Matlab для роботи із деревами вирішальних правил

№	Формат виклику	Призначення функції
1	t = treefit(X,y)	створює дерево вирішальних правил t на основі заданих значень незалежних змінних (матриця X) та значень вихідного параметру (вектор y)
2	t1 = treeprune(t,'level',n)	створює дерево вирішальних правил t1 на основі заданого дерева t, скороченого до n-го рівня
	t1 = treeprune(t,'nodes',nod)	створює дерево вирішальних правил t1 на основі заданого дерева t, видаляючи при цьому вузли, вказані в змінній nod

3	treedisp(t)	відображує дерево t у графічному вигляді
4	c = treetest(t,'test',X,y)	виконує тестування дерева t за допомогою тестової вибірки (X, y).
5	Ycalc = treeval(t,X)	за допомогою дерева t для масиву незалежних змінних X розраховує значення вихідного параметру Ycalc

Нижче наведено приклад створення дерева вирішальних правил на основі масивів даних meas та species, що зберігаються в структурі fisheriris.

load fisheriris; % завантажити з файлу fisheriris.mat змінні meas та species для створення дерева вирішальних правил

t = treefit(meas,species); % створити дерево на основі змінних meas та species

treedisp(t,'names',{'SL' 'SW' 'PL' 'PW'}); % вивести на екран побудоване дерево t у графічному вигляді. При цьому для підпису значень незалежних змінних, на основі яких було побудоване дерево t, використовується параметр names

[c,s,n,best] = treetest(t,'cross',meas,species); % протестувати дерево t

tmin = treeprune(t,'level',best); % мінімізувати дерево t

% розрахунок точності класифікації побудованого дерева
sfit = treeval(t,meas); % за допомогою дерева t отримати відповідні числові значення класів для екземплярів з масиву meas

```
sfit = t.classname(sfit);    % отримати відповідні назви класів
mean(strcmp(sfit,species))   % розрахунок точності класифікації
```

Завдання до роботи

1. Ознайомитися з конспектом лекцій та рекомендованою літературою
2. Сформувати набір даних для обробки та аналізу.
3. Використовуючи рекомендоване програмне забезпечення здійснити обробку набору даних з метою побудови дерева вирішальних правил.
4. Використати побудоване дерево для прийняття рішень на конкретному прикладі.
5. Оформити звіт з роботи.
6. Відповісти на контрольні питання.

Зміст звіту

1. Тема та мета роботи.
2. Короткі теоретичні відомості.
3. Набір даних для обробки (якщо він великий – навести фрагмент).
4. Лістинг основних функцій програми з коментарями.
5. Результати роботи програмного забезпечення (дерево вирішальних правил, результати прийняття рішень, інші характеристики).
6. Висновки, що містять відповіді на контрольні запитання, а також відображують результати виконання роботи та їх критичний аналіз.

Контрольні запитання

1. Що таке дерево вирішальних правил? Який спосіб подання правил в них використовується?

2. Дати означення основних понять, що відносяться до теорії дерев вирішальних правил: об'єкт, атрибут, мітка класу, вузол, лист, перевірка.
3. Навести основні класи задач, до яких можуть бути застосовані дерева вирішальних правил.
4. Яким чином відбувається побудова дерева рішень? Який метод використовується для цього?
5. В чому полягає процес навчання з учителем?
6. Порівняйте методи, що реалізують дерева вирішальних правил: CART та C4.5.
7. Поясніть принцип роботи "жадібних" алгоритмів.
8. Перелічіть основні аспекти, яким приділяється увага при побудові дерев рішень.
9. В чому полягає правило відбору ознаки для розбиття? Сформулюйте загальне правило для відбору атрибуту.
10. Виконайте порівняльний аналіз критеріїв оцінки якості розбиття множини на класи.

Лабораторна робота №3
Нейро-нечіткі системи

Мета роботи: вивчити основні моделі та методи синтезу нейро-нечітких систем, освоїти принципи побудови нейро-нечітких мереж за допомогою програмних засобів, навчитися використовувати нейро-нечіткі моделі для інтелектуального аналізу даних.

Теоретичні відомості

Формування бази знань нейро-нечіткої мережі

Розглянемо об'єкт виду $y = f(x_1, x_2, ..., x_n)$ для якого зв'язок «входи x_j – вихід y» можна подати у вигляді *експертної матриці знань* (табл. В.1).

Таблиця 3.1

Експертна матриця знань

Номер правила	Якщо (входи)				То (вихід)	Вага правила
	x_1	x_2	...	x_n		
11	a_1^{11}	a_2^{11}	...	a_n^{11}	d_1	w_{11}
12	a_1^{12}	a_2^{12}	...	a_n^{12}		w_{12}
...	...	...	...	...		...
$1k_1$	$a_1^{1k_j}$	$a_2^{1k_j}$	...	$a_n^{1k_j}$		w_{1k_1}
...	...	...	...	...	...	...
m1	a_1^{m1}	a_2^{m1}	...	a_n^{m1}	d_m	w_{m1}
m2	a_1^{m2}	a_2^{m2}	...	a_n^{m2}		w_{m2}
...	...	...	...	...		...
mk_m	$a_1^{mk_m}$	$a_2^{mk_m}$	...	$a_n^{mk_m}$		w_{mk_m}

Цій матриці відповідає нечітка база знань:

$$\text{Якщо } [(x_1 = a_1^{j1}) \text{ та } (x_2 = a_2^{j1}) \text{ та} \ldots (x_n = a_n^{j1})] \text{ (з вагою } w_{j1}) \ldots$$

$$\ldots \text{або } [(x_1 = a_1^{jk_j}) \text{ та } (x_2 = a_2^{jk_j}) \text{ та} \ldots (x_n = a_n^{jk_j})] \text{ (з вагою } w_{jk_j}),$$

$$\text{то} \quad y = d_j, \quad j = 1,2,\ldots,m, p = 1,2,\ldots,k_j,$$

де a_i^{jp} – лінгвістичний терм, що оцінює змінну x_i у рядку p, k_j – кількість рядків-кон'юнкцій, що відповідають класу d_j вихідної змінної y, w_{jp} – число в діапазоні [0,1], що характеризує суб'єктивну міру впевненості експерта щодо висловлення з номером p.

Якщо вихідна змінна (консеквент) є дискретною, то класи d_j, j=1, 2, ..., m, формуються як номери дискретних значень вихідної змінної.

Мережа Мамдані

Нейро-нечіткий апроксиматор Мамдані (Mamdani neuro-fuzzy approximator) є найбільш узагальненою моделлю нейро-нечіткої мережі, побудованої на правилах Мамдані. Структуру мережі Мамдані зображено на рис. 3.1.

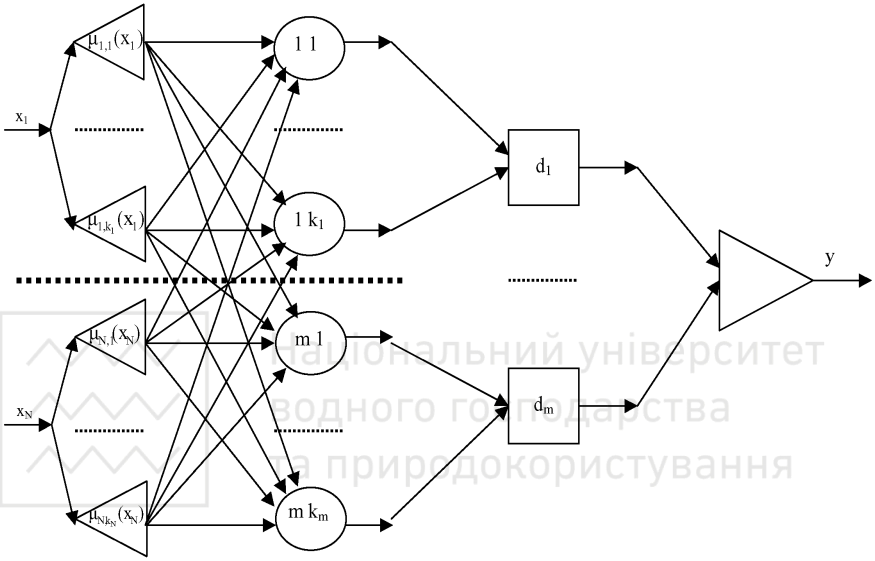

Рис. 3.1. Структура нейро-нечіткої мережі Мамдані

Нейро-нечітка мережа Мамдані має п’ять шарів.

Перший (вхідний) шар утворюють елементи вхідного вектора.

Другий шар містить нечіткі терми, які відповідають вхідним змінним, та обчислює приналежності вхідного вектора до кожного з нечітких термів:

$$\mu_{jp}(x_i) = \frac{1}{1+\left(\frac{x_i - b_i^{jp}}{c_i^{jp}}\right)^2}, \quad i=\overline{1,N}, p=\overline{1,k_j}, j=\overline{1,m},$$

де $\mu_{jp}(x_i)$ – функція приналежності змінної x_i терму p-го рядка j-го правила, a_i^{jp}, b_i^{jp}, c_i^{jp} – параметри настроювання функцій приналежності.

Третій шар утворюють рядки-кон'юнкції антецедентів правил нечіткої бази знань: кожний вузол цього шару реалізує оператор Т-норми та видає ступінь відповідності вхідного вектора до співставленого вузлу правила.

Четвертий шар поєднує правила в класи d_j, j=l, 2, …, m, та обчислює ступені приналежності вхідного вектора до відповідних термів вихідної змінної; кожний вузол цього шару реалізує оператор Т-конорми.

Функція приналежності вхідного вектора до j-го терму d_j вихідної змінної y визначається як:

$$\mu_{d_j}(y) = \max_{p=\overline{1,k_j}} \{w_{jp} \min_{i=\overline{1,N}} [\mu_{jp}(x_i)]\}.$$

П'ятий шар містить один нейрон, що поєднує приналежності до нечітких термів вихідної змінної та виконує операцію дефазифікації:

- для дискретного виходу: $$y = \frac{\sum_{j=1}^{m} d_j \mu_{d_j}(y)}{\sum_{j=1}^{m} \mu_{d_j}(y)};$$

- для неперервного виходу:

$$y = \frac{y\mu_{d_1}(y) + y_1\mu_{d_2}(y) + \ldots + y_{m-1}\mu_{d_m}(y)}{\sum_{j=1}^{m} \mu_{d_j}(y)}.$$

Число вузлів у нейро-нечіткій мережі визначається так: шар 1 – за кількістю елементів вхідного вектора; шар 2 – за кількістю нечітких термів у базі знань (n); шар 3 – за кількістю строк-кон'юнкцій у базі знань; шар 4 – за кількістю класів, на які розбивається діапазон вихідної змінної (m).

Дугам графа присвоюються такі ваги: одиниця (дуги між першим і другим шарами); функції приналежності входу до нечіткого терму (дуги між другим і третім шарами); ваги правил (дуги між третім і четвертим шарами); одиниця (дуги між четвертим і п'ятим шарами). Пороги нейронів вважають відсутніми або такими, що дорівнюють нулю.

Навчання нейро-нечіткої мережі Мамдані здійснюють методом зворотного поширення помилки з метою мінімізації критерію помилки

$$E = \frac{1}{2}\sum_{s=1}^{S}(y^{s*} - y^{s})^2 ,$$

що застосовується в теорії нейронних мереж, де y^{s*} та y^{s} – бажане та розрахункове значення виходу мережі для s-го екземпляра.

Доти, поки помилка E не досягне максимального прийнятного рівня, послідовно для кожного s-го екземпляра x^s, s = 1, 2, …, S, виконують два етапи. На першому етапі (прямий хід) для s-го екземпляра обчислюють розрахункове значення виходу мережі y^s. На другому етапі (зворотний хід) обчислюють значення миттєвої помилки E_t та перераховують ваги зв'язків:

$$E_t = \frac{1}{2}(y^{s*}(t) - y^{s}(t))^2,$$

$$w_{jp}(t+1) = w_{jp}(t) - \alpha\frac{\partial E_t}{\partial w_{jp}(t)},$$

$$c_i^{jp}(t+1) = c_i^{jp}(t) - \alpha\frac{\partial E_t}{\partial c_i^{jp}(t)},$$

$$b_i^{jp}(t+1) = b_i^{jp}(t) - \alpha\frac{\partial E_t}{\partial b_i^{jp}(t)}, \quad j = \overline{1, m}, \quad i = \overline{1, N}, \quad p = k_j,$$

де $y^{s*}(t)$ та $y^{s}(t)$ – бажане та розрахункове значення виходу мережі для s-го екземпляра на t-му кроці навчання, $w_{jp}(t)$, $c_i^{jp}(t)$, $b_i^{jp}(t)$ – ваги правил і параметри функцій приналежності на t-му кроці навчання; α – величина кроку навчання.

Часткові похідні, що використовуються у правилах корегування параметрів мережі, характеризують чуттєвість помилки E_t до зміни параметрів та обчислюються таким чином:

$$\frac{\partial E_t}{\partial w_{jp}} = \varepsilon_1\varepsilon_2\varepsilon_3 \frac{\partial \mu_{d_j}}{\partial w_{jp}}, \quad \frac{\partial E_t}{\partial c_i^{jp}} = \varepsilon_1\varepsilon_2\varepsilon_3\varepsilon_4 \frac{\partial \mu_{jp}(x_i)}{\partial c_i^{jp}},$$

$$\frac{\partial E_t}{\partial b_i^{jp}} = \varepsilon_1\varepsilon_2\varepsilon_3\varepsilon_4 \frac{\partial \mu_{jp}(x_i)}{\partial b_i^{jp}},$$

де $\varepsilon_1 = \frac{\partial E_t}{\partial y} = y_t^{s*} - y_t^{s}$,

$$\varepsilon_2 = \frac{\partial y}{\partial \mu_{d_j}} = \frac{d_j \sum_{q=1}^{m} \mu_{d_q} - \sum_{q=1}^{m} d_q \mu_{d_q}}{\left(\sum_{q=1}^{m} \mu_{d_q}\right)^2}, \quad \varepsilon_3 = \frac{\partial \mu_{d_j}}{\partial \left(\prod_{i=1}^{N} \mu_{jp}(x_i)\right)} = w_{jp},$$

$$\varepsilon_4 = \frac{\partial \left(\prod_{i=1}^{N} \mu_{jp}(x_i)\right)}{\partial \mu_{jp}(x_i)} = \frac{1}{\mu_{jp}(x_i)} \prod_{i=1}^{N} \mu_{jp}(x_i), \quad \frac{\partial \mu_{d_j}}{\partial w_{jp}} = \prod_{i=1}^{N} \mu_{jp}(x_i),$$

$$\frac{\partial \mu_{jp}(x_i)}{\partial c_i^{jp}} = \frac{2c_i^{jp}\left(x_i - b_i^{jp}\right)^2}{\left(\left(c_i^{jp}\right)^2 + \left(x_i - b_i^{jp}\right)^2\right)^2}, \quad \frac{\partial \mu_{jp}(x_i)}{\partial b_i^{jp}} = \frac{2\left(c_i^{jp}\right)^2\left(x_i - b_i^{jp}\right)}{\left(\left(c_i^{jp}\right)^2 + \left(x_i - b_i^{jp}\right)^2\right)^2}.$$

Мережа Такагі-Сугено-Канга

Нейро-нечітка мережа Такагі-Сугено-Канга (Takagi-Sugeno-Kang Neuro-fuzzy network, TSK) виконує нечітке виведення із використанням N змінних x_j та m правил:

$П_k$: Якщо $x_1 \in A^{(k)}{}_1$ та $x_2 \in A^{(k)}{}_2$ та … та $x \in A^{(k)}{}_N$, тоді

$$y_k = w_0^k + \sum_{j=1}^{N} w_j^k x_j,$$

де $A^{(k)}{}_j$ – нечіткий терм, до якого має належати j-та вхідна змінна, щоб активізувати k-те правило, $w^k{}_j$ – ваговий коефіцієнт, k = 1, 2, …, m.

Мережа Такагі-Сугено-Канга (рис. 3.2) складається з п’яти шарів.

Перший шар виконує окрему фазифікацію кожної вхідної змінної x_j, j=1, 2, ..., N, визначаючи для кожного k-го правила значення функції приналежності

$$\mu_k^{(1)}(x_j) = \frac{1}{1 + \left(\frac{x_j - c_{jk}}{\sigma_{jk}} \right)^{2b_{jk}}},$$

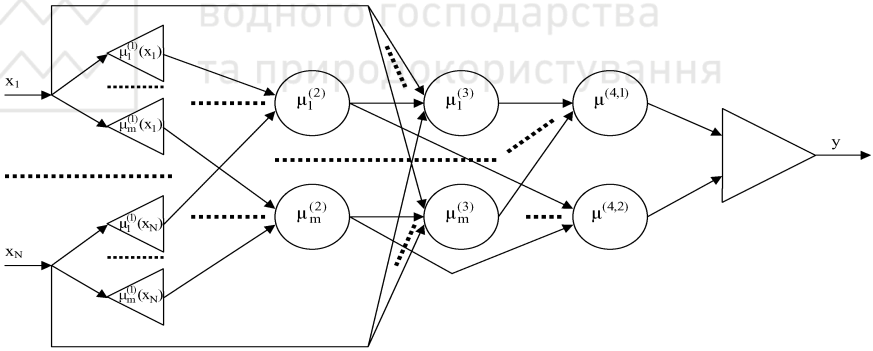

Рис. 3.2. Мережа Такагі-Сугено-Канга

параметри якої $c_{jk}, \sigma_{jk}, b_{jk}$ підлягають адаптації в процесі навчання.

Другий шар виконує агрегування функцій приналежності вхідних змінних до термів антецедентів нечітких правил, визначаючи приналежність вхідного вектора до k-го правила (рівень активації правила):

$$\mu_k^{(2)} = \prod_{j=1}^{N} \mu_k^{(1)}(x_j, c_{jk}, \sigma_{jk}, b_{jk}),\ k = 1, 2, \ldots, m.$$

Третій шар генерує значення функцій консеквентів нечітких правил з урахуванням рівнів активації правил:

$$\mu_k^{(3)} = \mu_k^{(2)}\left(w_0^k + \sum_{j=1}^{N} w_j^k x_j\right).$$

Це параметричний шар, у якому адаптації підлягають лінійні ваги w^k_j, k = 1,2, ..., m, j = 1, 2, ..., N.

Четвертий шар агрегує m правил виведення (перший нейрон) і генерує нормалізуючий сигнал (другий нейрон):

$$\mu^{(4,1)} = \sum_{k=1}^{m} w_k^{(4)} \mu_k^{(3)},\ \mu^{(4,2)} = \sum_{k=1}^{m} \mu_k^{(3)}.$$

П’ятий (вихідний) шар містить один нейрон і здійснює нормалізацію, формуючи вихідний сигнал y: $y(x_1, x_2, \ldots, x_N) = \mu^{(4,1)} / \mu^{(4,2)}$.

Мережа Такагі-Сугено-Канга містить тільки два параметричних шари (перший і третій), параметри яких уточнюються в процесі навчання. Параметри першого шару будемо називати нелінійними параметрами, а параметри третього шару – лінійними вагами.

Якщо прийняти, що в конкретний момент часу параметри умов зафіксовані, то функція $y(x_1, x_2, \ldots, x_N)$ є лінійною щодо змінних $x_1, x_2, \ldots, x_N$.

При наявності N вхідних змінних кожне правило формує N+1 змінних w^k_j лінійної залежності $y_k(x_1, x_2, \ldots, x_N)$. При m правилах виведення це дає m(N+1) лінійних параметрів мережі. У свою чергу, кожна функція приналежності використовує три параметри $c_{jk}, \sigma_{jk}, b_{jk}$, що підлягають адаптації. Якщо прийняти, що кожна змінна x_j, характеризується власною функцією приналежності, то при m правилах виведення ми одержимо 3m нелінійних

параметрів. У сумі це дає m(4N+1) лінійних і нелінійних параметрів, значення яких повинні підбиратися в процесі навчання мережі.

На практиці для зменшення кількості параметрів, що адаптуються, оперують меншою кількістю незалежних функцій приналежності для окремих змінних, керуючись правилами, у яких комбінуються функції приналежності різних змінних. Якщо прийняти, що кожна змінна x_j має k_j різних функцій приналежності, то максимальна кількість правил, яку можна створити при їхньому комбінуванні, складе: $m = k_j$. У такий спосіб сумарна кількість нелінійних параметрів мережі при m правилах висновку зменшується з 3m у загальному випадку до $3Nm^{1/N}$. Кількість лінійних параметрів при подібній модифікації залишається без змін, тобто m(N+1).

Завдання мережі полягає в такому відображенні пар даних <x, y>, щоб для вхідного вектора x^s розрахункове значення вихідної ознаки $y^{s*}=y(x^s)$ не сильно відрізнялося б від співставленого x^s фактичного значення цільової ознаки y^s.

Навчання мережі Такагі-Сугено-Канга засновано на мінімізації цільової функції

$$E = \frac{1}{2}\sum_{s=1}^{S}\left(y^{s*} - y^{s}\right)^2,$$

де S – кількість навчаючих пар $<x^s, y^s>$, та є контрольованим.

Гібридний алгоритм навчання застосовується для мережі Такагі-Сугено-Канга й інших нейро-нечітких мереж, подібних їй. У гібридному алгоритмі підлягаючі адаптації параметри розподіляються на дві групи: одна група складається з лінійних параметрів w^k_j третього шару, а інша група – з параметрів нелінійних функцій приналежності першого шару. Уточнення параметрів проводиться в два етапи.

На першому етапі при фіксації визначених значень параметрів функцій приналежності (у першому циклі – це значення, отримані в результаті ініціалізації) шляхом розв'язку системи лінійних рівнянь розраховуються лінійні параметри w^k_j. При відомих значеннях функції приналежності залежність $y(x_1, x_2, \ldots, x_N)$ можна подати в лінійній формі:

$$y(x_1,.x_2,..., x_N) = \sum_{k=1}^{m} w'_k \left(w^k_0 + \sum_{j=1}^{N} w^k_j x_j \right),$$

$$w'_k = \left(\sum_{p=1}^{m} \left(\prod_{j=1}^{N} \mu^{(1)}_p (x_j) \right) \right)^{-1} \prod_{j=1}^{N} \mu^{(1)}_k (x_j).$$

При S екземплярах навчаючої вибірки $<x^s, y^s>$, s= 1, 2, ..., S, одержимо систему з лінійних рівнянь у матричній формі: Aw=y, де $w=(w^1, w^2, \ldots, w^m)^T$, $w^k=(w^k_0, w^k_1, \ldots, w^k_N)^T$, $y=(y_1, y_2, \ldots, y_S)^T$, $A=(a_1, a_2, \ldots, a_S)^T$, $a_s=(a_{s1}, a_{s2}, \ldots, a_{sm})^T$, $a_{sk}=(\mu^{(2)}_k(x^s), \mu^{(2)}_k(x^s)x^s_1, \mu^{(2)}_k(x^s)x^s_2, \ldots, \mu^{(2)}_k(x^s)x^s_N)^T$. Розмірність матриці A дорівнює S(N+l)m, при цьому звичайно кількість рядків S значно більше кількості стовпців (N+1)m. Розв'язок цієї системи: $w = A^{-1}y$.

На другому етапі після фіксації значень лінійних параметрів w^k_j розраховуються вихідні сигнали мережі $y=\{y^{s*}\}$ (s=1, 2, …,S): y=Aw, а слідом за ними – вектор помилки $e=\{e^s\}$, $e^s = y^{s*} - y^s$. Сигнали помилок направляються через підключену мережу за напрямком до входу мережі (зворотне поширення) аж до першого шару, де можуть бути розраховані компоненти градієнта цільової функції щодо конкретних параметрів $c_{jk}, \sigma_{jk}, b_{jk}$.

Після формування вектора градієнта параметри уточнюються з використанням одного з градієнтних методів навчання. Якщо застосовується найпростіший метод

найшвидшого спуска, то відповідні формули адаптації приймають форму:

$$c_{jk}(t+1) = c_{jk}(t) - \alpha_c \frac{\partial E(t)}{\partial c_{jk}}, \quad \sigma_{jk}(t+1) = \sigma_{jk}(t) - \alpha_\sigma \frac{\partial E(t)}{\partial \sigma_{jk}},$$

$$b_{jk}(t+1) = b_{jk}(t) - \alpha_b \frac{\partial E(t)}{\partial b_{jk}},$$

де t – номер ітерації, α_c, α_σ, α_b – коригувальні прирости (кроки навчання).

Після уточнення нелінійних параметрів знову запускається процес адаптації лінійних параметрів (перший етап) і нелінійних параметрів мережі (другий етап). Цей цикл повторюється аж до стабілізації всіх параметрів процесу навчання.

Остаточний вид формул настроювання параметрів залежить як від використовуваного визначення функції помилки на виході мережі, так і від виду функції приналежності. Так при використанні функцій Гауса як функцій приналежності та середньоквадратичної помилки, часткові похідні цільової функції приймають вигляд:

$$\frac{\partial E}{\partial c_{jk}} = \left(y^{s*} - y^s\right)\sum_{p=1}^{m}\left(w_0^p + \sum_{i=1}^{N} w_i^p x_i\right)\frac{\partial w_p^{'}}{\partial c_{jk}},$$

$$\frac{\partial E}{\partial \sigma_{jk}} = \left(y^{s*} - y^s\right)\sum_{p=1}^{m}\left(w_0^p + \sum_{i=1}^{N} w_i^p x_i\right)\frac{\partial w_p^{'}}{\partial \sigma_{jk}},$$

$$\frac{\partial E}{\partial b_{jk}} = \left(y^{s*} - y^s\right)\sum_{p=1}^{m}\left(w_0^p + \sum_{i=1}^{N} w_i^p x_i\right)\frac{\partial w_p^{'}}{\partial b_{jk}},$$

$$\frac{\partial w_p^{'}}{\partial c_{jk}} = \beta_{pk}\left(\prod_{i=1, i\neq j}^{N}\mu_k^{(1)}(x_i)\right)\left(\frac{2b_{jk}}{\sigma_{jk}}\right)\left(\frac{x_j - c_{jk}}{\sigma_{jk}}\right)^{2b_{jk}-1}\left(1 + \left(\frac{x_j - c_{jk}}{\sigma_{jk}}\right)^{-2b_{jk}}\right)^{-2},$$

$$\frac{\partial w'_p}{\partial \sigma_{jk}} = \beta_{pk}\left(\prod_{i=1, i\neq j}^{N} \mu_k^{(1)}(x_i)\right)\left(\frac{2b_{jk}}{\sigma_{jk}}\right)\left(\frac{x_j - c_{jk}}{\sigma_{jk}}\right)^{2b_{jk}}\left(1 + \left(\frac{x_j - c_{jk}}{\sigma_{jk}}\right)^{-2b_{jk}}\right)^{-2},$$

$$\frac{\partial w'_p}{\partial c_{jk}} = \beta_{pk}\left(\prod_{i=1, i\neq j}^{N} \mu_k^{(1)}(x_i)\right)\left(-2\left(\frac{x_j - c_{jk}}{\sigma_{jk}}\right)^{2b_{jk}} \ln\left(\frac{x_j - c_{jk}}{\sigma_{jk}}\right)\right)\left(1 + \left(\frac{x_j - c_{jk}}{\sigma_{jk}}\right)^{-2b_{jk}}\right)^{-2},$$

$$\beta_{pk} = \left(\delta_{pk}\sum_{q=1}^{m}\left(\prod_{j=1}^{N}\mu_q^{(1)}(x_j)\right) - \prod_{j=1}^{N}\mu_k^{(1)}(x_j)\right)\left(\sum_{q=1}^{m}\left(\prod_{j=1}^{N}\mu_q^{(1)}(x_j)\right)\right)^{-2},$$

$$\delta_{pk} = \begin{cases} 1, p = k, \\ 0, p \neq k. \end{cases}$$

Незважаючи на складну структуру наведених формул, що виражають компоненти вектора градієнта, вони дозволяють аналітично визначити величини, необхідні для уточнення параметрів нечіткої мережі.

При практичній реалізації гібридного методу навчання нечітких мереж домінуючим фактором їхньої адаптації вважається перший етап, на якому ваги w^k_j підбираються з використанням псевдоінверсії за один крок. Для зрівноважування його впливу другий етап (підбір нелінійних параметрів градієнтним методом) багаторазово повторюється в кожному циклі.

Гібридний алгоритм є одним з найбільш ефективних способів навчання нейро-нечітких мереж. Його головна особливість полягає в розподілі процесу навчання на два відособлених у часі етапи. На кожному етапі уточнюється тільки частина параметрів мережі. Якщо взяти до уваги, що обчислювальна складність кожного алгоритму оптимізації пропорційна (нелінійно) кількості параметрів, то зменшення розмірності задачі оптимізації істотно скорочує кількість математичних операцій і збільшує

швидкість виконання алгоритму. Завдяки цьому гібридний алгоритм є значно більш ефективним, ніж звичайний градієнтний алгоритм фронтального типу, відповідно до якого уточнення всіх параметрів мережі робиться паралельно й одночасно.

Мережа ANFIS

Нейро-нечітка мережа ANFIS (Adaptive Neuro Fuzzy Inference System – адаптивна система нейро-нечіткого виведення) запропонована Янгом (R. Jang) і реалізує систему нечіткого виведення Такагі-Сугено у вигляді п’ятишарової нейронної мережі прямого поширення сигналу (рис. 3.3).

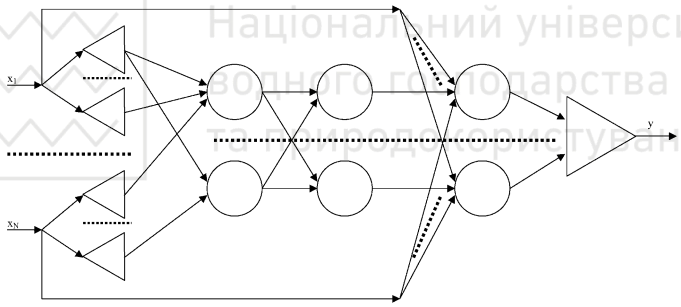

Рис. 3.3. Мережа ANFIS

Мережа ANFIS є ізоморфною базі правил Такагі-Сугено.

Перший шар містить нейрони, які відповідають нечітким термам вхідних змінних із функціями приналежності $\mu_{i,j}^{(1)}(x_i)$, де x_i – і-ий вхід, j – номер нечіткої множини, визначеної для і-го входу. Як функція приналежності зазвичай обирається колоколообразна функція або функція Гауса. Входи мережі з’єднані тільки зі

своїми термами. Кількість вузлів першого шару дорівнює сумі потужностей терм-множин вхідних змінних.

Другий шар містить m нейронів, на входи яких поступають значення з виходів вузлів першого шару, що формують антецеденти правил. Кожний k-ий нейрон другого шару знаходить ступінь виконання відповідного правила:

$$\mu_k^{(2)} = \min_{\substack{i=1,2,\dots,N \\ j=1,2,\dots,k_i}} (w_{i,j}^{(2,k)} \mu_{i,j}^{(1)}) \text{ або } \mu_k^{(2)} = \prod_{i=1}^{N} \prod_{j=1}^{k_i} \mu_{i,j}^{(1)}, \; w_{i,j}^{(2,k)} = 1,$$

де $w_{i,j}^{(2,k)} = 1$, якщо j-ий терм i-ої ознаки входить в умову k-го правила, $w_{i,j}^{(2,k)} = 0$ – у протилежному випадку.

Третій шар містить m нейронів, які знаходять нормалізовані ступені виконання правил:

$$\mu_j^{(3)} = \frac{\mu_j^{(2)}}{\sum_{i=1}^{m} \mu_i^{(2)}}, \; j = 1, 2, \dots, m.$$

Четвертий шар містить m нейронів, які обчислюють консеквенти (лінійні комбінації вхідних сигналів з урахуванням ступенів виконання) правил:

$$y_j = \mu_j^{(3)} \sum_{i=1`}^{N} w_i^{(4,j)} x_i, \; j = 1, 2, \dots, m.$$

Кожний вузол четвертого шару з’єднаний з одним вузлом третього шару а також із усіма входами мережі.

П’ятий шар містить один нейрон, який обчислює загальний вихід мережі, складаючи консеквенти правил:

$$y = \sum_{j=1}^{m} y_j \; .$$

Для налагодження параметрів ANFIS застосовують комбінацію градієнтного спуску у вигляді алгоритму зворотного поширення помилки і методу найменших квадратів.

Алгоритм зворотного поширення помилки налагоджує параметри антецедентів правил (функцій приналежності). Методом найменших квадратів оцінюються коефіцієнти висновків правил, тому що вони лінійно зв'язані з виходом мережі.

Кожна ітерація процедури настроювання виконується в два етапи. На першому етапі на входи подається навчаюча вибірка, і за відхилом між бажаною і дійсною поведінкою мережі ітераційним методом найменших квадратів знаходяться оптимальні параметри вузлів четвертого шару. На другому етапі залишковий відхіл передається з виходу мережі на входи, і методом зворотного поширення помилки модифікуються параметри вузлів першого шару. При цьому знайдені на першому етапі коефіцієнти висновків правил не змінюються. Ітераційна процедура настроювання продовжується поки відхіл перевищує заздалегідь установлене значення.

Завдання до роботи

1. Ознайомитися з конспектом лекцій та рекомендованою літературою
2. Обґрунтовано сформувати набір даних для обробки та аналізу.
3. Використовуючи рекомендоване програмне забезпечення здійснити обробку набору даних з метою побудови нейро-нечітких мереж.
4. Спробувати використати різні алгоритми кластеризації, різні кількості функцій приналежності для входів, різні кількості циклів навчання та різні алгоритми навчання.
5. Використати побудовані нейро-нечіткі мережі для прийняття рішень на конкретному прикладі.
6. Проаналізувати отримані результати та відповісти на питання: який алгоритм кластер-аналізу призводить до отримання мережі меньшої

складності; як впливає задана кількість циклів навчання на точність навчання; як вливає задана точність навчання на тривалість навчання; які вимоги мають пред'являтися до навчальної вибірки та як це вплине на процес навчання?

7. Оформити звіт з роботи.
8. Відповісти на контрольні питання.

Зміст звіту

1. Тема та мета роботи.
2. Короткі теоретичні відомості.
3. Обґрунтовано обраний набір даних для обробки (якщо він великий – навести фрагмент).
4. Опис процесу використання програмного забезпечення для обробки набору даних, що має бути ілюстрований зображеннями екранних форм.
5. Тексти програм.
6. Результати роботи програмного забезпечення (зображення структури отриманої мережі, її ваги, результати прийняття рішень, інші характеристики).
7. Висновки, що містять відповіді на контрольні запитання, а також відображують результати виконання роботи та їх критичний аналіз.

Контрольні запитання

1. У чому полягають особливості інтегрованих нейро-нечітких систем?

2. Опишіть архітектуру та методи навчання нейро-нечітких мереж: апроксиматора Мамдані, FALCON, мережі Такагі-Сугено-Канга, мережі Ванга-Менделя, ANFIS, NNFLC, SONFIN, нечіткого персептрона, NEFCON, NEFCLASS, NEFPROX, NNDFR, GARIC, FINEST, FUN,

EFuNN, dmEFuNN, гібридної мережі з нечіткою самоорганізацією.

3. У чому полягають основні принципи використання методу зворотного поширення помилки для навчання нейро-нечітких мереж?

3. Які існують основні етапи гібридного алгоритму навчання мереж Такагі-Сугено-Канга?

5. Як визначити апріорну інформацію про навчаючу вибірку?

6. У чому полягає метод редукції нечітких термів?

7. Як і для чого можна виконати об'єднання суміжних термів по ознаках?

8. Як побудувати тришарову розпізнаючу нейро-нечітку модель у неітеративному режимі?

9. Як побудувати чотиришарову розпізнаючу нейро-нечітку модель у неітеративному режимі?

10. Як і для чого можна побудувати ієрархічну логічно прозору нейро-нечітку модель?

Лабораторна робота № 4
Нечітка логіка та нейро-нечіткі множини

Мета роботи: вивчити основні методи нечіткої логіки, методи роботи із нейро-нечіткими множинами в середовищі MATLAB FuzzyLogic Toolbox

Теоретичні відомості

Операції з нечіткою логікою у пакеті MATLAB дозволяє виконувати модуль *Fuzzy Logic Toolbox*. Він дозволяє створювати системи нечіткого логічного виведення і нечіткої класифікації в рамках середовища MatLab, з можливістю їхнього інтегрування в Simulink.

Fuzzy Logic Toolbox містить наступні категорії

програмних інструментів: функції; інтерактивні модулі з графічним користувальницьким інтерфейсом (з GUI); блоки для пакета Simulink; демонстраційні приклади.

FIS-структура

Базовим поняттям Fuzzy Logic Toolbox є *FIS-структура* – система нечіткого виведення (Fuzzy Inference System). FIS-структура містить усі необхідні дані для реалізації функціонального відображення “входи-виходи” на основі нечіткого логічного виведення відповідно до схеми, приведеної на рис. 4.1.

Рис. 4.1. Нечітке логічне виведення

Позначення: X – вхідний чіткий вектор; $\tilde{X}$ - вектор нечітких множин, що відповідає вхідному вектору X; $\tilde{Y}$ - результат логічного виведення у виді вектора нечітких множин; Y - вихідний чіткий вектор.

Система нечіткого логічного виведення представляється в робочій області MatLab у вигляді структури даних, зображеної на рис. 4.2.

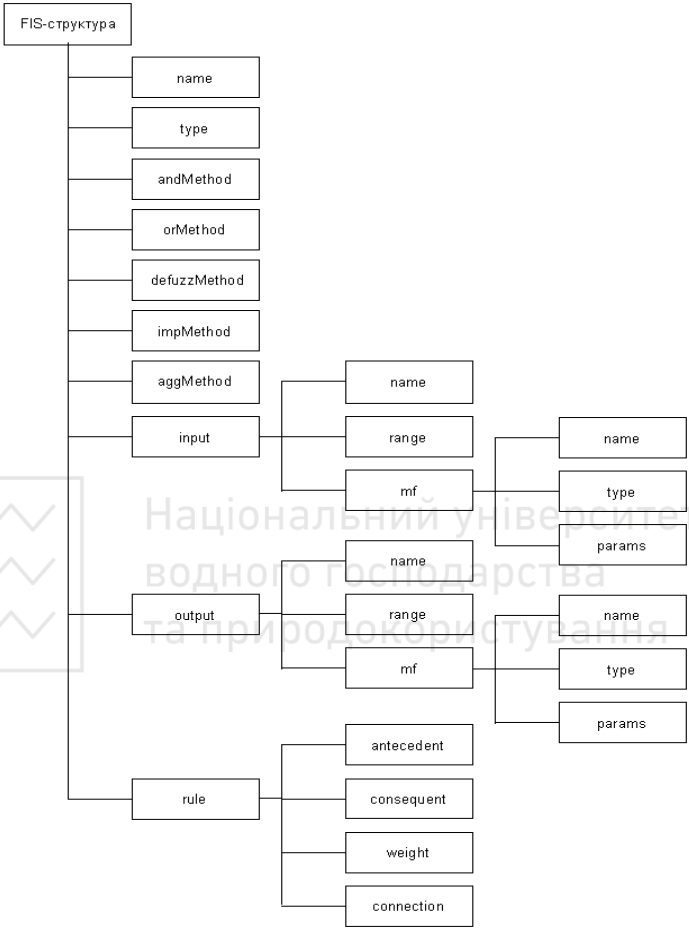

Рис. 4.2. FIS-структура

Існує два *способи завантаження FIS* у робочу область: зчитування з диска за допомогою функції readfis; передача з основного fis-редактора шляхом вибору в меню File підміню Export і команди To workspace.

Поля структури дані системи нечіткого логічного виведення призначені для збереження наступної інформації:

– name – найменування системи нечіткого логічного виведення;

– type – тип системи (припустимі значення 'Mamdani' та 'Sugeno');

– andMethod – реалізація логічної операції "ТА" (запрограмовані реалізації: 'min' – мінімум і 'prod' – множення);

– orMethod – реалізація логічної операції "АБО" (запрограмовані реалізації: 'max' – максимум і 'probor' – імовірнісне "АБО");

– defuzzMethod – метод дефазифікації (запрограмовані методи для систем типу Мамдані: 'centroid' – центр ваги; 'bisector' – медіана; 'lom' – найбільший з максимумів; 'som' – найменший з максимумів; 'mom' – середнє з максимумів; запрограмовані методи для систем типу Сугено: 'wtaver' – зважене середнє і 'wtsum' – зважена сума);

– impMethod – реалізація операції імплікації (запрограмовані реалізації: 'min' – мінімум і 'prod' – множення);

– aggMethod – реалізація операції об'єднання функцій приналежності вихідної змінної (запрограмовані реалізації: 'max' – максимум; 'sum' – сума і 'probor' – імовірнісне "АБО");

– input – масив вхідних змінних;

– input.name – найменування вхідної змінної;

– input.range – діапазон зміни вхідної змінної;

– input.mf – масив функцій приналежності вхідної змінної;

– input.mf.name – найменування функції приналежності вхідної змінної;

– input.mf.type – модель функції приналежності вхідної змінної (запрограмовані моделі: dsigmf - функція приналежності у виді різниці між двома сигмоїдними функціями; gauss2mf – двостороння гаусівська функція приналежності; gaussmf – гаусівська функція приналежності; gbellmf – узагальнена колоколообразна функція приналежності; pimf – пі-подібна функція приналежності; psigmf – добуток двох сигмоїдних функцій приналежності; sigmf – сигмоїдна функція приналежності; smf – s-подібна функція приналежності; trapmf – трапецієподібна функція приналежності; trimf - трикутна функція приналежності; zmf – z-подібна функція приналежності);

– input.mf.params – масив параметрів функції приналежності вхідної змінної;

– output – масив вихідних змінних;

– output.name – найменування вихідної змінної;

– output.range – діапазон зміни вихідної змінної;

– output.mf – масив функцій приналежності вихідної змінної;

– output.mf.name – найменування функції приналежності вихідної змінної;

– output.mf.type – модель функції приналежності вихідної змінної (запрограмовані моделі для системи типу Мамдані: dsigmf – функція приналежності у виді різниці між двома сигмоїдними функціями; gauss2mf – двостороння гаусівська функція приналежності; gaussmf – гаусівська функція приналежності; gbellmf – узагальнена колоколообразна функція приналежності; pimf – пі-подібна функція приналежності; psigmf – добуток двох сигмоїдних функцій приналежності; запрограмовані моделі для

системи типу Сугено: constatnt – константа (функція приналежності у виді сінглтона); linear – лінійна комбінація вхідних змінних);

– output.mf.params – масив параметрів функції приналежності вихідної змінної;

– rule - масив правил нечіткої бази знань;

– rule.antecedent – посилки правила (вказуються порядкові номери термів у порядку запису вхідних змінних. Число 0 указує на те, що значення відповідної вхідний змінної не впливає на істинність правила);

– rule.consequent – висновок правила (вказуються порядкові номери термів у порядку запису вихідних змінних. Число 0 указує на те, що правило не поширюється на відповідну вихідну змінну);

– rule.weight – вага правила. Задається числом з діапазону [0, 1];

– rule.connection – логічне зв'язування змінних усередині правила: 1 – логічне "ТА"; 2 – логічне "АБО".

Для доступу до властивостей системи нечіткого логічного виведення досить вказати ім'я відповідного поля. Наприклад, команда FIS_NAME.rule(1).weight=0.5 встановлює вагу першого правила в 0.5, команда length(FIS_NAME.rule) визначає кількість правил у базі знань, а команда FIS_NAME.input(1).mf(1).name='низький' перейменовує перший терм першої вхідної змінної в “низький”.

FIS-редактор

FIS-редактор (рис. 4.3) призначений для створення, збереження, завантаження і виведення у друк систем нечіткого логічного виведення, а також для редагування наступних властивостей: тип системи; найменування системи; кількість вхідних і вихідних змінних; найменування вхідних і вихідних змінних; параметри нечіткого логічного виведення.

Завантаження FIS-редактора відбувається за допомогою команди fuzzy. У результаті з'являється інтерактивне графічне вікно. У нижній частині графічного вікна FIS-редактори розташовані кнопки Help і Close, що дозволяють викликати вікно довідки і закрити редактор, відповідно.

FIS-редактор містить 8 меню. Це три загальносистемних меню - File, Edit, View, і п'ять меню для вибору параметрів нечіткого логічного виведення – And Method, Or Method, Implication, Aggregation і Defuzzification.

Меню File – це загальне меню для всіх GUI-модулів використовуваних із системами нечіткого логічного виведення.

За допомогою команди New FIS... користувач має можливість створити нову систему нечіткого логічного виведення. При виборі цієї команди з'являться дві альтернативи: Mamdani і Sugeno, що визначають тип створюваної системи. Створити систему типу Mamdani можна також натисканням Ctrl+N.

За допомогою команди Import користувач має можливість завантажити раніше створену систему нечіткого логічного виведення. При виборі цієї команди з'являться дві альтернативи From Workspace... і From disk, що дозволяють завантажити систему нечіткого логічного виведення з робочої області MatLab і з диска, відповідно. При виборі команди From Workspace... з'явиться діалогове вікно, у якому необхідно вказати ідентифікатор системи нечіткого логічного виведення, що знаходиться в робочій області MatLab. При виборі команди From disk з'явиться діалогове вікно, у якому необхідно вказати ім'я файлу системи нечіткого логічного виведення. Файли систем нечіткого логічного виведення мають розширення .fis . Завантажити систему нечіткого логічного виведення з

диска можна також натисканням Ctrl+N чи командою fuzzy FIS_name, де FIS_name – ім'я файлу системи нечіткого логічного виведення.

При виборі команди Export з'являться дві альтернативи To Workspace... і To disk, що дозволяють скопіювати систему нечіткого логічного виведення в робочу область MatLab і на диск, відповідно. При виборі команди To Workspace... з'явиться діалогове вікно, у якому необхідно вказати ідентифікатор системи нечіткого логічного виведення, під яким вона буде збережена в робочій області MatLab. При виборі команди To disk з'явиться діалогове вікно, у якому необхідно вказати ім'я файлу системи нечіткого логічного виведення. Скопіювати систему нечіткого логічного виведення в робочу область і на диск можна також натисканням Ctrl+T і Ctrl+S, відповідно.

Команда Print дозволяє вивести на принтер копію графічного вікна. Печатка можлива також по натисканню Ctrl+P.

Команда Close закриває графічне вікно. Закриття графічного вікна відбувається по натисканню Ctrl+W чи однократного щиглика лівої кнопки миші по кнопці Close.

Меню Edit:

Команда Undo скасовує раніше зроблену дію. Виконується також по натисканню Ctrl+Z.

Команда Add Variable... дозволяє додати в систему нечіткого логічного виведення ще одну змінну. При виборі цієї команди з'являться дві альтернативи Input і Output, що дозволяють додати вхідну і вихідну змінну, відповідно.

Команда Remove Selected Variable видаляє поточну змінну із системи. Ознакою поточної змінної є червона окантовка її прямокутника. Призначення поточної змінної відбувається за допомогою однократного щиглика лівої

кнопки миші по її прямокутнику. Видалити поточну змінну можна також за допомогою натискання Ctrl+X.

Команда Membership Function... відкриває редактор функцій приналежностей. Ця команда може бути також виконана натисканням Ctrl+2.

Команда Rules…відкриває редактор бази знань. Ця команда може бути також виконана натисканням Ctrl+3.

Меню View – це загальне меню для всіх GUI-модулів, використовуваних із системами нечіткого логічного виведення. Дане меню дозволяє відкрити вікно візуалізації нечіткого логічного виведення (команда Rules або натискання клавіш Ctrl+5) і вікно виведення поверхні “вхід-вихід”, що відповідає системі нечіткого логічного виведення (команда Surface або натискання клавіш Ctrl+6).

Меню And Method - це меню дозволяє установити наступні реалізації логічної операції "ТА": min – мінімум; prod – множення.

Користувач також має можливість установити власну реалізацію операції "ТА". Для цього необхідно вибрати команду Custom…і в графічному вікні, що з'явилося, надрукувати ім'я функції, що реалізує цю операцію.

Меню Or Method - це меню дозволяє установити наступні реалізації логічної операції "АБО": max – множення; probor - імовірнісне "АБО".

Користувач також має можливість установити власну реалізацію операції "АБО". Для цього необхідно вибрати команду Custom… і в графічному вікні, що з'явилося, надрукувати ім'я функції, що реалізує цю операцію.

Меню Implication - це меню дозволяє установити наступні реалізації імплікації: min – мінімум; prod – множення.

Користувач також має можливість установити власну реалізацію імплікації. Для цього необхідно вибрати

команду Custom… і в графічному вікні, що з'явилося, надрукувати ім'я функції, що реалізує цю операцію.

Меню Aggregation - це меню дозволяє установити наступні реалізації операції об'єднання функцій приналежності вихідної змінної: max – максимум; sum – сума; probor - імовірнісне "АБО".

Користувач також має можливість установити власну реалізацію цієї операції. Для цього необхідно вибрати команду Custom…і в графічному вікні, що з'явилося, надрукувати ім'я функції, що реалізує цю операцію

Меню Defuzzification - це меню дозволяє вибрати метод дефазифікації. Для систем типу Мамдані запрограмовані наступні методи: centroid – центр ваги; bisector –медіана; lom – найбільший з максимумів; som – найменший з максимумів; mom – середнє з максимумів. Для систем типу Сугено запрограмовані наступні методи: wtaver – зважене середнє; wtsum – зважена сума.

Користувач також має можливість установити власний метод дефазифікації. Для цього необхідно вибрати команду Custom…і в графічному вікні, що з'явилося, надрукувати ім'я функції, що реалізує цю операцію.

Редактор функцій приналежності

Редактор функцій приналежності (Membership Function Editor) призначений для завдання наступної інформації про терми-множини вхідних і вихідних змінних: кількість термів; найменування термів; тип і параметри функцій приналежності, що необхідні для представлення лінгвістичних термів у вигляді нечітких множин.

Редактор функцій приналежності може бути викликаний з будь-якого GUI-модуля, використовуваного із системами нечіткого логічного виведення, командою

Membership Functions... меню Edit або натисканням клавіш Ctrl+2.

У FIS-редакторі відкрити редактор функцій приналежності можна також подвійним щигликом лівою кнопкою миші по полю вхідної або вихідної змінних. Загальний вид редактора функцій приналежності з указівкою функціонального призначення основних полів графічного вікна приведений на рис. 4.4.

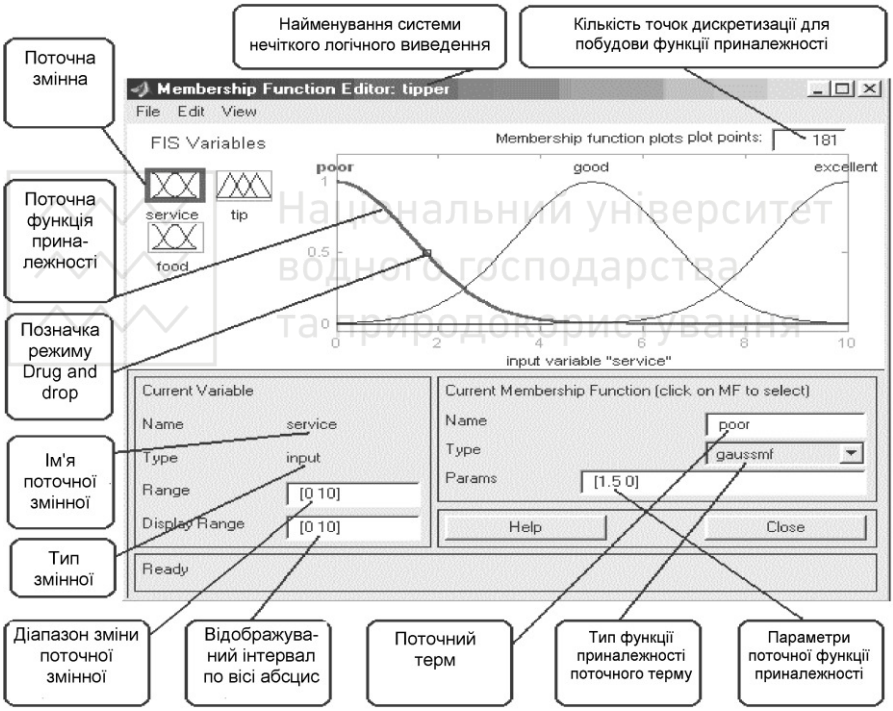

Рис. 4.4. Редактор функцій приналежності

У нижній частині графічного вікна розташовані кнопки Help і Close, що дозволяють викликати вікно довідки і закрити редактор, відповідно.

Редактор функцій приналежності містить чотири меню – File, Edit, View, Type і чотири вікна введення інформації – Range, Display Range, Name і Params. Ці чотири вікна призначені для завдання діапазону зміни поточної змінної, діапазону виведення функцій приналежності, найменування поточного лінгвістичного терму і параметрів його функції приналежності, відповідно.

Параметри функції приналежності можна підбирати й у графічному режимі, шляхом зміни форми функції приналежності за допомогою технології “Drug and drop”. Для цього необхідно позиціонувати курсор миші на знаку режиму “Drug and drop”, натиснути на ліву кнопку миші і не відпускаючи її змінювати форму функції приналежності. Параметри функції приналежності будуть перераховуватися автоматично.

Меню Edit:

Команда Undo скасовує раніше зроблену дію. Виконується також по натисканню Ctrl+Z.

Команда Add MFs…дозволяє додати терми в терми-множину, використовувана для лінгвістичної оцінки поточної змінної. При виборі цієї команди з'явиться діалогове вікно, у якому необхідно вибрати тип функції приналежності і кількість термів. Значення параметрів функцій приналежності будуть встановлені автоматично таким чином, щоб рівномірно покрити область визначення змінної, заданої у вікні Range. При зміні області визначення у вікні Range параметри функцій приналежності будуть промаштабовані.

Команда Add Custom MF…дозволяє додати одних лінгвістичний терм, функція приналежності якого відрізняється від убудованих. Після вибору цієї команди з'явиться графічне вікно, у якому необхідно надрукувати лінгвістичний терм (поле MF name), ім'я функції

приналежності (поле M-File function name) і параметри функції приналежності (поле Parameter list).

Команда Remove Selected MF видаляє поточний терм із терм-множини поточної змінної. Ознакою поточної змінної є червона окантовка її прямокутника. Ознакою поточного терму є червоний колір його функції приналежності. Для вибору поточного терму необхідно провести позиціонування курсору миші на графіку функції приналежності і зробити щиглик лівою кнопкою миші.

Команда Remove All MFs видаляє всі терми з терм-множини поточної змінної.

Команда FIS Properties…відкриває FIS-редактор. Ця команда може бути також виконана натисканням Ctrl+1.

Команда Rules…відкриває редактор бази знань. Ця команда може бути також виконана натисканням Ctrl+3.

Меню Type дозволяє установити тип функцій приналежності термів, використовуваних для лінгвістичної оцінки поточної змінної.

Редактор бази знань

Редактор бази знань (Rule Editor) призначений для формування і модифікації нечітких правил. Редактор бази знань може бути викликаний з будь-якого GUI-модуля, використовуваного із системами нечіткого логічного виведення, командою Rules…меню Edit або натисканням клавіш Ctrl+3. У FIS-редакторі відкрити редактор бази знань можна також подвійним щигликом лівою кнопкою миші по прямокутнику з назвою системи нечіткого логічного виведення, розташованого в центрі графічного вікна.

Загальний вид редактора бази знань із указівкою функціонального призначення основних полів графічного вікна приведений на рис. 4.5.

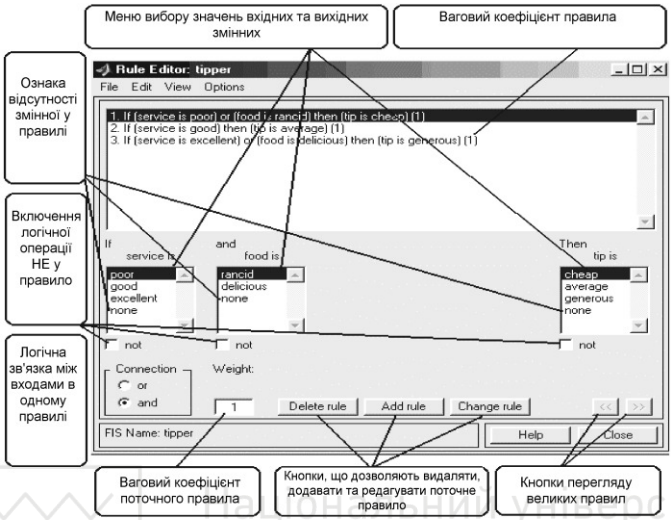

Рис. 4.5. Редактор бази знань

У нижній частині графічного вікна розташовані кнопки Help і Close, що дозволяють викликати вікно довідки і закрити редактор, відповідно.

Редактор функцій приналежності містить чотири системних меню File, Edit, View, Options, меню вибору термів вхідних і вихідних змінних, поля установки логічних операцій ТА, АБО, НЕ і ваг правил, а також кнопки редагування і перегляду правил.

Для введення нового правила в базу знань необхідно за допомогою миші вибрати відповідну комбінацію лінгвістичних термів вхідних і вихідних змінних, установити тип логічного зв'язування (ТА або АБО) між змінними усередині правила, установити чи наявність відсутність логічної операції НЕ для кожної лінгвістичної змінний, увести значення вагового коефіцієнта правила і натиснути кнопку Add Rule. За замовчуванням установлені наступні параметри: логічне зв'язування змінних усередині

правила – ТА; логічна операція НЕ – відсутня; значення вагового коефіцієнта правила – 1.

Можливі випадки, коли істинність правила не змінюється при довільної значенні деякої вхідної змінної, тобто ця змінна не впливає на результат нечіткого логічного виведення в даній області факторного простору. Тоді як лінгвістичне значення цієї змінної необхідно установити none.

Для видалення правила з бази знань необхідно зробити однократний щиглик лівою кнопкою миші на цьому правилі та натиснути кнопку Delete Rule.

Для модифікації правила необхідно зробити однократний щиглик лівою кнопкою миші на цьому правилі, потім установити необхідні параметри правила і натиснути кнопку Edit Rule.

Меню Edit:

Команда Undo скасовує раніше зроблену дію. Виконується також по натисканню Ctrl+Z.

Команда FIS Properties…відкриває FIS-редактор. Ця команда може бути також виконана натисканням Ctrl+1.

Команда Membership Function…відкриває редактор функцій приналежностей. Ця команда може бути також виконана натисканням Ctrl+2.

Меню Options дозволяє установити мову і формат правил бази знань. При виборі команди Language з'явиться список мов English (Англійська), Deutsch (Німецька), Francais (Французька), з якого необхідно вибрати одну.

При виборі команди Format з'явиться список можливих форматів правил бази знань: Verbose - лінгвістичний; Symbolic – логічний; Indexed – індексований.

Візуалізація нечіткого логічного виведення

Візуалізація нечіткого логічного виведення здійснюється за допомогою GUI-модуля *Rule Viewer*. Цей

модуль дозволяє проілюструвати хід логічного виведення за кожним правилом, одержання результуючої нечіткої множини і виконання процедури дефазифікації. Rule Viewer може бути викликаний з будь-якого GUI-модуля, використовуваного із системами нечіткого логічного виведення, командою View rules … меню View чи натисканням клавіш Ctrl+4. Вид Rule Viewer для системи логічного виведення tipper із указівкою функціонального призначення основних полів графічного вікна приведений на рис. 4.6.

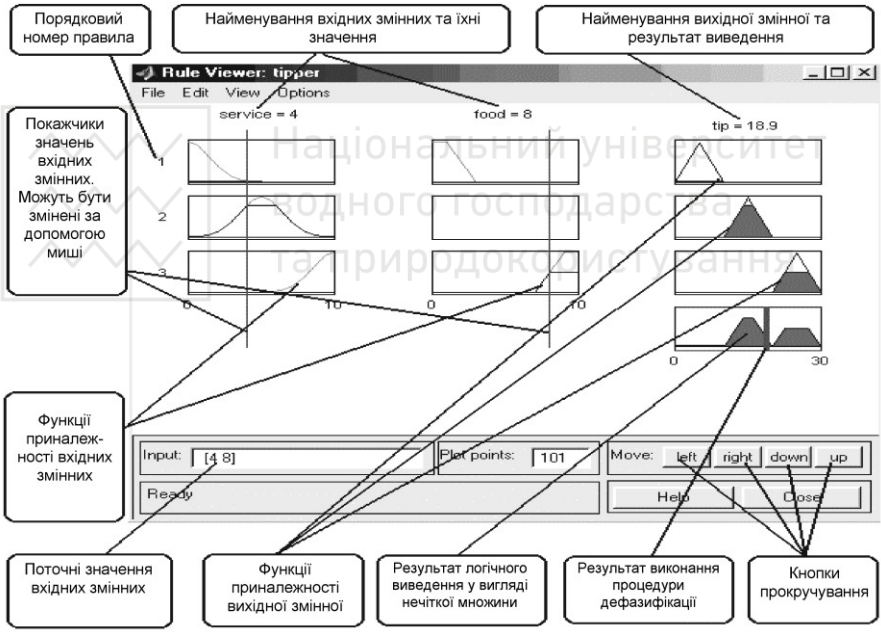

Рис. 4.6. Візуалізація логічного виведення для системи tipper за допомогою Rule Viewer

Rule Viewer містить чотири меню – File, Edit, View, Options, два поля уведення інформації – Input і Plot points та кнопки прокручування зображення вліво - вправо (left-

right), униз (up-down). У нижній частині графічного вікна розташовані також кнопки Help і Close, що дозволяють викликати вікно довідки і закрити редактор, відповідно. Кожне правило бази знань представляється у виді послідовності горизонтально розташованих прямокутників. При цьому перші два прямокутники відображають функції приналежностей термів посилки правила (Якщо-частина правила), а останній третій прямокутник відповідає функції приналежності терму-наслідку вихідної змінної (То-частина правила).

Порожній прямокутник у візуалізації другого правила означає, що в цьому правилі посилка за змінною food відсутня (food is none). Жовте заливання графіків функцій приналежностей вхідних змінних указує наскільки значення входів, відповідають термам даного правила. Для виведення правила у форматі Rule Editor необхідно зробити однократний щиглик лівої кнопки миші по номері відповідного правила. У цьому випадку зазначене правило буде виведено в нижній частині графічного вікна.

Блакитне заливання графіка функції приналежності вихідної змінної являє собою результат логічного виведення у вигляді нечіткої множини за даним правилом. Результуючу нечітку множину, що відповідає логічному виведенню за всіма правилами, показано в нижньому прямокутнику останнього стовпця графічного вікна. У цьому ж прямокутнику червона вертикальна лінія відповідає чіткому значенню логічного виведення, отриманого в результаті дефазифікації.

Уведення значень вхідних змінних може здійснюватися двома способами: шляхом уведення чисельних значень у поле Input; за допомогою миші, шляхом переміщення ліній-покажчиків червоного кольору.

В останньому випадку необхідно позиціонувати курсор миші на червоній вертикальній лінії, натиснути на ліву

кнопку миші і не відпускаючи неї перемістити покажчик на потрібну позицію. Нове чисельне значення відповідної вхідної змінної буде перелічено автоматично і виведене у вікно Input.

У поле Plot points задається кількість крапок дискретизації для побудови графіків функцій приналежності. Значення за замовчуванням – 101.

Меню Edit:

Команда FIS Properties…відкриває FIS-редактор. Ця команда може бути також виконана натисканням Ctrl+1.

Команда Membership Functions…відкриває редактор функцій приналежностей. Ця команда може бути також виконана натисканням Ctrl+2.

Команда Rules…відкриває редактор бази знань. Ця команда може бути також виконана натисканням Ctrl+3.

Меню Options містить тільки одну команду Format, що дозволяє установити один з наступних форматів виведення обраного правило в нижній частині графічного вікна: Verbose - лінгвістичний; Symbolic – логічний; Indexed – індексований.

ANFIS-редактор

ANFIS є абревіатурою Adaptive Neuro-Fuzzy Inference System – адаптивна нейро-нечітка система. ANFIS-редактор дозволяє автоматично синтезувати з експериментальних даних нейро-нечіткі мережі. Нейро-нечітку мережу можна розглядати як один з різновидів систем нечіткого логічного виведення типу Сугено. При цьому функції приналежності синтезованих систем налагоджено (навчено) так, щоб мінімізувати відхилення між результатами нечіткого моделювання й експериментальних даних.

Завантаження ANFIS-редактора здійснюється за командою anfisedit. У результаті виконання цієї команди

з'явиться графічне вікно, зображене на рис. 4.7. На цьому ж рисунку зазначено функціональні області ANFIS-редактора, опис яких приведено нижче.

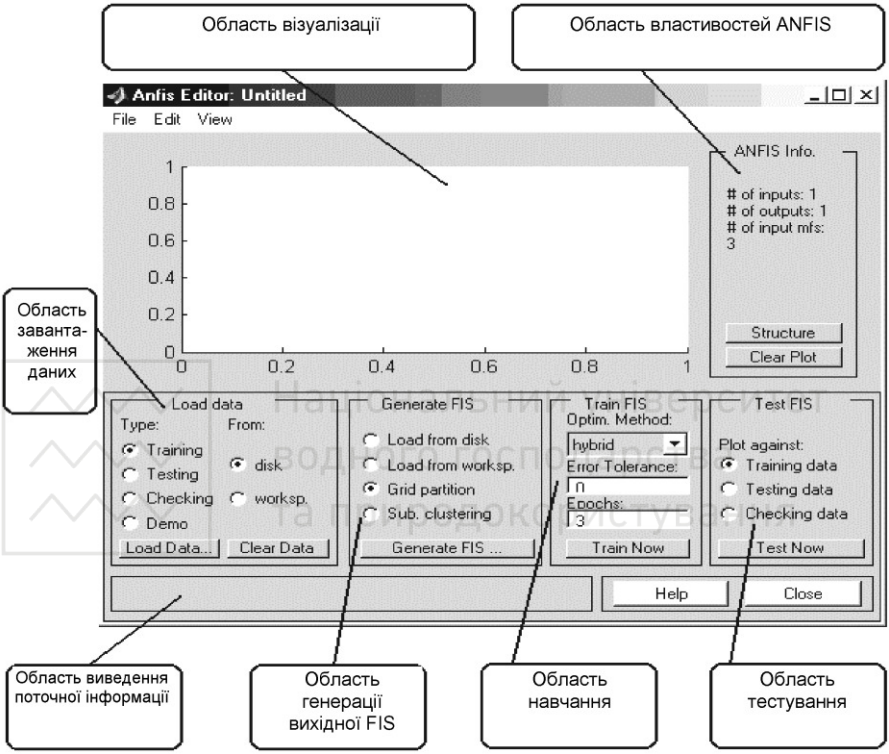

Рис. 4.7. Основне вікно ANFIS-редактора

ANFIS-редактор містить 3 верхніх меню – File, Edit і View, область візуалізації, область властивостей ANFIS, область завантаження даних, область генерування вихідної системи нечіткого логічного виведення, область навчання, область тестування, область виведення поточної інформації, а також кнопки Help і Close, що дозволяють викликати вікно довідки і закрити ANFIS-редактор, відповідно.

Меню Edit:

Команда Undo скасовує раніше зроблену дію. Виконується також по натисканню Ctrl+Z.

Команда FIS Properties…відкриває FIS-редактор. Ця команда може бути також виконана натисканням Ctrl+1.

Команда Membership Functions…відкриває редактор функцій приналежності. Ця команда може бути також виконана натисканням Ctrl+2.

Команда Rules…відкриває редактор бази знань. Ця команда може бути також виконана натисканням Ctrl+3.

Команда Anfis…відкриває ANFIS-редактор. Ця команда може бути також виконана натисканням Ctrl+3. Помітимо, що дана команда, запущена з ANFIS-редактора не приводить до виконання яких-небудь дій, так цей редактор уже відкритий. Однак, у меню Edit інших GUI-модулів, використовуваних із системами нечіткого логічного виведення, додається команда Anfis…,що дозволяє відкрити ANFIS-редактор з цих модулів.

Область візуалізації містить два типи інформації: при навчанні системи – крива навчання у виді графіка залежності помилки навчання від порядкового номера ітерації; при завантаженні даних і тестуванні системи – експериментальні дані і результати моделювання.

Експериментальні дані і результати моделювання виводяться у виді множини крапок у двовимірному просторі. При цьому по вісі абцис відкладається порядковий номер рядка даних у вибірці (навчальної, тестової або контрольної), а по осі ординат - значення вихідної змінної для даного рядка вибірки. Використовуються наступні маркери: блакитна крапка (**.**) – тестова вибірка; блакитна окружність (o) – навчальна вибірка; блакитний плюс (+) – контрольна вибірка; червона зірочка (*) – результати моделювання.

В *області властивостей* ANFIS (ANFIS info) виводиться інформація про кількість вхідних і вихідних змінних, про кількість функцій приналежності для кожної вхідний змінний, а також про кількість рядків у вибірках. У цій області розташовані дві кнопки Structure і Clear Plot.

Натискання кнопки Structure відчиняє нове графічне вікно, у якому система нечіткого логічного виведення представляє у виді нейро-нечіткої мережі. Натискання кнопки Clear Plot дозволяє очистити область візуалізації.

В *області завантаження даних* (Load data) розташовані: меню вибору типу даних (Type), що містить альтернативи (Traning - навчальна вибірка; Testing - тестова вибірка; Checking - контрольна вибірка; Demo - демонстраційний приклад); меню вибору джерела даних (From), що містить альтернативи (disk – диск; worksp. - робоча область MatLab); кнопка завантаження даних Load Data…, по натисканню якої з'являється діалогове вікно вибору файлу, якщо завантаження даних відбувається з диска, або вікно введення ідентифікатора вибірки, якщо завантаження даних походить з робочої області; кнопка очищення даних Clear Data.

В *області генерування* (Generate FIS) розташовані меню вибору способу створення вихідної системи нечіткого логічного виведення. Меню містить наступні альтернативи: Load from disk – завантаження системи з диска; Load from worksp. – завантаження системи з робочої області MatLab; Grid partition - генерування системи по методу ґрат (без кластеризації); Sub. clustering – генерування системи за методом субкластеризації.

В області також розташована кнопка Generate, по натисканню якої генерується вихідна система нечіткого логічного виведення.

При виборі Load from disk з'являється стандартне діалогове вікно відкриття файлу.

При виборі Load from worksp. з'являється стандартне діалогове вікно введення ідентифікатора системи нечіткого логічного виведення.

При виборі Grid partition з'являється вікно введення параметрів методу ґрат, у якому потрібно вказати кількість термів для кожен вхідний змінної і тип функцій приналежності для вхідних і вихідної змінних.

При виборі Sub. clustering з'являється вікно введення наступних параметрів методу субкластеризації: Range of influence – рівні впливу вхідних змінних; Squash factor – коефіцієнт пригнічення; Accept ratio – коефіцієнт, що встановлює у скільки разів потенціал даної точки повинний бути вище потенціалу центра першого кластера для того, щоб центром одного з кластерів була призначена розглянута точка; Reject ratio - коефіцієнт, що встановлює у скількох разів потенціал даної точки повинний бути нижче потенціалу центра першого кластера, щоб розглянута точка була виключена з можливих центрів кластерів.

В *області навчання* (Train FIS) розташовані меню вибору методу оптимізації (Optim. method), поле завдання необхідної точності навчання (Error tolerance), поле завдання кількості ітерацій навчання (Epochs) і кнопка Train Now, натискання якого запускає режим навчання. Проміжні результати навчання виводяться в область візуалізації й у робочу область MatLab. У ANFIS-редакторі реалізовані два методи навчання: backprop – метод зворотного поширення помилки, заснований на ідеях методу найшвидшого спуска; hybrid – гібридний метод, що поєднує метод зворотного поширення помилки з методом найменших квадратів.

В *області тестування* (Test FIS) розташовані меню вибору вибірки і кнопка Test Now, по натисканню по який

відбувається тестування нечіткої системи з виведення результатів в область візуалізації.

Область виведення поточної інформації: у цій області виводиться найбільш істотна поточна інформація, наприклад, повідомлення про закінчення виконанні операцій, значення помилки чи навчання тестування і т.п.

Функції Fuzzy Logic Toolbox

Функції, що входять до модуля Fuzzy Logic Toolbox можуть бути викликані з командного рядка. Для отримання переліку функцій слід ввести команду: help fuzzy. Наведемо короткий огляд функцій модуля Fuzzy Logic.

Редактори з графічним інтерфейсом користувача: anfisedit – інструмент для навчання та тестування ANFIS; findcluster - інструмент для кластеризації; fuzzy - базовий редактор FIS; mfedit – редактор функцій приналежності; ruleedit – редактор та аналізатор правил; ruleview – демонстратор правил та діаграм нечіткого виведення; surfview - демонстратор вихідної поверхні.

Функції приналежності: dsigmf, gauss2mf, gaussmf, gbellmf, pimf, psigmf, smf, sigmf, trapmf, trimf, zmf.

Модуль Fuzzy Logic Toolbox пакету MATLAB включає 11 убудованих функцій приналежності, що використовують такі основні функції: кусочно-лінійну; гаусівській розподіл; сигмоїдну криву; квадратичну та кубічну криві.

Для зручності імена всіх убудованих функцій приналежності закінчуються на mf. Виклик функції приналежності здійснюється в такий спосіб: *namemf(x, params),* де namemf – найменування функції приналежності; x – вектор, для координат якого необхідно розрахувати значення функції приналежності; params – вектор параметрів функції приналежності.

Найпростіші функції приналежності *трикутна* (trimf) і *трапецієподібна* (trapmf) формуються з використанням кусочно-лінійної апроксимації. Трапецієподібна функція приналежності є узагальненням трикутної, вона дозволяє задавати ядро нечіткої множини у виді інтервалу. У випадку трапецієподібної функції приналежності можлива наступна зручна інтерпретація: ядро нечіткої множини – оптимістична оцінка; носій нечіткої множини – песимістична оцінка.

Дві функції приналежності – *симетрична гаусівська* (gaussmf) і *двостороння гаусівська* (gaussmf) формуються з використанням гаусівського розподілу. Функція gaussmf дозволяє задавати асиметричні функції приналежності. Узагальнена колоколообразна функція приналежності (gbellmf) за своєю формою схожа на гаусівські. Ці функції приналежності часто використовуються в нечітких системах, тому що на всій області визначення вони є гладкими і приймають ненульові значення.

Функції приналежності sigmf, dsigmf, psigmf засновані на використанні *сигмоїдної кривої*. Ці функції дозволяють формувати функції приналежності, значення яких починаючи з деякого значення аргументу і до +(-) ∞ рівні 1. Такі функції зручні для завдання лінгвістичних термів типу «високий» або «низький».

Поліноміальна апроксимація застосовується при формуванні функцій zmf, pimf і smf, графічні зображення яких схожі на функції sigmf, dsigmf, psigmf, відповідно.

У Fuzzy Logic Toolbox передбачена можливість для користувача створення власної функції приналежності. Для цього необхідно створити m-функцію, що містить два вхідних аргументи – вектор, для координат якого необхідно розрахувати значення функції приналежності і вектор параметрів функції приналежності. Вихідним

аргументом функції повинний бути вектор ступенів приналежності.

Функції FIS: addmf – додає функцію приналежності до FIS; addrule – додає правило до FIS; addvar - додає змінну до FIS; defuzz – дефузифікує функцію приналежності; evalfis - здійснює обчислення нечіткого виведення; evalmf - обчислює функцію приналежності; gensurf – генерує поверхню виходу FIS; getfis - повертає властивості нечіткої системи; mf2mf - транслює параметри між функціями приналежності; newfis - створює нову FIS; parsrule – аналізує нечіткі правила; plotfis – показує діаграму "вхід-вихід" для FIS; plotmf – показує усі функції приналежності для однієї змінної; readfis – завантажує FIS з диску; rmmf – видаляє функцію приналежності з FIS; rmvar – видаляє змінну з FIS; setfis – встановлює властивості нечіткої системи; showfis – показує анотовану FIS; showrule – відображує правила FIS; writefis – зберігає FIS на диску.

Функція output=evalfis(input, fis, numofpoints). виконує обчислення для вибірки екземплярів за допомогою вказаної нейро-нечіткої мережі. Результатом є обчислені виходи функції, яку апроксимує нейро-нечітка мережа. Аргументи та результат функції: input – входи обчислюваної вибірки; fis – нейро-нечітка мережа; numofpoints – кількість точок для проведення дефазифікації (рекомендується брати значення не менше 100, зменшення цього параметру прискорює процес обчислень, але зменшує точність); output – обчислені виходи;

Вибірки даних та нейро-нечіткі мережі, з якими працюють описані функції повинні зберігатися в робочій області MATLAB (в оперативній пам’яті). В стовпцях матриць, що представляють собою вибірки, зберігаються

значення входів (ознак) та виходів, в рядках – екземпляри вибірки.

Для завантаження вибірок з диску або запису на диск необхідно використовувати команду MATLAB: s=load(filename), яка. завантажує вміст файлу в змінну s. Функція save(filename, s). зберігає змінну s у файл.

Для можливості використання створеної нейро-нечіткої мережі в наступних сеансах роботи або використання мережі з попередніх слід використовувати такі функції модуля Fuzzy Logic Toolbox: writefis(fis,filename) - зберігає нейро-нечітку мережу fis в файл; fis=readfis(filename) – завантажує з файлу нейро-нечітку мережу в змінну fis.

Передові технології: anfis – функція навчання для системи Сугено; функції кластер-аналізу: fcm, genfis1, genfis2; subclust.

Різні функції: convertfis – перетворює нечітку матрицю структури версії 1.0 у матрицю структури версії 2.0; discfis - дискретизує FIS; evalmmf - використовується для обчислення множинних функцій приналежності; fstrvcat- поєднує матриці різного розміру; fuzarith – функція нечіткої арифметики; findrow – шукає рядки матриці, що відповідають вхідному рядку; genparam – генерує початкові параметри передумов для навчання ANFIS; probor – імовірнісне АБО; sugmax – максимальний вихідний діапазон для системи Сугено.

Функція C = fuzarith(X, A, B, OPERATOR) реалізує базові операції нечіткої логіки та повертає нечітку множину C як результат застосування оператора OPERATOR до нечітких множин A та B з універсальної множини X. Змінні A, B та X мають бути векторами однакової розмірності. OPERATOR має бути одним з рядків: 'sum' – сума, 'sub' – вирахування, 'prod' – добуток, 'div' – ділення. Нечітка множина C, яка повертається, є

вектор-стовпцем тієї є довжини, що й A та B. Зауважимо, що ця функція використовує інтервальну арифметику та передбачає, що: A та B є опуклими нечіткими множинами; функції приналежності для A та B поза X є нулем.

Допоміжні функції графічного користувальницького інтерфейсу: cmfdlg – додає діалог вибору функцій приналежності; cmthdlg – додає діалог вибору методу виведення; fisgui – дискрипторне посилання на інтерфейсні засоби модуля Fuzzy Logic Toolbox; gfmfdlg – генерує FIS з використанням діалогу методу ґрат; mfdlg – додає діалог функції приналежності; mfdrag – перетягування функції приналежності за допомогою миші; popundo – відновлює зі стеку останні зміни (відміняє останні дії); pushundo- передає поточні дані у стек відновлення; savedlg – діалог запису перед закриттям; statmsg – зображує повідомлення у полі статусу; updtfis – оновлює засоби графічного інтерфейсу Fuzzy Logic Toolbox; wsdlg – діалог "відкриття з" / "збереження до" робочої області.

Завдання до роботи

1. Ознайомитися з конспектом лекцій та рекомендованою літературою
2. Обґрунтовано сформувати набір даних для обробки та аналізу.
3. Використовуючи рекомендоване програмне забезпечення здійснити обробку набору даних з метою побудови нейро-нечітких мереж.
4. Спробувати використати різні алгоритми кластеризації, різні кількості функцій приналежності для входів, різні кількості циклів навчання та різні алгоритми навчання.
5. Використати побудовані нейро-нечіткі мережі для прийняття рішень на конкретному прикладі.
6. Проаналізувати отримані результати та відповісти

на питання: який алгоритм кластер-аналізу призводить до отримання мережі меншої складності; як впливає задана кількість циклів навчання на точність навчання; як вливає задана точність навчання на тривалість навчання; які вимоги мають пред'являтися до навчальної вибірки та як це вплине на процес навчання?

7. Оформити звіт з роботи.
8. Відповісти на контрольні питання.

Зміст звіту

1. Тема та мета роботи.
2. Короткі теоретичні відомості.
3. Обґрунтовано обраний набір даних для обробки (якщо він великий – навести фрагмент).
4. Опис процесу використання програмного забезпечення для обробки набору даних, що має бути ілюстрований зображеннями екранних форм.
5. Тексти програм.
6. Результати роботи програмного забезпечення (зображення структури отриманої мережі, її ваги, результати прийняття рішень, інші характеристики).
7. Висновки, що містять відповіді на контрольні запитання, а також відображують результати виконання роботи та їх критичний аналіз.

Контрольні запитання

1. Порівняйте властивості нечітких систем та нейронних мереж.
2. Що таке нейро-нечітка мережа?
3. Які властивості мають нейро-нечіткі мережі?
4. Які існують типи нейро-нечітких мереж?
5. Як формують базу знань нейро-нечіткої мережі?

6. З яких елементів складаються нейро-нечіткі мережі?

7. У чому полягають особливості паралельних нейро-нечітких мереж?

8. Що таке нейро-нечтітка асоціативна пам’ять Коско?

9. Як відбувається виділення нечітких правил за допомогою карт, що самоорганізуються?

10. Опишіть особливості систем, що здатні навчати нечіткі множини.

11.У чому полягають особливості конкурентних нейро-нечітких систем?

Рекомендована література

1. Алиев Р. А., Абдикеев Н. М., Шахназаров М. М. Производственные системы с искусственным интеллектом. М. : Радио и связь, 1990. 264 с.

2. Архангельский В. И., Богаенко И. Н., Грабовский Г. Г., Рюмшин Н. А. Системы фуцци-управления. К. : Тэхника, 1997. 208 с.

3. Беллман Р., Заде Л. Принятие решений в расплывчатых условиях *Вопросы анализа и процедуры принятия решений*. М. : Мир, 1976. С.172–215.

4. Бокша В. В., Силов В. Б. Нечеткое целевое управление системами с заданным конечным состоянием. *Автоматика*. 1985. № 3. С.3–8.

5. Борисов А. Н., Алексеев А. В., Крумберг О. А. и др. Модели принятия решений на основе лингвистической переменной. Рига : Зинатне, 1982. 256 с.

6. Борисов А. Н., Алексеев А. В., Меркурьева Г. В. и др. Обработка нечеткой информации в системах принятия решений. М. : Радио и связь. 1989. 304 с.

7. Борисов А. Н., Крумберг О. А., Федоров И. П. Принятие решений на основе нечетких моделей. Примеры использования. Рига : Зинатне, 1990. 184 с.

8. Гусев Л. А., Смирнова И. М. Размытые множества. Теория и приложения (обзор). *Автоматика и телемеханика*. 1973. № 5. С.66–85.

9. Дьяконов В. MATLAB 6: учебный курс. СПб. : Питер, 2001. 592 с.

10. Заде Л. А. Основы нового подхода к анализу сложных систем и процессов принятия решений. *Математика сегодня*. М. : Знание, 1974. С. 5–49.