2.7. РЕКУРСИВНІ НЕЙРОННІ МЕРЕЖІ ДЛЯ АВТОМАТИЗОВАНОГО АНАЛІЗУ ДАНИХ АТОМНО-СИЛОВОЇ СПЕКТРОСКОПІЇ

Оксана Литвин
(Київський університет імені Бориса Грінченка)
Петро Литвин
(Інститут фізики напівпровідників імені В.Є. Лашкарьова НАН України)

З огляду на розвиток сучасних цифрових технологій, зростання потужності комп'ютерних систем, розвиток хмарних сервісів та формування гігантських баз даних цілком закономірним є залучення інтелектуальних інформаційних систем до аналізу й використання великих масивів даних [1].

Зокрема, реалізація нових принципів та технологій обробки інформації, накопичення і збереження знань, їх доступність широкому загалу у вигляді хмарних сервісів відкривають нові перспективи в наукових дослідженнях та наукоємних виробництвах у контексті інтерпретації даних спостережень, класифікації властивостей матеріалів і систем, теоретичних підходів та моделювання в технологіях. Особливо перспективним для технологічних розробок у біоінженерії, медицині, астрономії та інших сферах є використання штучних нейронних мереж [2–5].

Штучні нейронні мережі — сучасна інтелектуальна технологія, яка знаходить своє застосування в різних галузях науки: моделюванні, розпізнаванні образів, обробці сигналів, апроксимації функцій тощо. Її використання для розв'язування складних (масштабних) завдань зумовлюється важливими властивостями, а саме здатністю до навчання та узагальнення, тобто здатністю отримувати обґрунтований результат на основі даних, які не траплялися в процесі навчання, а також розпаралелювання процесу обчислень. У статті представлено застосування нейромереж для аналізу даних атомно-силової спектроскопії на прикладі дослідження наномеханічних властивостей біоматеріалів.

Атомно-силовий мікроскоп (АСМ) широко використовується для вимірювання властивостей механічних зразків на нанорівні. Для цього реєструються силові криві взаємодії чутливого зонду на гнучкій консолі із поверхнею досліджуваного зразка. Дальший аналіз особливостей кривих атомно-силової спектроскопії дає можливість отримати широкий спектр даних про наномеханічні, адгезійні, сорбційні, енергетичні та ін. характеристики досліджуваних об'єктів на молекулярному та атомарному рівнях, що недоступно іншим методам діагностики. Наприклад, серії значень локальних модулів пружності тонких плівок, ригідності клітинних стінок, енергії адгезії до поверхні, молекулярних сил денатурації ДНК, меж та параметрів руйнування матеріалів. Сканування зразка забезпечує отримання даних на деякій площі його поверхні (локальні значення) та карти механічних характеристик. Застосування методів наноіндентування, реалізованих на базі атомно-силової мікроскопії і, зокрема, методу силової спектроскопії, відкриває унікальні можливості для дослідження локальних механічних властивостей відносно м'яких матеріалів, у тому числі 2D матеріалів і біологічних об'єктів.

Однак взаємозв'язки між механічною реакцією, морфологією та функціональною поведінкою діагностичної системи є досить складними і їх важко інтерпретувати однозначно. Тому існує значний розрив між застосуванням методів атомно-силової мікроскопії і спектроскопії в академічних, фундаментальних дослідженнях та їх впровадженням, наприклад у контролі технологічних процесів на виробництві й клінічній біомедичній діагностиці. Якщо ж ще взяти до уваги розмаїття фізичних процесів у контакті зонд — поверхня, які мають місце в специфічних матеріалах та біологічних об'єктах, а також обсяги інформації, яку потрібно аналізувати, то навіть для підготовленого фахівця обробка експериментальних даних атомно-силової спектроскопії стає надзвичайно складним завданням. Саме використання алгоритмів штучних нейронних мереж дасть змогу суттєво спростити та автоматизувати аналіз великих масивів даних.

Теорія штучних нейронних мереж

Модель штучного нейрона. Нейрон — обчислювальна одиниця, яка отримує інформацію (input data), виконує над нею прості обчислення і передає її далі (output data). Штучний нейрон імітує в першому наближенні властивості біологічного нейрона.

На *рис. 2.7.1* наведена модель формального нейрона за [6]. Саме ця праця вважається першою роботою, яка заклала теоретичний фундамент для створення штучних моделей нейронів і нейронних мереж.

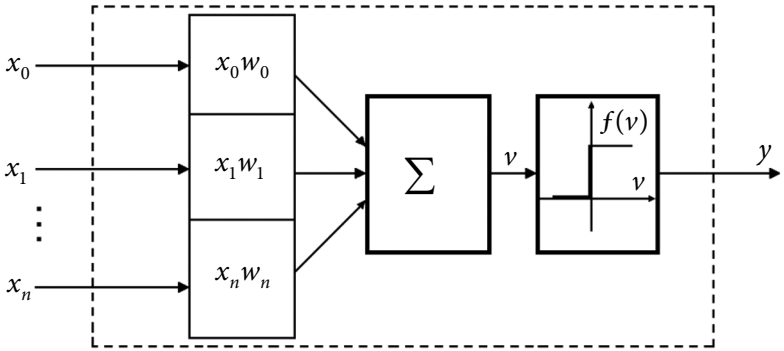

Рис. 2.7.1. Функціональна схема формального нейрона

З сучасної точки зору формальний нейрон — це математична модель простого процесора, що має кілька входів й один вихід. На вхід нейрона через дендрити надходить деяка множина сигналів, кожен з яких є виходом іншого нейрона. Вектор вхідних сигналів X (x_0, x_1,…, x_n) перетворюється нейроном у вихідний сигнал y (що поширюється аксоном, передається через синапс — контакт з дендритом іншого нейрона) з використанням трьох функціональних блоків: локальної пам'яті, блоку підсумовування і блоку нелінійного перетворення.

Вектор локальної пам'яті містить інформацію про вагові коефіцієнти W (w_0, w_1,…, w_n), з якими вхідні сигнали будуть інтерпретуватися нейроном. Ці змінні ваги є аналогом чутливості пластичних синаптичних контактів. Вибором ваг досягається та чи та інтегральна функція нейрона: чим більший ваговий коефіцієнт, тим більший внесок цей сигнал здійснюватиме. Ваги можуть змінюватися в процесі навчання мережі (див. далі) відповідно до топології мережі та навчальних правил.

У блоці підсумовування відбувається накопичення загального вхідного сигналу, що дорівнює зваженій сумі входів v.

У різних програмних реалізаціях функція суматора може бути складнішою, наприклад вибір мінімуму, максимуму, середнього арифметичного, добутку тощо. Тобто перед надходженням до блоку перетворення (т. зв. функції активації / збудження нейрона) вхідні сигнали та вагові коефіцієнти можуть комбінуватися багатьма способами. Алгоритми для комбінування входів нейронів визначають відповідно до мережної архітектури. Сюди також може бути включений пороговий елемент b (bias), який відображає збільшення або зменшення вхідного сигналу, що передається на функцію активації f. Тоді функціонування нейрона можна описати парою рівнянь:

$$v = \sum_{i=0}^{n} w_i x_i$$

$$y = f(v + b).$$

Функція активації, або передавальна (activation function, excitation function, squashing function, transfer function), штучного нейрона може бути різною. Здебільшого це сигмоїда, яка відображає дійсні числа на інтервал (-1,1) (гіперболічний тангенс) або (0,1) (логістична функція).

Архітектура нейронних мереж. Штучна нейронна мережа (ШНМ, або artificial neural network (ANN)) — математична модель та її апаратно-програмна реалізація біологічних нейронних мереж живого організму.

Можна виділити шість основних завдань, які можуть розв'язувати нейронні мережі [7]:

1) асоціативна пам'ять — у мережі запам'ятовуються образи, які вона потім відтворює за неповними або зашумленими описами;

2) розпізнавання образів — процес, у якому отримуваний образ / сигнал має бути віднесений до якогось означеного класу;

3) керування процесом або системою;

4) фільтрація — процес вилучення корисної інформації із набору зашумлених даних;

5) формування діаграми направленості — просторовий випадок фільтрації;

6) апроксимація функцій.

Спосіб зв'язку нейронів у нейромережі та типи нейронів визначають її архітектуру. Серед відомих архітектурних рішень виділяють слабозв'язані мережі, коли кожен нейрон мережі зв'язаний лише із сусідніми, та повнозв'язані, коли входи кожного нейрона зв'язані з виходами решти нейронів. Найпоширенішою архітектурою є мережі з пошаровою структурою: нейрони певним чином об'єднуються в шари. У кожній нейромережі існує один вхідний шар, хоча б один прихований та один вихідний шар. Шар вхідних нейронів отримує дані ззовні. Вихідний шар пересилає інформацію до зовнішнього середовища, до вторинного комп'ютерного процесу або інших пристроїв. Між цими двома шарами може бути багато прихованих шарів, які містять безліч нейронів у різноманітних зв'язаних структурах. Входи та виходи кожного з прихованих нейронів сполучені з іншими нейронами.

Першою моделлю нейромереж вважається персептрон Розенблатта — мережа з одним прихованим шаром [8–9]. Теорія персептронів є основою для багатьох типів штучних нейромереж.

Елементарний персептрон (*рис. 2.7.2*) складається з елементів трьох типів: S-елемент — сенсор або рецептор (input), A-елемент — асоціативний (*f(x)*), R-елемент — реагуючий (output). Нейрони прихованого шару A обчислюють зважену суму елементів вхідного сигналу, пропускають результат через порогову функцію, вихід якої дорівнює +1 чи 0. R-елемент видає сигнал +1, якщо зважена сума його вхідних сигналів від попереднього шару є суворо додатною, і сигнал –1, якщо сума його вхідних сигналів є суворо від'ємною. Якщо сума вхідних сигналів дорівнює нулю, вихід вважається або рівним нулю, або невизначеним. Тоді у разі розв'язання задачі

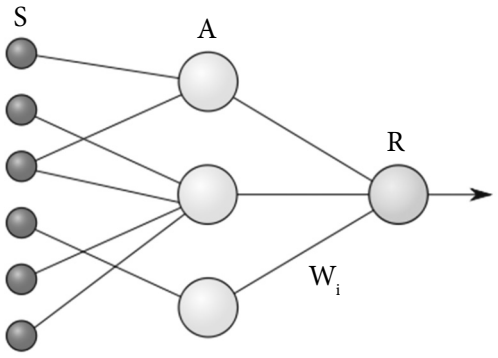

Рис. 2.7.2. Одношаровий персептрон Розенблатта. Ваги зв'язків S — A можуть мати значення –1, 1 або 0 (тобто відсутність зв'язку). Ваги зв'язків A — R W можуть мати довільне значення

класифікації залежно від значення вихідного сигналу приймається рішення: +1 — вхідний сигнал належить класу A, –1 — вхідний сигнал належить класу B.

Класифікація нейронних структур та їх модифікацій, що орієнтовані на розв'язання конкретного типу задач, — завдання непросте, оскільки їх є велика кількість. На нашу думку, досить повно та логічно здійснено класифікацію в [10] (*рис. 2.7.3*).

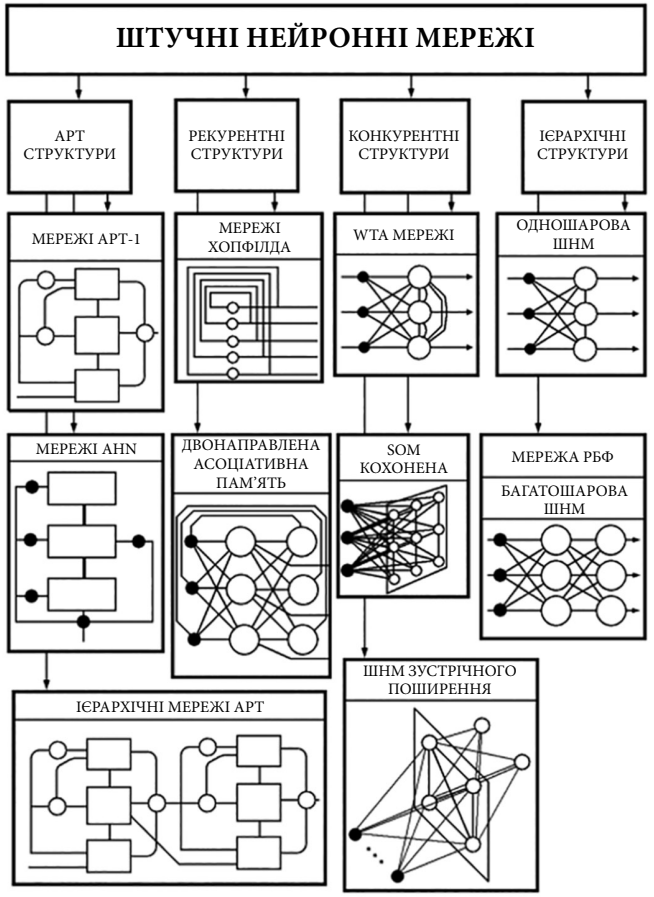

Рис. 2.7.3. Типи нейронних структур [10]

Оскільки в нашому дослідженні були використані одно- та багатошарові мережі прямого поширення, розглянемо їх детальніше.

Як було зазначено вище, персептрон Розенблатта за сучасною термінологією зараховують до одношарових ШНМ прямого поширення (feed-forward), яка складається із m нейронів, здатних одночасно прийняти вхідний вектор сигналів $X = (x_1, ..., x_з, ..., x_n)$. Згідно з моделлю формального нейрона, кожен з його вхідних сигналів множать на ваговий коефіцієнт w_{ij}, де i — поточний номер елемента вектора X, а j — поточний номер нейрона (формула (1)).

Вагові коефіцієнти одношарової нейронної мережі утворюють матрицю вагових коефіцієнтів:

$$W = \begin{pmatrix} w_{11} & \cdots & w_{1j} & \cdots & w_{1m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{i1} & \cdots & w_{ij} & \cdots & w_{im} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ w_{n1} & \cdots & w_{nj} & \cdots & w_{nm} \end{pmatrix}.$$

Тоді вектор аргументів визначатиметься як добуток $V = XW$, а вектор вихідних сигналів є вектором значень активаційних функцій:

$$Y = F(V) = \begin{cases} f_1(v_1), \\ \cdots \\ f_j(v_j), \\ \cdots \\ f_m(v_m). \end{cases} \qquad (2)$$

Спроби застосувати одношарові нейронні мережі для розв'язування деякого кола задач наштовхнулися на низку труднощів, пов'язаних з проблемою лінійної роздільності [11]. Природним вирішенням цієї проблеми стало застосування багатошарових ШНМ. Ієрархічна структура багатошарової мережі складається із m нейронів першого (прихованого) шару, які приймають вхідний вектор сигналів, n нейронів другого шару і т. д., аж до k нейронів останнього — вихідного шару. У таких нейронних мережах напрям поширення сигналів «прямий»: сигнали рухаються, починаючи з входу, через один або декілька прихованих шарів до вихідного шару. Тобто синаптичні зв'язки організовані таким чином, що кожен нейрон цього рівня ієрархії отримує інформацію тільки від деякої множини нейронів, розміщених на більш низькому рівні.

Очевидно, що багатошарова нейронна мережа може бути отримана шляхом каскадного об'єднання одношарових мереж з матрицями вагових коефіцієнтів $W^1, W^2, ..., W^p$, де p — кількість шарів нейронної мережі. При цьому при лінійній активаційній функції багатошарова мережа буде еквівалентна одношаровій з матрицею вагових коефіцієнтів $W = W^1 \cdot W^2 \cdot ... \cdot W^p$, тому побудова таких мереж виправдана тільки у разі використання у нейронах нелінійних активаційних функцій.

Навчання нейронної мережі прямого поширення. Загальне визначення поняття «машинне навчання» дав Томас Мітчелл у класичній книзі «Машинне навчання» [12]: «Комп'ютерна програма навчається в міру накопичення досвіду щодо деякого класу задач T і цільової функції P, якщо якість розв'язання цих задач (щодо P) поліпшується з отриманням нового досвіду». Тут важливі два моменти: 1) центральне місце при машинному навчанні займають не дані (хоча вони теж є), а цільова функція; 2) задачі розв'язуються з даних без того, аби бути програмованими явно. Тобто вже перед розв'язанням будь-якої практичної задачі необхідно визначити цільову функцію і домовитися про те, як оцінюватимуться результати. Вибір цільової функції цілком визначає всю дальшу роботу, і навіть у схожих завданнях різні цільові функції можуть привести до абсолютно різних моделей.

Навчання (тренування) нейронної мережі — це процес, при якому вільні параметри нейронної мережі адаптуються в результаті її безперервної стимуляції зовнішнім оточенням. Тип навчання визначається тим способом, яким виробляються зміни параметрів. Розрізняють два основні види навчання нейронної мережі — з учителем (supervised learning) та без учителя (self-organized learning), алгоритми якого базуються на принципі самоорганізації, де в ролі «вчителя» виступає зовнішнє середовище. Перше передбачає підлаштування вагових коефіцієнтів сигналів нейронів кожного шару так, що похибка у вихідних векторах цільової функції стає міні-

мальною. Алгоритми навчання зазвичай функціонують покроково. Ці кроки прийнято називати епохами або циклами. На кожному циклі на вхід мережі послідовно подаються всі елементи тренувального набору даних (навчальна вибірка), потім обчислюються вихідні значення мережі, порівнюються з цільовими й обчислюється функціонал помилки. Значення функціоналу та його градієнта використовуються для коригування ваг і зміщень, після чого всі дії повторюються. Початкові значення ваг і зміщень вибираються випадковим чином, а процес навчання припиняється, коли виконано певну кількість циклів або коли помилка досягне деякого мінімального значення чи перестане зменшуватися.

При такому підході постає питання щодо адекватності розміру навчаючої вибірки. Виявляється, що для хорошого узагальнення досить, аби її розмір N задовольняв співвідношення $N = O(W/\varepsilon)$, де W — загальна кількість вільних параметрів (синаптичних ваг і порогових значень); ε — допустима точність похибки; $O(*)$ — порядок величини в дужках. Зазначений вираз отримано з емпіричного правила Відроу для алгоритму найменших квадратів [13].

Задача апроксимації функції. Оскільки під час дослідження ми використовували нейронні мережі саме для задач апроксимації функцій, далі розглядатимемо саме цю частину теорії нейронних мереж.

Нехай маємо нелінійне відображення типу «вхід» — «вихід» F: $X \to Y$, де вектор X — вхід, вектор $Y = F(X)$ — вихід, а функція F — невідома. Щоб визначити цю функцію дано множину прикладів (навчальна вибірка): $\Psi = \left\{(\tilde{O}^{(3)}, y^{(3)})\right\}_{i=1}^{N}$. Завдання нейронної мережі знайти найкраще сімейство функцій Φ та вибрати найкращу функцію з параметром W, що входить до цього сімейства $\phi_W(X^{(i)}) \subset \Phi$.

Ця задача завжди є розв'язною, оскільки математично доведено, що ШНМ прямого поширення, у якій зв'язки не утворюють циклів, з одним прихованим шаром може апроксимувати будь-яку неперервну функцію багатьох змінних з будь-якою точністю (теорема Цибенко — універсальна теорема апроксимації) [14]. Умовами є достатня кількість нейронів прихованого шару, вдалий добір вагових коефіцієнтів між вхідними нейронами і нейронами прихованого шару та між зв'язками від нейронів прихованого шару й вихідним

нейроном, а також порогового елемента. З іншого боку, одну і ту ж функцію часто можна набагато краще наблизити глибшою мережею (з більшою кількістю прихованих шарів), навіть якщо загальну кількість нейронів у мережі залишити незмінною [15].

Критерієм вибору апроксимаційної функції є цільова функція $E\left[\phi_W(X^{(i)}), y^{(i)}\right]$, яка визначає «відстань» між результатом дії апроксимуючої функції $\phi_W(X^{(i)})$ та заданим навчальною вибіркою вихідним елементом даних $y^{(i)}$. Таким чином нейронна мережа має навчитися розв'язувати завдання оптимізації: за заданою функцією знайти аргументи, в яких ця функція максимізується або мінімізується.

Найпростішим сімейством апроксимуючих функцій є лінійне сімейство, що задають виразом (член $x_0 = 1$ введено для спрощення виразу) [10]:

$$\phi_W(X^{(i)}) = w_0 + \sum_{j=1}^{m} w_j x_j^{(i)} = \sum_{j=0}^{m} w_j x_j^{(i)} = W^T X^{(i)}. \quad (3)$$

Залежно від типу задач застосовують різні способи визначення цільової функції. Широко поширена середньоквадратична цільова функція, яку задають таким чином:

$$E = \sum_{i=1}^{N} \left[\phi_W(X^{(i)}) - y^{(i)}\right]^2. \quad (4)$$

Виходячи з виду функції (3), для пошуку мінімуму середньоквадратичної цільової функції E (4) можна застосовувати аналітичний метод лінійної регресії. Проте його практичне використання зумовлює значні обчислювальні труднощі. Подолати цю проблему можна за допомогою ітераційних підходів.

Розглянемо двошарову повнозв'язну нейронну мережу (*рис. 2.7.4*).

Нульовий шар цієї мережі — шар вхідних сигналів — не містить нейронів. Останній шар — вихідний (у цьому разі — із лінійною активаційною функцією нейронів). Всі шари, розміщені між нульовим та вихідним, є прихованими шарами з нелінійною активаційною функцією нейронів. Кожен нейрон мережі продукує зважену суму своїх входів, пропускає цю величину через активаційну функцію і видає вихідне значення. І як було зазначено, така мережа може

моделювати функцію практично будь-якої складності, причому число шарів і число нейронів у кожному шарі може бути змінено залежно від складності апроксимованої функції.

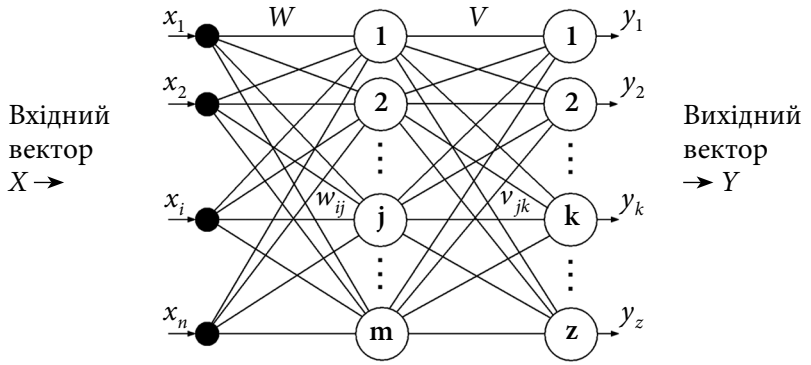

Рис. 2.7.4. Двошарова нейронна мережа прямого поширення

Робота такої мережі описується рівняннями:

$$\tilde{y} = \phi\big(s(x)\big), \qquad s(x) = v_0 + \sum_{j=1}^{m} v_j h_j,$$
$$h_j = f\left(t_j\right) = f\left(w_{0j} + \sum_{i=1}^{n} w_{ij} x_j\right). \tag{5}$$

Тут j, i — індекси елемента вхідного вектора та нейрона прихованого шару відповідно; x_i — елемент вхідного вектора; t_j — аргумент активаційної функції нейрона прихованого шару; h_j — елемент вихідного вектора прихованого шару; s — аргумент активаційної функції нейрона вихідного шару; $\tilde{y}$ — вихідний вектор мережі; $f(t)$ — активаційна функція нейронів прихованого шару; $\phi(s)$ — активаційна функція нейронів вихідного шару.

Існує велика кількість алгоритмів мінімізації цільової функції. Нижче розглянемо ті, які ми використали в дослідженні.

Метод зворотного поширення похибки. Найбільш відомим, ефективним та легким у застосуванні при навчанні штучних нейронних мереж є алгоритм зворотного поширення похибки (Back Propagation) [16–17].

Нехай дано навчальну вибірку $\Psi = \left\{(X^{(n)}, y^{(n)})\right\}_{n=1}^{N}$, що містить N пар вхідних і вихідних векторів $X^{(n)}, y^{(n)}$ та множину $W = \{w, v\}$ параметрів нейронів прихованого шару w і параметрів вихідного шару v. Метод зворотного поширення похибки полягає в мінімізації цільової функції

$$E = \frac{1}{2N} \sum_{n=1}^{N} \left[y^{(n)} - \tilde{y}^{(n)}(x) \right]^2 \tag{6}$$

шляхом двох проходів обчислень. При прямому проході синаптичні ваги залишаються незмінними в усій мережі, а функціональні сигнали обчислюються послідовно від шару до шару. На виході отримуємо вектор $\tilde{y}$ (5) і обчислюємо цільову функцію (6). Зворотний прохід передбачає поширення сигналів помилки від виходів мережі до її входів, модифікуючи ваги кожного шару на відповідний крок:

$$\Delta v_j = -\eta \frac{\partial E}{\partial v_j} = \frac{\eta}{N} \sum_{n=1}^{N} \left(y^{(n)} - \tilde{y}^{(n)} \right) \phi'\left(s^{(n)}\right) h_j,$$
$$\Delta w_{ij} = -\eta \frac{\partial E}{\partial w_{ij}} = \frac{\eta}{N} \sum_{n=1}^{N} \left(y^{(n)} - \tilde{y}^{(n)} \right) \phi'\left(s^{(n)}\right) v_j f'\left(t_j^{(n)}\right) x_i^{(n)},$$

де $0 < \eta < 1$ — множник, який задає швидкість навчання. Для обчислення чергового кроку параметра ΔW за цим алгоритмом необхідно виконати обробку всієї навчальної вибірки Ψ. Ознакою закінчення процесу навчання може бути досягнення умови $\partial E / \partial W \le \varepsilon$ або мінімальна зміна цільової функції на такі ітерації $\left|\frac{E(t+1) - E(t)}{E(t)}\right| < \varepsilon$, де $\varepsilon > 0$ — значення порога точності.

Алгоритм реального часу дозволяє модифікацію кроку параметра після обробки кожної навчальної пари $(X^{(n)}, y^{(n)})$, що забезпечує можливість кращої адаптації алгоритму до зміни вхідної вибірки. Величина кроку параметра ΔW модифікується в реальному часі за формулою:

$$\Delta W^{(n)} = -\eta \frac{\partial E^{(n)}}{\partial W} + \alpha \Delta W^{(n-1)},$$

де $0 < \alpha < 1$ — коефіцієнт впливу попередньої ітерації; $\lambda(n) = \eta^{(n+1)}$ — функція динамічного формування коефіцієнта навчання η.

При ретельному доборі функції $\lambda(n)$ алгоритм реального часу дає змогу підвищити швидкість збіжності методу зворотного поширення.

Приклад блок-схеми алгоритму реального часу для методу зворотного поширення наведено на *рис. 2.7.5*.

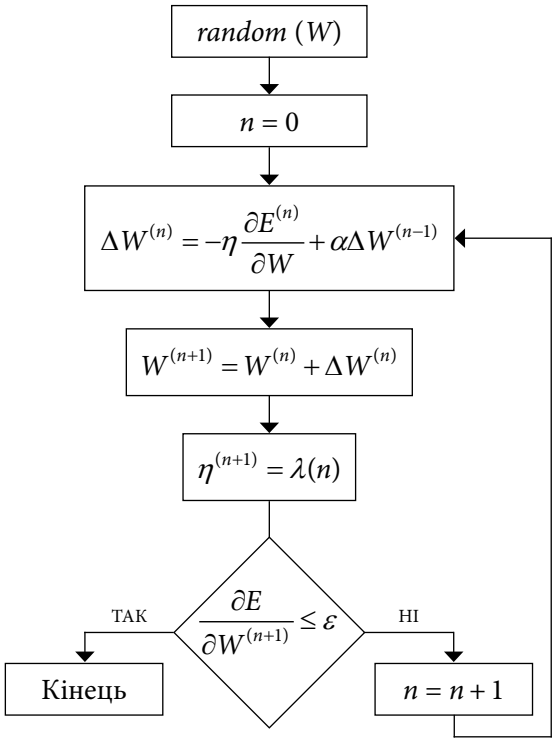

Рис. 2.7.5. Алгоритм реального часу для методу зворотного поширення [10]

Слід зазначити, що у разі навчання мережі методом зворотного поширення помилки функції активації мають бути монотонно зростаючими і диференційовними.

При реалізації алгоритму можна змінювати не всі ваги мережі, а деякі з них залишати фіксованими. Цього можна досягти, встановивши для відповідних ваг параметр інтенсивності навчання η рівний нулю.

Незважаючи на успіх застосування методу зворотного поширення в різних задачах, трапляються випадки, коли процес навчання триває нескінченно довго. Причинами цього є [18]:

1) параліч мережі — у процесі навчання мережі значення ваг можуть стати дуже великими величинами. Це може призвести до того, що всі або більшість нейронів будуть функціонувати при дуже великих значеннях OUT у ділянці, де похідна стискаючої функції дуже мала. Оскільки помилка, що посилається назад під час навчання, пропорційна цій похідній, то процес навчання може практично завмерти. Для запобігання цьому можна зменшити розмір кроку η, але це значно сповільнює час навчання;

2) локальні мінімуми — зворотне поширення використовує різновид градієнтного спуску, тобто здійснює спуск вниз поверхнею помилки, безперервно підлаштовуючи ваги в напрямку до мінімуму. Поверхня помилки складної мережі дуже неоднорідна в просторі високої розмірності. Мережа може потрапити в локальний мінімум (неглибоку долину), не досягнувши глобального мінімуму. У точці локального мінімуму всі напрямки ведуть вгору, і мережа нездатна з нього вибратися.

Алгоритм Левенберга — Марквардта. Алгоритм Левенберга — Марквардта (Levenberg — Marquardt algorithm (LMA)) — метод оптимізації параметрів нелінійних регресійних моделей у задачах апроксимації кривої [19–20]. Як критерій оптимізації використовується середньоквадратична похибка моделі в навчальній вибірці. Алгоритм полягає в послідовному наближенні заданих початкових значень параметрів до шуканого локального оптимуму.

Нехай задана регресійна вибірка множина пар $D = \{(x_n, y_n)\}_{n=1}^{N}$ вільної змінної x та залежної змінної y. Задана регресійна модель — функція $f(w, x_n)$, диференційована в ділянці $W \times X$. Потрібно знайти таке значення вектора параметрів w, яке б доставляло локальний мінімум функції похибки:

$$E_D = \sum_{n=1}^{N} \left(y_n - f(w, x_n) \right)^2. \quad (7)$$

Перед початком алгоритму задається початковий вектор параметрів w, який на кожній ітерації заміняється на вектор $w + \Delta w$.

Для оцінки приросту Δw використовується лінійне наближення функції:

$$f(w+\Delta w,x)-f(w,x)\approx J\Delta w,$$

де J — якобіан функції $f(w,x_n)$ у точці w. Матриця J має вигляд:

$$J=\begin{bmatrix} \frac{\partial f(w,x_1)}{\partial w_1} & \cdots & \frac{\partial f(w,x_1)}{\partial w_R} \\ \vdots & \ddots & \vdots \\ \frac{\partial f(w,x_N)}{\partial w_1} & \cdots & \frac{\partial f(w,x_N)}{\partial w_R} \end{bmatrix}.$$

Тут вектор параметрів $w=\left[w_1,...,w_R\right]^T$.

Приріст Δw у точці w, при якому функція E_D досягатиме мінімуму, рівний нулю. Тому для пошуку наступного значення приросту дальшим кроком Δw прирівняємо до нуля вектор частинних похідних функції E_D за w. Для цього формулу (7) подаємо у вигляді $E_D=\left|y-f(w+\Delta w)\right|^2$, де $y=\left[y_1,...,y_N\right]^T$ і $f(w+\Delta w)=\left[f(w+\Delta w,x_1),...,f(w+\Delta w,x_N)\right]^T$.

Тоді дістанемо:

$$E_D=\left|y-f(w+\Delta w)\right|^2=\left(y-f(w+\Delta w)\right)^T\left(y-f(w+\Delta w)\right)=$$
$$=f^T(w+\Delta w)f(w+\Delta w)-2y^Tf(w+\Delta w)+y^Ty$$

і, диференціюючи за w, отримаємо:

$$\frac{\partial E_D}{\partial w}=(J^TJ)\Delta w-J^T\left(y-f(w)\right)=0.$$

Тоді, щоб знайти вектор Δw, потрібно розв'язати систему лінійних рівнянь:

$$\Delta w=J^T\left(y-f(w)\right)/(J^TJ).$$

Оскільки матриця J^TJ може бути суттєво виродженою, Марквард [20] запропонував ввести параметр регуляризації $\lambda\geq 0$ (призначається на кожній ітерації алгоритму): $\Delta w=J^T\left(y-f(w)\right)/(J^TJ+\lambda I)$, де I — одинична матриця.

Алгоритм зупиняється в тому разі, якщо приріст Δw у наступній ітерації менше заданого значення або якщо параметри w доставляють похибку функції E_D, меншу заданої величини. Значення вектора w на останній ітерації вважається шуканим.

Недоліком алгоритму є значне збільшення параметра λ при поганій швидкості апроксимації. При цьому перетворення матриці $J^TJ+\lambda I$ втрачає зміст. Цей недолік можна усунути, замінивши одиничну матрицю I діагоналлю матриціJ J^TJ:

$$\Delta w=J^T\left(y-f(w)\right)/(J^TJ+\lambda\,\mathrm{diag}(J^TJ)).$$

Алгоритм спряжених градієнтів. Алгоритм зворотного поширення помилки коригує настроювані параметри мережі в напрямку найшвидшого зменшення функціоналу помилки. Проте останній далеко не завжди є найсприятливішим напрямком, таким, щоб за мале число кроків забезпечити збігання до мінімуму функціоналу. Існують напрямки, рухаючись за якими, можна визначити шуканий мінімум набагато швидше. Зокрема, це можуть бути так звані спряжені напрямки, а відповідний метод оптимізації — метод спряжених градієнтів (conjugate gradient) [21]. Це метод знаходження локального екстремуму функції на основі інформації про її актуальне значення та інформації, отриманої на попередніх кроках (градієнт).

Розглянемо задачу оптимізації квадратичної функції:

$$f(X)\frac{1}{2}=X^TAX-B^TX+c, \qquad (8)$$

де X, B — вектори розмірністю $W\times 1$; A — симетрична матриця розмірністю $W\times W$; c — скаляр. Мінімум цієї функції досягається при єдиному значенні $X^*=A^{-1}B$. Тобто мінімізація функції e і розв'язок системи лінійних рівнянь $AX^*=B$ є еквівалентними задачами.

Для цієї матриці A множина ненульових векторів $\mathrm{S}=\left\{\mathrm{S}^{(0)},\mathrm{S}^{(1)},\ldots\mathrm{S}^{(\mathrm{n})},\right\}$ називається A-спряженою (A-conjugate), якщо для всіх $i\neq j$ виконується відношення $S^{(i)}A\,S^{(j)}=0$. Якщо матриця A — одинична, то спряженість є еквівалентом ортогональності.

Для пошуку локального мінімуму як матриці A застосовується матриця Гессе:

$$H(f)=\begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1\,\partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1\,\partial x_n} \\ \frac{\partial^2 f}{\partial x_2\,\partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2\,\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n\,\partial x_1} & \frac{\partial^2 f}{\partial x_n\,\partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}.$$

Тоді спряженими векторами будуть вектори, що задовольняють умову: $S^{(i)T}HS^{(j)}=0$ $(i\neq j)$, $S^{(i)T}HS^{(i)}\geq 0$ для $i=1, 2, \ldots, n$.

Алгоритм спряжених градієнтів на першій ітерації починає пошук у напрямі антиградієнта квадратичної функції:

$$s_k^{(i)}=-f'(x_k^{(i)})+\beta_{k-1}s_{k-1}^{(i)},\quad k\geq 1$$
$$s_0^{(i)}=-f'(x_0^{(i)}).$$

Величини β_{k-1} — вибираються так, щоб напрямки $s_k^{(i)}$ і $s_{k-1}^{(i)}$ були H-спряженими.

У результаті для квадратичної функції (8) ітераційний процес мінімізації має вигляд: $x_{k+1}^{(i)}=x_k^{(i)}+a_k s_k^{(i)}$, де a_k — величина кроку; $s_k^{(i)}$ — напрямок «спуску».

Однією з найбільш істотних проблем у методах сполучених градієнтів є проблема ефективного побудови напрямків, тобто найоптимальнішого обчислення константи β_{k-1}. Існує багато алгоритмів, які по-різному вирішують цю проблему. У нашому разі був застосований масштабований алгоритм спряжених градієнтів (scaled conjugate gradient algorithm) [22].

Оптимізація параметрів та навчання нейронної мережі для аналізу даних атомно-силової спектроскопії

Як приклад застосування ШНМ для аналізу даних атомно-силової спектроскопії наведемо наші дослідження серій силових кривих при визначенні локального модуля пружності клітин крові людини.

Типова схема вимірювань в атомно-силовій спектроскопії взаємодій зонд — поверхня подано на *рис. 2.7.6.*

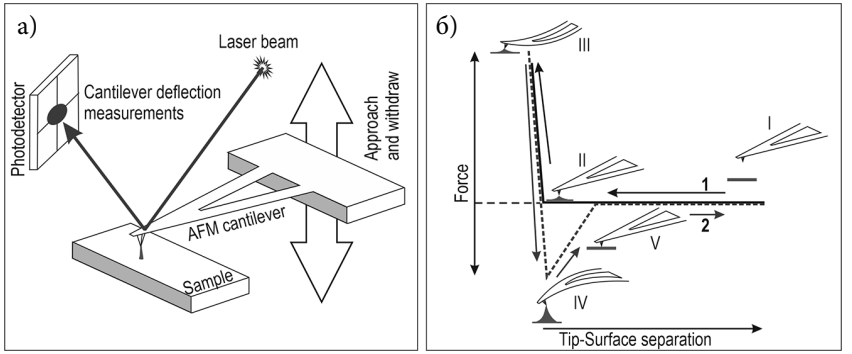

Рис. 2.7.6. Схема вимірювань в атомно-силовій спектроскопії (а) і схематичний вигляд силової кривої при вимірюваннях на повітрі (б): крива підведення (1), крива відведення нановістря від поверхні (2) [23]

У положенні I зонд розміщений далеко від поверхні й взаємодія між ними відсутня. У положенні II зонд наближається до поверхні так близько (рух по кривій позначається стрілкою), що переходить у контакт з поверхнею за рахунок дії Ван-дер-Ваальсових сил притягання. У положенні III зонд досягає заданого максимуму сили притискання до поверхні, після чого починається стадія відведення IV. Загалом ділянки III та IV силової кривої не збігаються через наявність гістерезису привнесеного за рахунок дії адгезійних сил. У положенні IV сила пружної деформації вигнутого кантилевера (консолі, на якій розміщене нановістря) перевищує силу адгезії і зонд стрибкоподібно відривається від поверхні. У положенні V кантилевер зонда АСМ повертається в стан рівноваги. Таким чином результатом вимірювань є серії силових кривих, аналіз яких і буде завданням нейромережі.

Для конструювання, навчання та тестування нейронної мережі ми вибрали середовище Matlab [24]. Це потужний пакет, який включає велике число макросів (так званих *m*-функцій) і дає можливість значно пришвидшити розробку прикладних програм різного рівня складності. Для роботи з нейронними мережами в Matlab передбачено кілька рівнів. Найпростіші мережі з використанням алгоритмів навчання Левенберга — Марквардта, Байєсової регуляризації та спряжених градієнтів можна дуже швидко конструювати,

застосовуючи майстер побудови мереж. Більш складні мережі пропонується формувати, використовуючи графічну оболонку блочного конструювання. І найбільш гнучкий та функціональний підхід відкриває можливість написання власного програмного коду за допомогою набору відповідних макрокоманд. У дослідженні ми використали майстер побудови спрощених одношарових мереж та власний програмний код. При потребі останній може бути оформлений у вигляді окремого додатка (програми) для запуску на операційних системах Windows або Linux. Так само відпрацьовані алгоритми можуть бути перенесені на інші програмні платформи для створення комерційного продукту IT-компаніями.

Першим етапом у створенні нейронної мережі є вибір її архітектури та методу навчання. З огляду на поставлені завдання нам потрібно побудувати класифікаційну або апроксимуючу мережу. Оскільки загалом залежність сили від навантаження не є складною функцією, то можна вибрати оптимальний набір вхідних метрик, який би забезпечив не тільки класифікацію кривих силової спектроскопії, але й дав можливість прогнозувати конкретні значення жорсткості контакту зонд — поверхня. Тому на першому етапі ми побудували одношарову апроксимуючу нейронну мережу. Для відпрацювання алгоритмів навчання тестова вибірка включала тисячу модельних кривих залежності сили навантаження від глибини проникнення зонду в поверхню при різних значеннях модуля Юнга поверхні, отриманих на серійній моделі сканувального зондового мікроскопа класу Dimension / Icon фірми “Bruker”. З метою уніфікації вхідних даних всі криві містили по 256 точок. При індентуванні задається максимальна сила навантаження, через те діапазон зміни сили на всіх кривих є однаковим, а глибина проникнення зонду в поверхню змінюється. Тому значення глибини містять більше унікальної інформації про криву і їх варто використовувати насамперед. Додатковим параметром для характеризації кривої ми вибрали площу під кривою. Як вихідний параметр було взято відповідні значення модуля Юнга. Таким чином, мережа містила 257 вхідних нейронів та один прихований шар. (Схема мережі наведена на *рис. 2.7.7*, набір вхідних кривих — на *рис. 2.7.8.*)

Результати тестування простої апроксимаційної мережі, навченої за різними алгоритмами, наведено на *рис. 2.7.9–2.7.11*. Чотири вікна

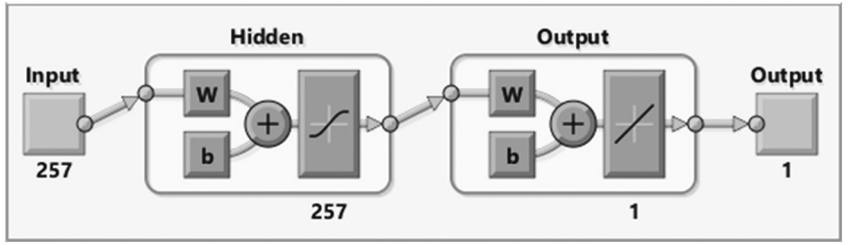

Рис. 2.7.7. Схема апроксимуючої нейронної мережі з великою кількістю вхідних нейронів

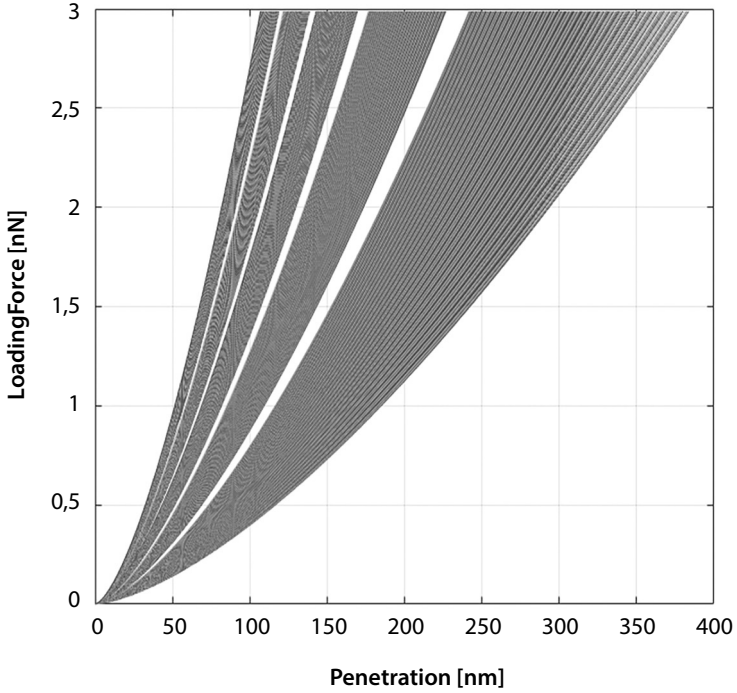

Рис. 2.7.8. Набір модельних силових кривих, що використовувався при тестуванні нейронної мережі

кожного рисунка відображають ступінь збігу результату передбачення, виданого мережею, із заданими значеннями в навчальному наборі та двох наборах вхідних даних, що не брали участі у навчанні — масив

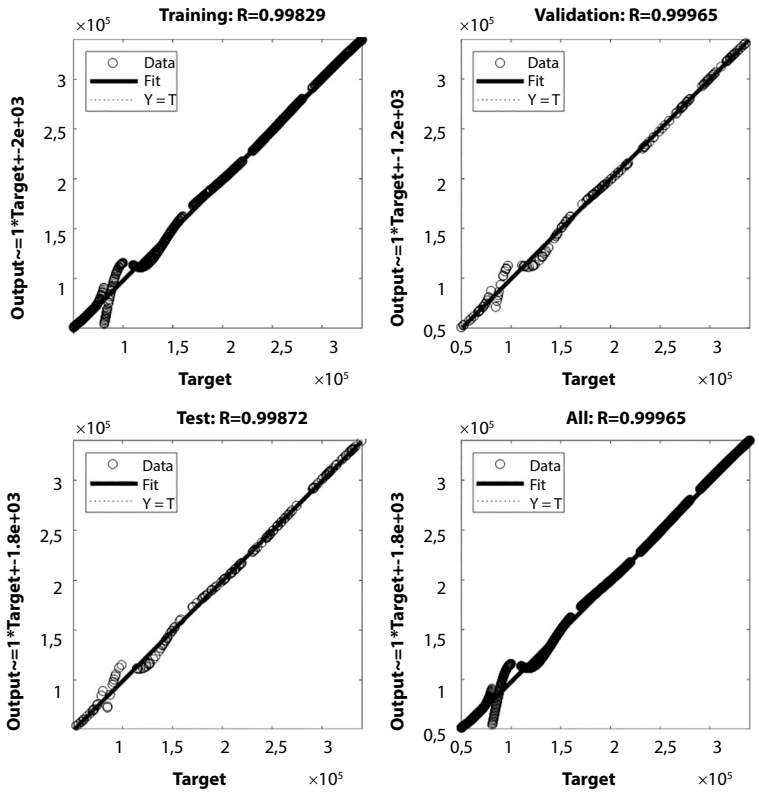

Рис. 2.7.9. Графіки регресій, що ілюструють результати апроксимації навчального та перевірочних наборів нейронною мережею, навченою за алгоритмом Байєсової регуляризації. На правому нижньому графіку представлений узагальнений результат

даних для перевірки в ході навчання та тестовий набір для перевірки після завершення процесу навчання. Дані представлені у вигляді регресії, де ідеальний результат, що повністю збігається з цілями навчання, розміщується вздовж діагональної прямої лінії. Для наших завдань доброю вважається підгонка із коефіцієнтом регресії R не меншим за 9,99.

Як видно з графіків на *рис. 2.7.9–2.7.11*, найкращого результату досягла мережа, навчена за алгоритмом спряжених градієнтів. Практично всі прогнозні значення зосереджені на прямій цілей. У представленні

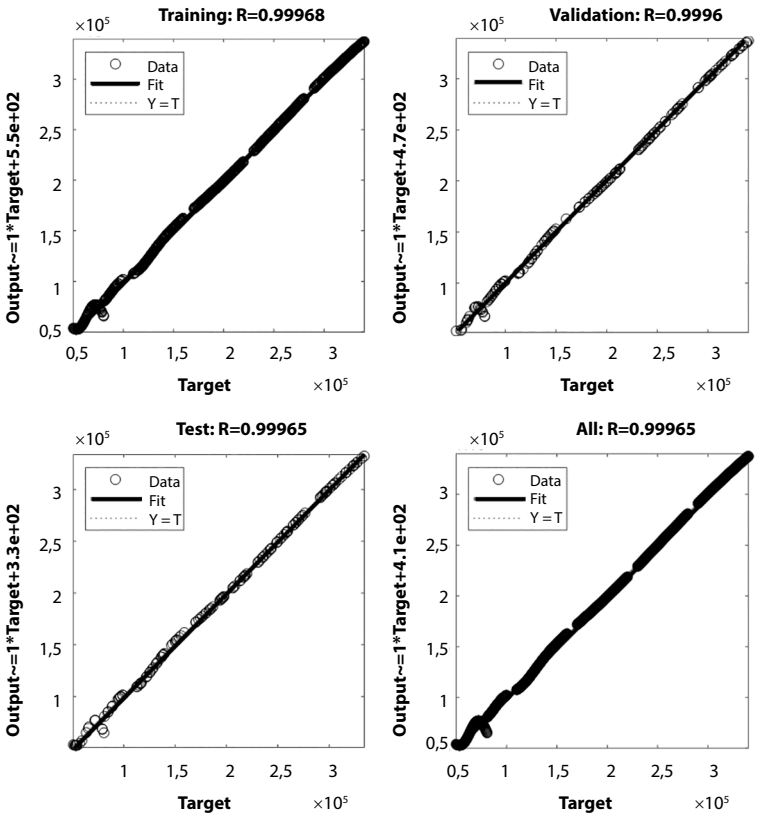

Рис. 2.7.10. Графіки регресій, що ілюструють результати апроксимації навчального та перевірочних наборів нейронною мережею, навченою за алгоритмом Левенберга — Марквардта

значень відносної похибки прогнозного та реального значень дані відображені на *рис. 2.7.12*. Видно, що найбільша похибка зосереджена в ділянці малих значень модуля Юнга. Це пов'язано з відносно малою кількістю даних для навчання в цьому діапазоні значень глибини проникнення зонду в поверхню. Крок зміни модуля Юнга в розрахункових кривих *рис. 2.7.8* вибрано постійним для всіх груп кривих. Таким чином група кривих із найбільшим модулем Юнга покриває діапазон значень глибини від 0 до 100 нм, у той час як така ж кількість кривих

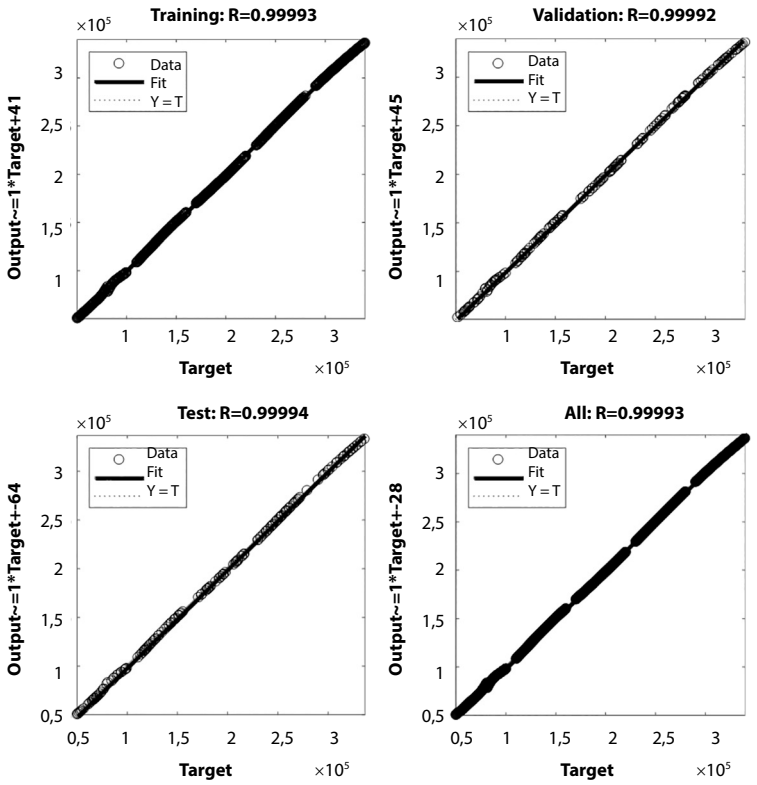

Рис. 2.7.11. Графіки регресій, що ілюструють результати апроксимації навчального та перевірочних наборів нейронною мережею, навченою за алгоритмом спряжених градієнтів

із найменшим модулем Юнга — діапазон значень глибини від 0 до 380 нм. Наведений приклад ілюструє важливість правильного добору і структурування даних для навчання нейронної мережі.

При роботі із масивом експериментальних кривих важливою є їх підготовка. Окрім згаданих вище калібрувань та вилучення складової кантилевера потрібно перенести початок системи координат в точку контакту зонду з поверхнею та коректно видалити апаратні шуми. Однак, не дивлячись на прискіпливу підготовку експериментальних

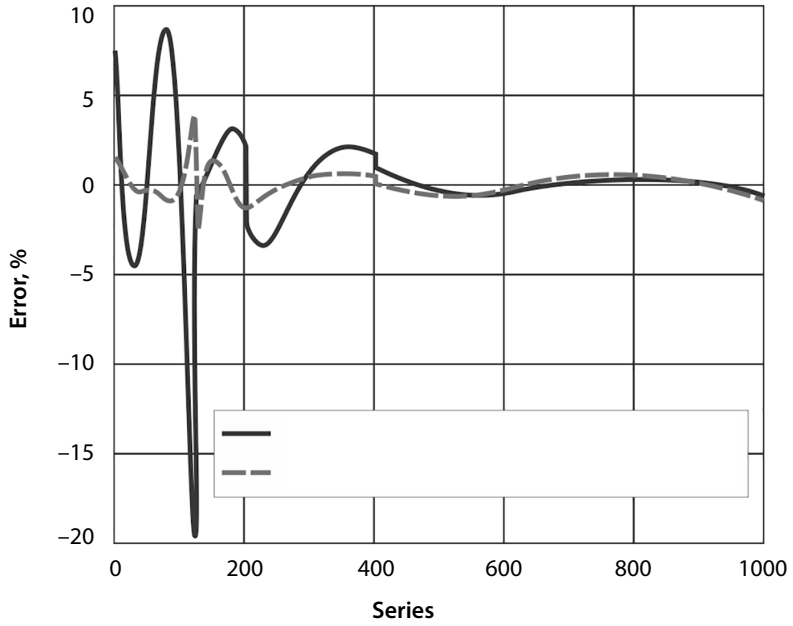

Рис. 2.7.12. Порівняння відносної похибки прогнозування значення модуля Юнга нейронними мережами, навченими за алгоритмами Левенберга — Марквардта та спряжених градієнтів

даних для використання нейронною мережею, результат виявися незадовільним (*рис. 2.7.13*). Мілка нейронна мережа не досягає поставлених цілей. Загальний коефіцієнт регресії є меншим за 0,92.

У цій ситуації звичним рішенням є конструювання складнішої багатошарової мережі з оптимальними активаційними функціями в шарах, краще структурування та збільшення числа даних для навчання.

На цьому етапі досліджень для коректної обробки експериментальних даних ми запрограмували власну нейронну мережу прямої передачі з двома прихованими шарами та логічною сигмоїдальною функцією активації. Число вхідних нейронів було зменшене до 4, що відповідало опису лінійної частини кривих навантаження та площі під ними. Найбільш оптимальним алгоритмом навчання виявся алгоритм Левенберга — Марквардта. Сумарний графік регресії нашої мережі наведено на *рис. 2.7.14*.

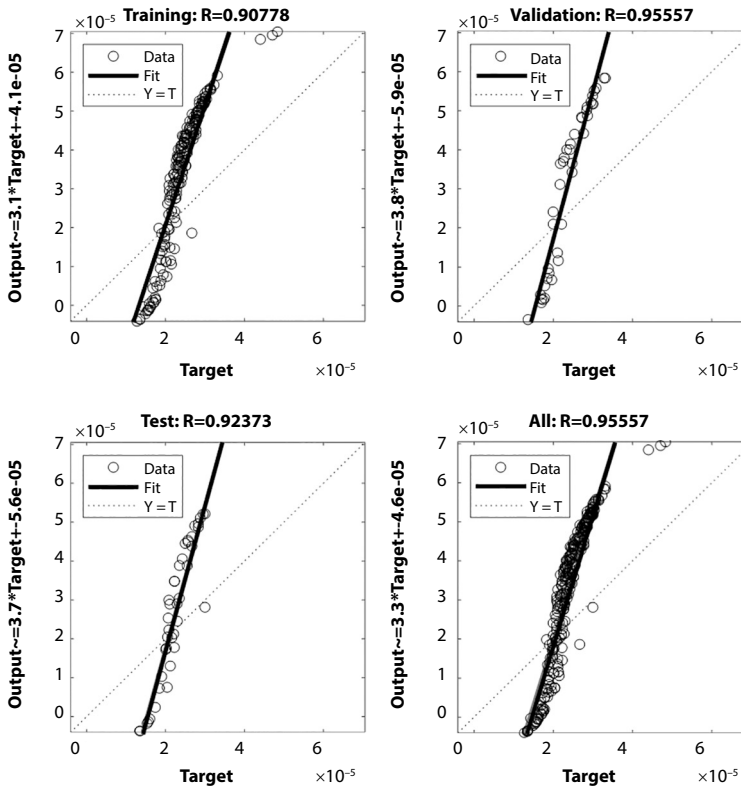

Рис. 2.7.13. Графіки регресій, що ілюструють результати апроксимації навчального та перевірочних наборів нейронною мережею, навченою за алгоритмом спряжених градієнтів на масиві експериментальних кривих

Коефіцієнт регресії є цілком прийнятним, а відносна похибка прогнозування менша за 2 % (*рис. 2.7.15*). Виключення складають елементи із великими значеннями модуля Юнга. Як видно з графіка на *рис. 2.7.16*, таких значень не багато і тому їх вага в мережі мінімальна.

Збільшення кількості кривих із такими високими значеннями модуля Юнга в навчальній вибірці легко підвищить точність прогнозу (*рис. 2.7.17*).

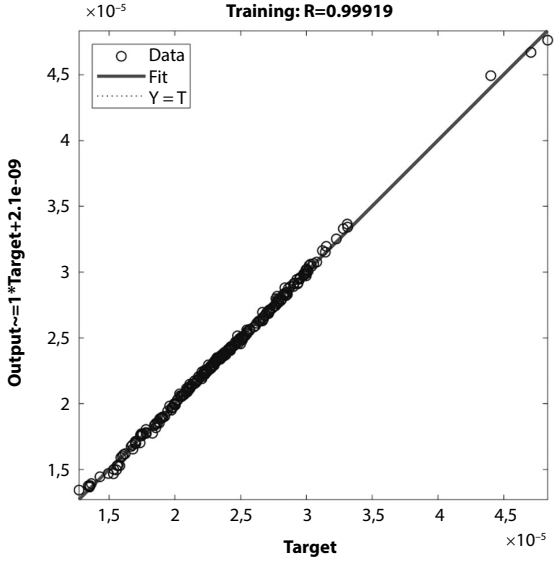

Рис. 2.7.14. Графік регресії запрограмованої нейронної мережі прямої передачі

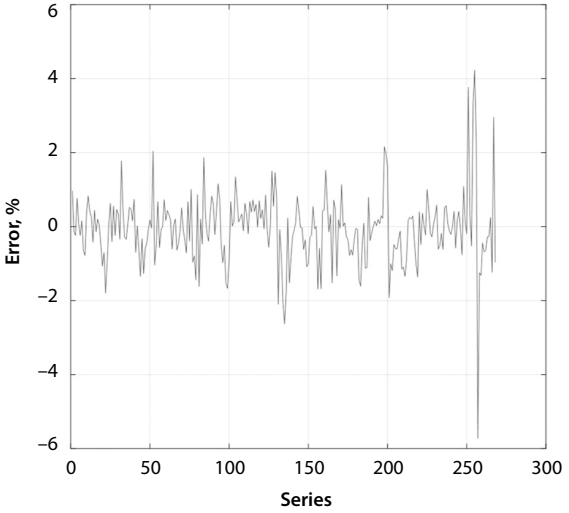

Рис. 2.7.15. Відносна похибка запрограмованої нейронної мережі прямої передачі в межах навчального масиву даних розміром 272×4

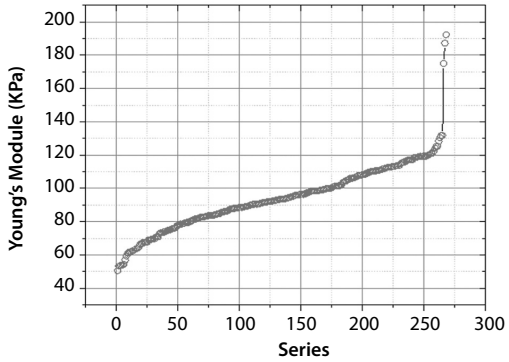

Рис. 2.7.16. Прогнозовані (лінія) та визначені за допомогою моделювання (точки) значення модуля Юнга

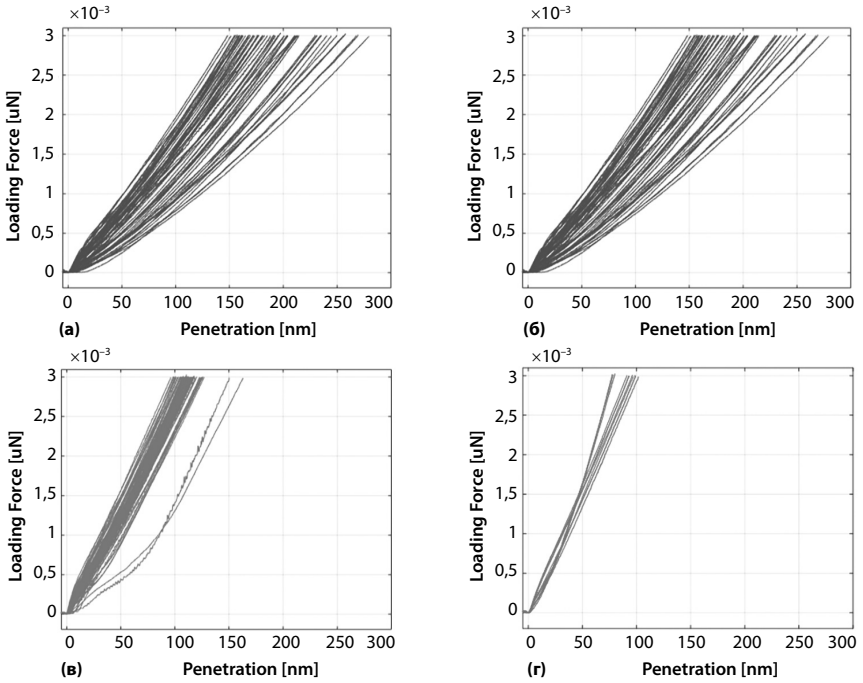

Рис. 2.7.17. Приклад класифікації експериментальних силових кривих нейронною мережею: (а) — масив експериментальних кривих; (б)—(г) — класифіковані криві відповідно до заданих значень модуля Юнга

Подяка. Дослідження, результати якого викладені в статті, частково проведено в рамках наукового проєкту «Розроблення комплексу нанометрики фізичних параметрів напівпровідникових та гібридних наносистем на основі електросилових та струмочутливих методів скануючої зондової мікроскопії», виконаного відповідно до Цільової програми наукових досліджень НАН України «Напівпровідникові матеріали, технології і датчики для технічних систем діагностики, контролю та управління», 2018–2020 рр.

ДЖЕРЕЛА

1. Ященко В. Искусственный интеллект. Теория. Моделирование. Применение. К: Логос, 2013. 294 с.

2. Sacha G.M., Varona P. Artificial intelligence in nanotechnology. *Nanotechnology*. 2013. V. 24, no. 45. P. 452002.

3. Falk D. How Artificial Intelligence Is Changing Science. *Quantamagazine*. URL: https://www.quantamagazine.org/how-artificial-intelligence-is-changing-science-20190311/

4. Graupe D. Principles of Artificial Neural Networks. Advanced Series in Circuits and Systems. 3rd edition. 2013. Vol. 7. 294 p.

5. Minelli E. et al. A fully-automated neural network analysis of AFM force-distance curves for cancer tissue diagnosis. *Applied Physics Letters*. 2017. Vol. 111, no. 14. P. 143701.

6. McCulloch W.S., Pitts W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*. 1943. Vol. 5, no. 4. P. 115–133.

7. Хайкин С. Нейронные сети: полный курс. 2-е издание. Москва — Санкт-Петебург — Киев: Издательский дом Вильямс, 2016. 1104 с.

8. Rosenblatt F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*. 1958. Vol. 65, no. 6. P. 386–408.

9. Rosenblatt F. Principles of Neurodymanics: Perceptrons and the Theory of Brain Mechanisms. Spartan books, 1962. 616 p.

10. Новотарський М., Нестеренко Б. Штучні нейронні мережі: обчислення. *Праці Інституту математики НАН України.* К.: Ін-т математики НАН України, 2004. Т. 50. 408 с.

11. Minsky M., Papert S.A. Perceptrons: An introduction to computational geometry. Expanded Edition. MIT press, 1987. 308 p.

12. Mitchell T.M. Machine learning. Burr Ridge, IL: McGraw Hill, 1997. P. 870–877.

13. Widrow B., Sterns S.D. Adaptive Signal Processing. Englewood Cliffs, NJ, USA: Prentice-Hall, 1985. 247 p.

14. Cybenko G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*. 1992. Vol. 5, no. 4. P. 455–455.

15. Montúfar G.F., Pascanu R., Cho K.H., Bengio Y. On the number of linear regions of deep neural networks. *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems*, 2014. P. 2924–2932.

16. Галушкин А.И. Синтез многослойных систем распознавания образов. Москва: Энергия, 1974. 368 с.

17. Rumelhart D.E., Hinton G.E., Williams R.J. Learning internal representations by error propagation. Parallel distributed processing: explorations in the microstructure of cognition, vol. 1: foundations. California Univ San Diego La Jolla Inst for Cognitive Science, 1986. P. 318–362.

18. Яхъева Г.Э. Основы теории нейронных сетей. Москва: Национальный открытый ун-т «Интуит», 2016. 200 с.

19. Levenberg K. A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*. 1944. Vol. 2, no. 2. P. 164–168.

20. Marquardt D.W. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*. 1963. Vol. 11, no. 2. P. 431–441.

21. Hagan M.T., Demuth H.B. Neural networks for control. *Proceedings of the American Control Conference*. 1999. Vol. 3. P. 1642–1656.

22. Møller M.F. A scaled conjugate gradient algorithm for fast supervised learning. *Neural networks*. 1993. Vol. 6, no. 4. P. 525–533.

23. Lytvyn P.M. Scanning Probe Microscopy in Practical Diagnostic. 3D Topography Imaging and Nanometrology. Springer, 2014. P. 179–219.

24. Marvin L. Neural networks with MATLAB. CreateSpace Independent Publishing Platform, 2016. 231 p.