Як критерій розпізнання $F(X,\overline{X}_j)$ в цьому випадку найкращим виявився баєсовський, що досить легко реалізується, оскільки оптимальний набір ознак n склав п'ять значень:

$$F(X,\overline{X}_j)=ln|C_j|+(X-\overline{X}_j)'C_j^{-1}(X-\overline{X}_j),$$

де $\overline{X}_j$ та C_j – відповідно n-вимірний вектор математичного сподівання та коваріаційна матриця ознак навчальних образів j-го класу.

Класифікація – за мінімумом критерію $F(X,\overline{X}_j)$.

Отже, показані можливості різних методів обробки спектрометричних даних та перспективність процедур з використанням критеріїв теорії розпізнання образів, на основі формалізованих правил дають змогу швидко обробляти величезні масиви експериментальних даних на комп'ютерах з метою якісного і навіть кількісного експрес-аналізу складу чи стану об'єктів. При цьому найбільш формалізованою та простою є методика вибору оптимального набору ознак за значеннями парних похідних в точках перегину.

Список літератури

1. *Антипова-Коротаева И.Н., Казанова Н.Н.* Математическое разложение сложных спектральных контуров на компоненты с частично известными параметрами // Журн. прикладной спектроскопии. – 1972. – Т.16. № 5. – С. 855–858.

2. *Гречушников В.Н., Калинкина И.Н., Старостина Л.С.* Разложение перекрытых спектральных линий методом Фур'е // Журн. прикладной спектроскопии. – 1975. – Т.23. №6. – С.1059–1066.

3. *Фок М.В.* Разделение сложных спектров на индивидуальные полосы при помощи обобщенного метода Аленцева // Тр. ФИАН. – 1972. – Т.59. – С. 3–10.

4. *Allen L.C., Gladney H.M.* Resolution enhancement for spectra of chemical and physical interest // The jornal of chemical physics. – 1964. – Vol. 40. № 11. – P. 3125–3143.

5. *Marrey J.R.* On determining spectral peak positions from composite spectra with a digital computer // Analytical Chemistry. – 1968. – Vol. 40. – P. 905–914.

6. *Кондратьев К.Я., Васильев О.В., Федченко П.П.* Опыт распознавания почв по их спектрам отражения // Почвоведение. – 1983. – № 4. – С. 5–17.

7. *Жук Л.А., Кочубей С.М., Сураев В.Ф.* Автоматизация экспресс-анализа количественного состава многокомпонентных объектов // Механизация и автоматизация управления. – 1989. – № 2. – С. 24–27.

Стаття надійшла до редакції 30.03.02.

УДК 004.032

Ю.М. Мінаєв, д-р техн. наук, проф.,
Бенамур Лієс,
М.М. Гузій, канд. техн. наук, доц.,
В.В. Давиденко, асп.

ІДЕНТИФІКАЦІЯ АТАК НА КОМП'ЮТЕРНІ МЕРЕЖІ НА ПІДСТАВІ НЕЙРОМЕРЕЖНИХ ТЕХНОЛОГІЙ У СИСТЕМІ МОДЕЛЮВАННЯ MATLAB/SIMULINK

Розглянуто методику побудови нейромережі для ідентифікації атак на комп'ютерні мережі за допомогою засобів пакету математичного моделювання MatLab/Neural Network/ Simulink. Наведено приклад, що ілюструє ефективність нейромережної технології.

Проблема виявлення атак на комп'ютерні мережі (КМ) є домінуючою в теорії та практиці захисту інформації. Сучасні системи виявлення атак IDS (Intrusion Detecting System) працюють на двох рівнях залежно від того, до якої інформації існує доступ. Загальним у цих підходах є пошук відповідних ознак (сигнатур), комбінації цих ознак (шаблонів), які вказують на ворожі дії або на їх підозру. Якщо пошук цих сигнатур та шаблонів виконується на рівні мережного

трафіку, то IDS працюють на мережному рівні, якщо в системному журналі або в журналі додатків – на системному рівні. Зазвичайно, найбільш ефективною буде технологія, яка працює, враховуючи обидва рівні.

Однією із сучасних технологій, від якої чекають певних досягнень, є нейромережна технологія. У роботі Кеннеді [1] наводиться приклад виявлення наявності атаки на КМ за допомогою нейромережних технологій. Досвід показує, що для рівня правильної ідентифікації атак доцільно застосовувати декілька нейромережних технологій, зокрема чіткі нейромережі і нечіткі (fuzzy) нейронні мережі. При цьому ці технології повинні застосовуватися не альтернативно, але паралельно.

Для ідентифікації атак у нейромережному логічному базисі доцільно вибрати ті ознаки, за якими система Internet Scanner виконує пошук «слабких» місць в існуючих системах захисту. Зокрема Internet Scanner може готувати звіти за такими ознаками: скануючі пристрої, мережний сервіс, номера портів, імена користувачів, слабкі місця, IP-адреси. З метою перевірки технологічності використання нейромережних моделей для ідентифікації атак були розглянуті дві моделі: багатошарова нейронна мережа (персептрон), робота якої організована за методом Back Propagation; нечітка нейромережа, яка дозволяє виконати нечітку кластеризацію.

Персептрон є практично ідеальним класифікатором і являє собою структуру, яка складається з персептронних нейронів (рис. 1). Передатна функція в такій структурі, як правило, вибирається пороговою. На рис. 2 наведено архітектуру одношарового персептрона, процедуру навчання якого можна уявити у вигляді алгоритму

$$\mathbf{W}^{new} = \mathbf{W}^{old} + \mathbf{e}\mathbf{p}^{T};$$

$$\mathbf{b}^{new} = \mathbf{b}^{old} + \mathbf{e},$$

де ***old*** , ***new*** – старе та нове значення матриці ваг W та зміщень b; $e = t - a$.

На рис. 3 графічно показано процедуру поділу бінарного простору на дві частини за допомогою головного розв'язувального правила, що реалізує персептрон.

Детальні відомості про роботу персептрона і технологію навчання за методом Back Propagation можна знайти в роботах [2; 3].

Постановку задачі ідентифікації атак на КМ будемо розглядати як задачу класифікації, реалізація якої передбачена за допомогою одношарового або багатошарового персептрона [2]. Припустимо, що ми маємо тренувальні дані $(\mathbf{x_1}, \mathbf{c_1})$, $(\mathbf{x_2}, \mathbf{c_2})$,…,$(\mathbf{x_n}, \mathbf{c_n})$, які вміщують множину $X_t = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_{Nt}}\}N_t$ тренувальних спостережень у n-вимірному просторі ($\mathbf{x_j} \in R^n$, $n \geq 2$) і їхній пов'язаний клас-індикатор вектор $\mathbf{c_j}$, $j = 1,2\ldots N_t$. Обмежимося розглядом двокласової проблеми («атака на КМ», «відсутність атаки»), хоча в реальних умовах це обмеження може бути критичним, оскільки залежно від типу даних (наприклад, різниця між даними складає величину 10^8 ÷ 10^{10}!) вибір ваг матриці зв'язків може бути вкрай ускладненим, і кількість класів доведеться збільшувати. Саме така ситуація має місце при намаганні аналізу трафіка. Постановка задачі вимагає, щоб c_j був двовимірним вектором $(\mathbf{c_j}=(\mathbf{c_1}, \mathbf{c_2})^T)$, який вказує, що $\mathbf{x_j}$ може належати одному з двох класів ω_1 або ω_2.

Компоненти c_{1j}, c_{2j} визначені, як «нуль» або «одиниця» відповідно до клас-належності $\mathbf{x_j}$, тобто $c_{1j}=1$ & $c_{2j}=0$ if $\mathbf{x_j} \in \omega_1$ та $c_{1j}=0$ & $c_{2j}=1$ if $x_j \in \omega_2$. Клас-індикатор вектор $\mathbf{c_j}$ передбачає декомпозицію множин X_t на підмножини відповідно до окремих класів. Через N_{ti} позначимо кількість тренувальних спостережень у класі ω_i. Можна вимагати нормалізацію тренувальних даних x_t, яке часто називають відбілюванням. Подальший виклад використовує ненормалізовані дані.

Через $y(\mathbf{x}; \mathbf{w})$ позначимо відтворення функції багатошарового персептрона (MLP) нейронної мережі для класифікації з вектором $\mathbf{w}$, який вміщує регульовані ваги нейромережі. Тренування мережі виконується мінімізацією середньоквадратичної (MS) похибки

$$E^{MLP}(w)=\frac{1}{N_t}\sum_{j=1}^{N_t}[y(x_j;w)-(c_{1j}-c_{2j})]^2, \qquad (1)$$

де приймач $(c_{1j}–c_{2j}) = 1$ для тренувальних векторів $x_j \in \omega_1$ та $(c_{1j}–c_{2j}) = –1$ (або нуль) для $x_j \in \omega_2$. Подальше поліпшення виразу для $E^{MLP}(\,)$ можливо шляхом регуляризації виразу (1) для зменшення узагальнюючих властивостей нейромережі.

Похибка $E^{MLP}(\mathbf{w})$ нейромережі з нелінійною активаційною функцією на прихованому шарі моделі є суттєво нелінійною функцією. Послідовна мінімізація $E^{MLP}(\,)$ може бути проведеною з використанням ітераційних оптимізаційних алгоритмів, які доступні із середовища MatLab. Головна мета – пошук глобального мінімуму $E^{MLP}(w)$. Найпростіше використання тренувального алгоритму – локальна мінімізація $E^{MLP}(\,)$. Обчислена величина спостереженого мінімуму може бути чистим локальним мінімумом. Розв'язок $\mathbf{w}^*$ строго залежить від стартових (початкових) значень локального оптимізатора. Рекурсивні (евристичні) методи використовуються для пошуку декількох невеликих (наприклад, досягнення величини 10^{-6}) локальних мінімумів, з яких обирається один. Саме на цьому мінімумі фіксується матриця ваг нейромережі, яка потім використовується як класифікатор.

Специфіка постановки задачі полягає в необхідності врахування таких обставин. Задані два вектори $\mathbf{x}^{(1)}=\{x_j^{(1)}\}$, $\mathbf{x}^{(2)}=\{x_j^{(2)}\}$, j=1,9, компоненти яких характеризують трафік, $\mathbf{x}^{(1)} \in \omega_1$, $\mathbf{x}^{(2)} \in \omega_2$, де ω_1,ω_2 – класи ситуації, що відповідають наявності атаки (100 % гарантії) та відсутності атаки. Враховуючи, що для реалізації гарантованого результату цих векторів недостатньо, сформовано тестові вибірки $\mathbf{x}^{(1t)}$ та $\mathbf{x}^{(2t)}$ відповідно до $\mathbf{x}^{(1)}$ та $\mathbf{x}^{(2)}$. Кожна компонента (крім бінарних) тестових вибірок $x_j^{(1t)} \in \mathbf{x}^{(1t)}$ та $x_j^{(2t)} \in \mathbf{x}^{(2t)}$ сформована за принципом

$$x_j^{1t} = x_j \pm 0.15 * x_j \,\mathbf{rand}(). \qquad (2)$$

Схему формування тестових наборів наведено на рис. 4. Реальна відокремлююча поверхня має зону невизначеності. Це зумовлено тим, що в реальних умовах отримати «чисті» значення 0 та 1 на заданому наборі даних (діапазон коливань значень 10^{10}) практично неможливо навіть при довготермінованому тренуванні. Навчання нейромережі вважалося виконаним правильно, якщо для набору $\mathbf{x}^{(1t)}$ отримували 0,95 та 0,05 і для $\mathbf{x}^{(2t)}$ відповідно 0,05 та 0,95, що звичайно призводить до виникнення певної зони невизначеності (5÷10 %). Додаткове підвищення точності класифікації можливе за рахунок використання нових ознак. Однак потрібно при цьому ретельно підрахувати економічну доцільність. Умову вибору можна зробити більш толерантними, а саме – відносити вихід до одного з класів за умови >> 0,5, але це окреме дослідження.

Правильно навчена мережа може правильно класифікувати ситуації за умов, що деякі дані (значення вхідного вектора) можуть бути відсутніми (прихованими), або визначатися на рівні «м'яких» вимірювань, тобто уявлених у вигляді нечітких чисел або змінних. Результати дослідів з «відсутніми» компонентами не наводяться.

Для перевірки ефективності нейромережних технологій сигнатури атак визначалися згідно з роботою [2] і включали такі параметри: $\mathbf{x} = \{ x_i \}$, i =1,9:

– x_1 – **Protocol ID** протокол, пов'язаний з подією (TCP=0, UDP=1, ICMP= =2, unknown=3);
– x_2 – **номер** порту джерела;
– x_3 – **номер** порту хвоста призначення;
– x_4 – **IP-адреса** джерела;
– x_5 – **IP-адреса** приймача;
– x_6 – **ICMP Type тип** ICMP-пакету (Echo Request or Null);
– x_7 – **ICMP Code** кодове поле з ICMP-пакета (None or Null);
– x_8 – **Raw Data Length** довжина даних у пакеті;
– x_9 – **Raw Data** порція даних у пакеті.

Схему нейромережі, яка використовувалася при дослідах, наведено на рис. 5. Мережа персептронного типу мала такі параметри:

– 2 шари;
– 18 нейронів на першому шарі;
– 9 вхідних змінних ($\mathbf{x} = \{ x_i \}$, i=1,9);
– 2 нейрона на вихідному шарі.

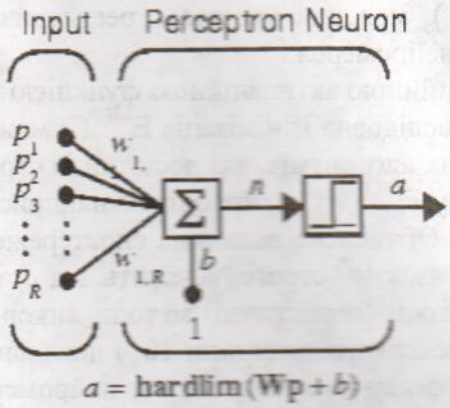

Рис.1. Загальний вигляд нейрона персептронного типу

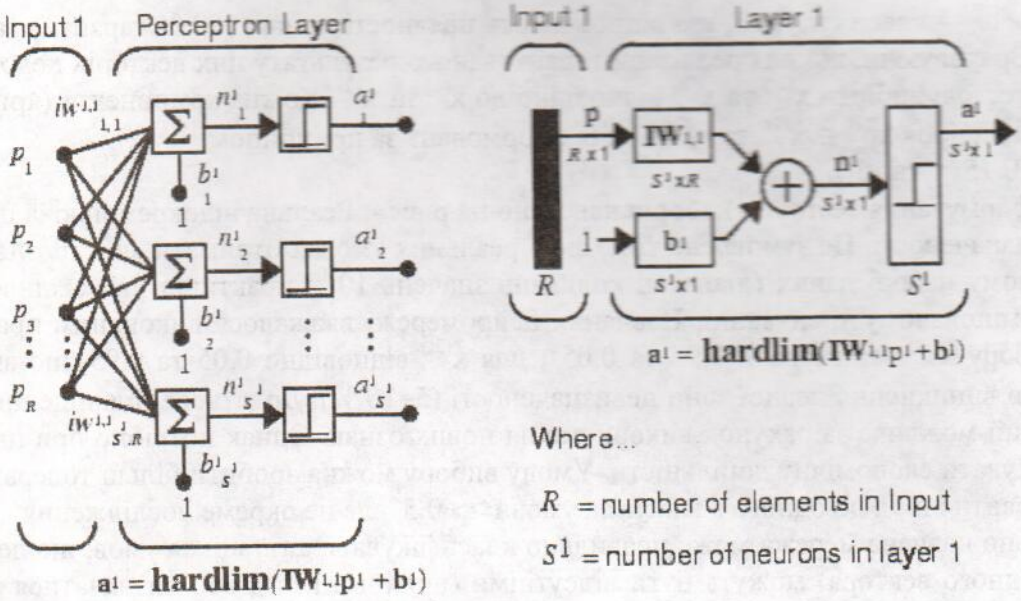

Рис. 2. Загальна схема нейронної мережі персептронного типу та її уявлення за допомогою нотації MatLab

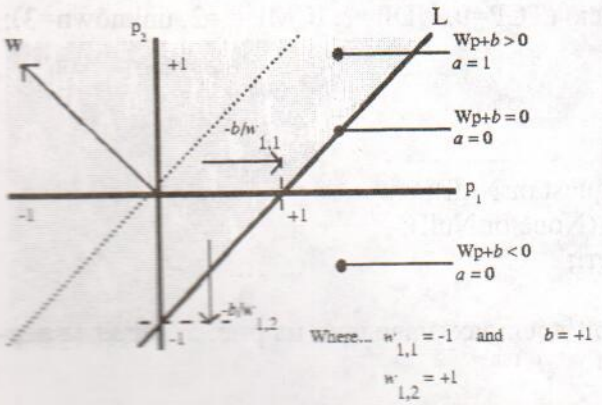

Рис. 3. Головне розв'язувальне правило персептрона

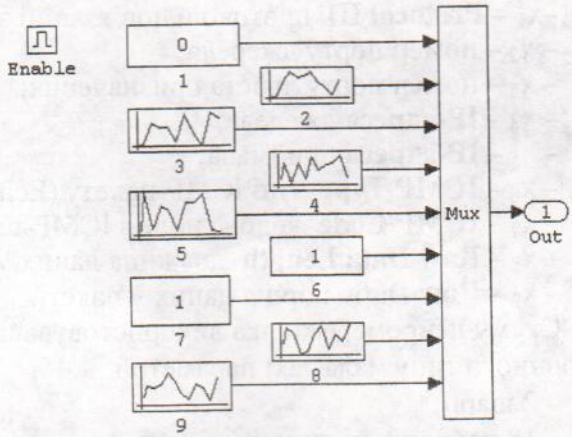

Рис. 4. Схема утворення та моделювання вхідних наборів

Навчання і тренування (тестування) мережі виконувалися на множинах [1] «атака безумовно **відсутня** (вихід y={0, 1})», «атака безумовно **присутня** (вихід y={1,0})».

Досліди показали, що навчена на наведених наборах (рис. 4) мережа, по-перше, має відносно невеликий час навчання, що дозволяє використовувати технологію в реальному часі, і, по-друге, всі тестові набори, які утворені з наборів першого та другого типів (коливання параметрів у межах ± 15-20%) і експертами діагностувалися як «відсутність атаки» та «присутність атаки» нейронною мережею, ідентифікувалися абсолютно безпомилково (див. таблицю). Це свідчить про високу ефективність нейромережної технології даного типу.

Множини тестових векторів створені на підставі виразу (2) і автоматично генеруються програмою.

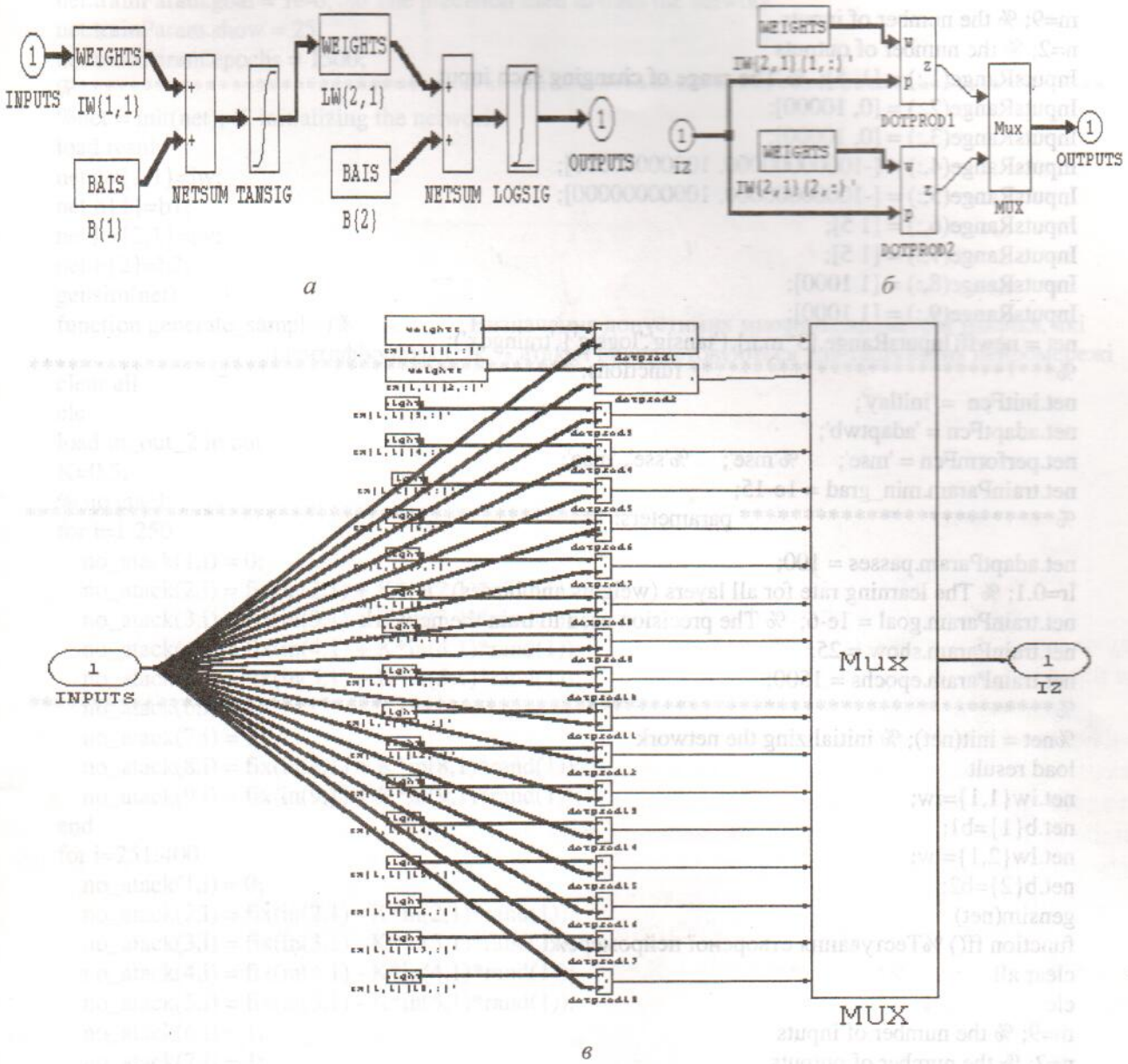

Рис. 5. Нейромережа в нотації MatLab:
а – загальна схема двошарової нейромережі; *б* – структура другого шару та вихід; *в* – структура першого шару

Базові вектори «атака присутня» – «атака відсутня»

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	y
0	2314	80	1573638018	-1580478590	1	1	401	3758	0, 1
0	1611	6101	8801886082	-926176166	1	1	0	2633	1, 0

Система навчена на базових векторах і протестована на створених тестових множинах.

Перевірка: подача на вхід сформованого тестового вектора, який не брав участі в тестуванні класу «атака присутня» дає на виході – {**1,0**}; відповідно «атака відсутня» – {**0,1**}.

Експерти можуть проаналізувати сформовані вектори з погляду їхньої приналежності до відповідних класів і порівняти з вихідними значеннями (0,1).

Запропонована програма дозволяє проаналізувати роботу нейромережної технології для ідентифікації атак.

```
function ff()
%Тестування створеної нейромережі
clear all
clc
m=9; % the number of inputs
n=2; % the number of outputs
InputsRange(1,:) = [1, 5];  % The range of changing each input
InputsRange(2,:) = [0, 10000];
InputsRange(3,:) = [0, 10000];
InputsRange(4,:) = [-10000000000, 10000000000];
InputsRange(5,:) = [-10000000000, 10000000000];
InputsRange(6,:) = [1 5];
InputsRange(7,:) = [1 5];
InputsRange(8,:) = [1 1000];
InputsRange(9,:) = [1 1000];
net = newff(InputsRange,[3*m,n],{'tansig','logsig'},'traingdx');
%***************************** functions: ****************************************
net.initFcn  = 'initlay';
net.adaptFcn = 'adaptwb';
net.performFcn = 'mse';     %'mse';     %'sse', 'mae';
net.trainParam.min_grad = 1e-15;
%************************* parameters: ****************************************

net.adaptParam.passes = 100;
lr=0.1; % The learning rate for all layers (weights and biases)
net.trainParam.goal = 1e-6;  % The precision used to train the network
net.trainParam.show = 25;
net.trainParam.epochs = 1500;
%*******************************************************************************
%net = init(net); % initializing the network
load result
net.iw{1,1}=iw;
net.b{1}=b1;
net.lw{2,1}=lw;
net.b{2}=b2;
gensim(net)
function ff() %Тестування створеної нейромережі
clear all
clc
m=9; % the number of inputs
n=2; % the number of outputs
InputsRange(1,:) = [1, 5];  % The range of changing each input
InputsRange(2,:) = [0, 10000];
InputsRange(3,:) = [0, 10000];
InputsRange(4,:) = [-10000000000, 10000000000];
InputsRange(5,:) = [-10000000000, 10000000000];
InputsRange(6,:) = [1 5];
InputsRange(7,:) = [1 5];
```

```
InputsRange(8,:) = [1 1000];
InputsRange(9,:) = [1 1000];
net = newff(InputsRange,[3*m,n],{'tansig','logsig'},'traingdx');
%***************************** functions: ****************************************
net.initFcn  = 'initlay';
net.adaptFcn = 'adaptwb';
net.performFcn = 'mse';     %'mse';     %'sse', 'mae';
net.trainParam.min_grad = 1e-15;
%************************* parameters: ***************************************

net.adaptParam.passes = 100;
lr=0.1; % The learning rate for all layers (weights and biases)
net.trainParam.goal = 1e-6;  % The precision used to train the network
net.trainParam.show = 25;
net.trainParam.epochs = 1500;
%*******************************************************************************
%net = init(net); % initializing the network
load result
net.iw{1,1}=iw;
net.b{1}=b1;
net.lw{2,1}=lw;
net.b{2}=b2;
gensim(net)
function generate_sampls()%          Визначення допустимих множин параметрів трафіка, які
                    ідентифікуються як % атака і використовуються для тестування нейромережі
clear all
clc
load in_out_2 in out
K=0.5;
% no atack
for i=1:250
   no_atack(1,i) = 0;
   no_atack(2,i) = fix(in(2,1) + K*in(2,1)*rand(1));
   no_atack(3,i) = fix(in(3,1) + K*in(3,1)*rand(1));
   no_atack(4,i) = fix(in(4,1) + K*in(4,1)*rand(1));
   no_atack(5,i) = fix(in(5,1) + K*in(5,1)*rand(1));
   no_atack(6,i) = 1;
   no_atack(7,i) = 1;
   no_atack(8,i) = fix(in(8,1) + K*in(8,1)*rand(1));
   no_atack(9,i) = fix(in(9,1) + K*in(9,1)*rand(1));
end
for i=251:400
   no_atack(1,i) = 0;
   no_atack(2,i) = fix(in(2,1) - K*in(2,1)*rand(1));
   no_atack(3,i) = fix(in(3,1) - K*in(3,1)*rand(1));
   no_atack(4,i) = fix(in(4,1) - K*in(4,1)*rand(1));
   no_atack(5,i) = fix(in(5,1) - K*in(5,1)*rand(1));
   no_atack(6,i) = 1;
   no_atack(7,i) = 1;
   no_atack(8,i) = fix(in(8,1) - K*in(8,1)*rand(1));
   no_atack(9,i) = fix(in(9,1) - K*in(9,1)*rand(1));
end
% atack
for i=1:50
   atack(1,i) = 0;
```

```
    atack(2,i) = fix(in(2,2) + 0.15*in(2,2)*rand(1));
    atack(3,i) = fix(in(3,2) + 0.15*in(3,2)*rand(1));
    atack(4,i) = fix(in(4,2) + 0.15*in(4,2)*rand(1));
    atack(5,i) = fix(in(5,2) + 0.15*in(5,2)*rand(1));
    atack(6,i) = 1;
    atack(7,i) = 1;
    atack(8,i) = fix(in(8,2) + 0.15*in(8,2)*rand(1));
    atack(9,i) = fix(in(9,2) + 0.15*in(9,2)*rand(1));
end
for i=51:100
    atack(1,i) = 0;
    atack(2,i) = fix(in(2,2) - 0.15*in(2,2)*rand(1));
    atack(3,i) = fix(in(3,2) - 0.15*in(3,2)*rand(1));
    atack(4,i) = fix(in(4,2) - 0.15*in(4,2)*rand(1));
    atack(5,i) = fix(in(5,2) - 0.15*in(5,2)*rand(1));
    atack(6,i) = 1;
    atack(7,i) = 1;
    atack(8,i) = fix(in(8,2) - 0.15*in(8,2)*rand(1));
    atack(9,i) = fix(in(9,2) - 0.15*in(9,2)*rand(1));
end
in=[in, atack, no_atack];
out(1,1)=0;
out(2,1)=1;
for i=3:102      % atack
  out(1,i)=1;
  out(2,i)=0;
end
for i=103:502          % no atack
   out(1,i)=0;
   out(2,i)=1;
end
save in_out in out
```

Автоматично генеруються випадкові вхідні набори, які відносяться до одного з класів, на екран одночасно виводиться згенерований набір і рішення, яке приймає нейромережа, відносячи його до одного з класів.

Другий тип нейромережних технологій, які доцільно застосовувати для ідентифікації атак, є нечіткі нейронні мережі, які застосовують правила висновків [4]. Цей тип нейромережних технологій планується застосовувати в тому випадку, коли стандартні класифікаційні моделі входять до зони невизначеності (рис. 6). З цією метою з множини входів експертами була виділена підмножина найбільш впливових входів. Кожна змінна уявляється як компонента нечіткої множини, яка, в свою чергу, упорядкована градаціями {**small, medium, large**}. Наприклад, твердження < довжина даних в пакеті **мала** > уявляється у вигляді, як показано на рис. 7.

Система правил для виявлення безумовної атаки може бути подана у вигляді:

$$\begin{cases} P_1 : \text{if } x_1 \text{ is small } \& \, x_2 \text{ is small } \& \ldots \& \, x_9 \text{ is large then } y = \{0,1\}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ P_n : \text{if } x_1 \text{ is large } \& \, x_2 \text{ is small } \& \ldots \& \, x_9 \text{ is medium then } y = \{0,1\}. \end{cases}$$

Аналогічна система правил може бути сформульована для безумовної відсутності атаки. Мережа, яка може реалізувати таку систему правил, може бути утвореною за загальними правилами, якщо додатково врахувати шар нечітких нейронів, які реалізують правила. Шар нечітких правил являє собою нечіткі правила типу P_i, нейрони мають лінійну функцію, і їхні виходи зв'язані з шаром дефадзифікації зваженими зв'язками. Навчання повинно відбити їхню фактичну важливість стосовно функцій приналежності виходу, що міститься в шарі дефадзифікації.

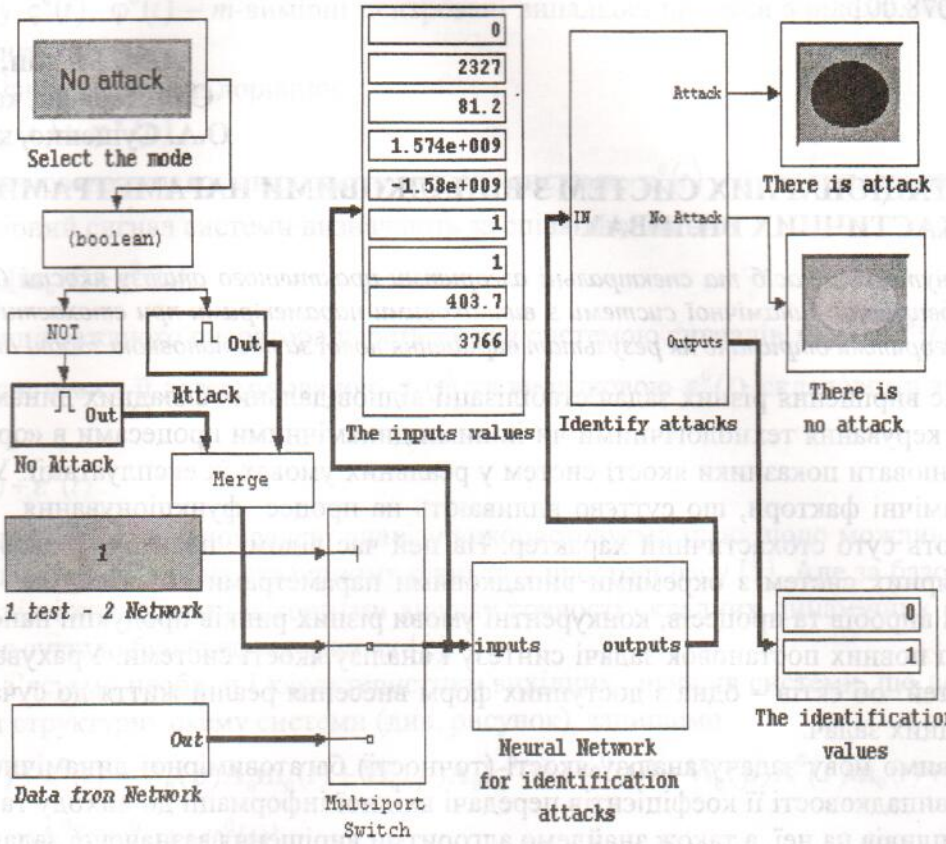

Рис. 6. Загальна схема моделювання ідентифікації атак в нотації MatLab/Simulink

На шарі дефадзифікації функції цього шару використовують для оцінки правила. Кожний нейрон на цьому шарі являє собою послідовне судження «...тоді у є...», і його функція приналежності може бути реалізована за рахунок поєднання однієї чи двох сигмоїдальних функцій. Навчання такої нечіткої мережі виконується загальними способами.

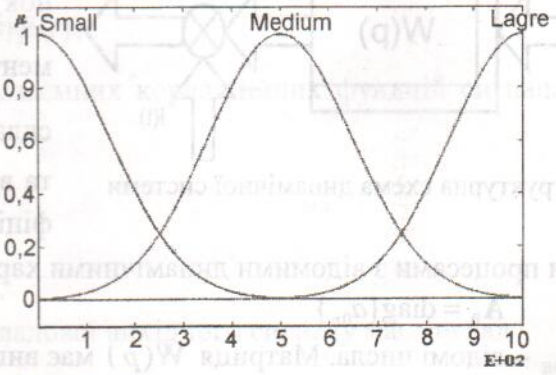

Рис. 7. Уявлення тверджень типу < довжина даних в пакеті **мала** >

Висновок. Нейромережні технології дозволяють ідентифікувати процедуру атаки на комп'ютерну мережу як на системному, так і на мережному рівнях. Раціональним є застосування нейромережних технологій в комплексних системах, які включають мережний та системний рівні.

Додаткове застосування fuzzy-нейромережних технологій дозволяє суттєво підвищити ефективність нейромережних технологій при ідентифікації атак.

Список літератури

1. *Кеннеди Дж.* Нейросетевые технологии в диагностике аномальной сетевой активности.– http://www.beda.stup.ae.ru/lib/neuro/content/id/ neuronet/htm.

2. *Hagan M.T., Demuth H.B.* Neural Network Design // PWS Publishing Company. – Boston, 1996.

3. *Caudill M., Butler C.* Understanding Neural Networks // Computer Explorations. – Vol. 1, 2. Cabridge, MA: the MIT Press, 1992.

4 . *Jang J.S., Sun C.T.* Neuro-fuzzy modeling and control // Proceedings of the IEEE. – 1995. – March.

Стаття надійшла до редакції 08.04.02.