ЛЕКЦІЯ 4.
ОСНОВНІ ТИПИ НЕЙРОННИХ МЕРЕЖ

Питання лекції

1. Перцептрон Розенблата
2. Нейромережа зворотного поширення похибки (Back Propagation)
3. Мережа Delta Bar Delta
4. Мережа Extended Delta Bar Delta
5. Мережа спрямованого випадкового пошуку
6. Нейрона мережа вищого порядку або функціонально - зв'язана нейрона мережа
7. Мережа Кохонена
8. Мережа квантування навчального вектора (Learning VectorQuantization)

ВСТУП

За архітектурою зв'язків, більшість відомих нейромереж можна згрупувати у два великих класи:

Мережі прямого поширення (з односкерованими послідовними зв'язками).

Мережі зворотного поширення (з рекурентними зв'язками).

Типові архітектури нейронних мереж

Мережі прямого поширення	**Рекурентні мережі**
Перцептрони	Мережа Хопфілда
Мережа Back Propagation	Мережа Хемінга
Мережа зустрічного поширення	Мережа адаптивної резонансної теорії
Карта Кохонена	Двоскерована асоціативна пам'ять

Мережі прямого поширення відносять до статичних, тут на входи нейронів надходять вхідні сигнали, які не залежать від попереднього стану мережі.

Рекурентні мережі вважаються динамічними, оскільки за рахунок зворотних зв'язків (петель) входи нейронів модифікуються в часі, що призводить до зміни станів мережі.

1. Перцептрон Розенблата

Першою моделлю нейромереж вважають перцептрон Розенблата. Теорія перцептронів є основою для багатьох типів штучних нейромереж прямого поширення і вони є класикою для вивчення.

Одношаровий перцептрон здатний розпізнавати найпростіші образи. Окремий нейрон обчислює зважену суму сигналів вхідних елементів, віднімає значення зсуву і пропускає результат через жорстку порогову функцію, вихід якої дорівнює +1 чи -1. В залежності від значення вихідного сигналу приймається рішення:

+1 - вхідний сигнал належить до класу А,

-1 - вхідний сигнал належить до класу В.

На рис. 5.1 показано схему одношарового перцептрона, графік передатної функції і схему вирішальних областей, створених у багатовимірному просторі вхідних сигналів. Вирішальні області визначають, які вхідні образи будуть віднесені до класу А, які - до класу В. Перцептрон, що складається з одного нейрона, формує дві вирішальні області, які розділено гіперплощиною.

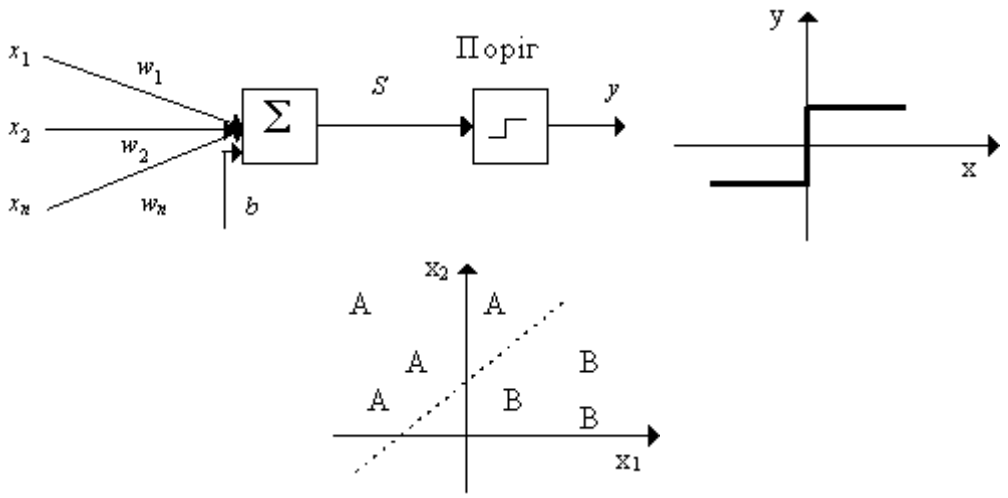

Рис. 5. 1. Схема нейрона, графік передатної функції і поділяюча поверхня

На рис.5.1 показано випадок з розмірністю вихідного сигналу - 2. Поділяюча поверхня є прямою лінією на площині. Рівняння, що задає поділяючу пряму, залежить від значень синаптичних ваг і зсуву.

Алгоритм навчання одношарового перцептрона

1. Ініціалізація синаптичних ваг і зсуву: синаптичні ваги приймають малі випадкові значення.

2. Пред'явлення мережі нового вхідного і бажаного вихідного сигналів: вхідний сигнал $x=(x_1, x_2, ..., x_n)$ пред'являється нейрону разом з бажаним вихідним сигналом d.

3. Обчислення вихідного сигналу нейрона:

$$y(t) = f\left(\sum_{i=1}^{N} w_i(t)x_i(t) - b\right)$$

4. Налаштування значень ваг:

$$w_i(t+1)=w_i(t)+r[d(t)-y(t)]x_i(t),\ i=1, ..., N$$

$$d(t) = \begin{cases} +1, \text{вихідний клас A} \\ -1, \text{вихідний клас B} \end{cases}$$

де $w_i(t)$ - вага зв'язку від i-го елемента вхідного сигналу до нейрона в момент часу t,

r - швидкість навчання (менше 1);

$d(t)$ - бажаний вихідний сигнал.

Якщо мережа приймає правильне рішення, синаптичні ваги не модифікуються.

5. Перехід до кроку 2.

Тип вхідних сигналів: бінарні чи аналогові (дійсні).

Розмірності входу і виходу обмежені при програмній реалізації тільки можливостями обчислювальної системи, на якій моделюється нейронна мережа, при апаратній реалізації - технологічними можливостями.

Області застосування: розпізнавання образів, класифікація.

Недоліки. Примітивні поділяючі поверхні (гіперплощини) дають можливість вирішувати лише найпростіші задачі розпізнавання.

Переваги. Програмні та апаратні реалізації моделі прості. Простий і швидкий алгоритм навчання.

Модифікації. Багатошарові перцептрони дають можливість будувати складні поділяючі поверхні і є більш поширеними.

На рис.5.2 показана схема перцептрона з декількома входами та виходами.

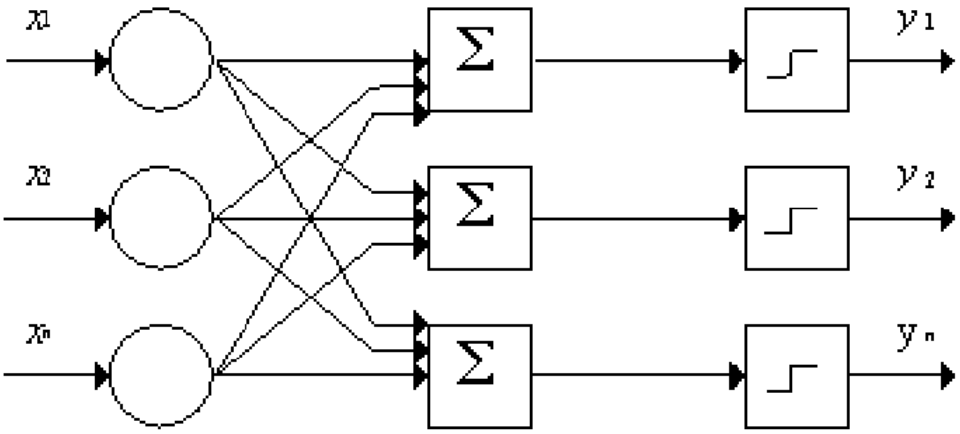

Рис. 5.2. Перцептрон з декількома входами та виходами

2. Нейромережа зворотного поширення похибки (Back Propagation)

Архітектура *FeedForward BackPropagation* була розроблена на початку 1970-х років декількома незалежними авторами: Вербор (*Werbor*); Паркер (*Parker*); Румельгарт (*Rumelhart*), Хінтон (*Hinton*) та Вільямс (*Williams*). На даний час, парадигма *BackPropagation* є популярною, ефективною та легкою моделлю навчання для складних, багатошарових мереж. Вона використовується в різних типах застосувань і породила великий клас нейромереж з різними структурами та методами навчання.

Типова мережа *BackPropagation* має вхідний прошарок, вихідний прошарок та принаймні один прихований прошарок. Теоретично, обмежень відносно числа прихованих прошарків не існує, але практично застосовують один або два. На рис. 5.3представлена схема багатошарового перцептрона

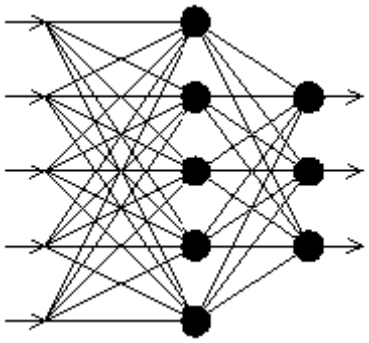

Рис. 5.3 Схема багатошарового перцептрона

Нейрони організовано в пошарову структуру з прямою передачею (вперед) сигналу. Кожний нейрон мережі продукує зважену суму своїх входів, пропускає цю величину через передатну функцію і видає вихідне значення. Мережа може моделювати функцію практично будь якої складності, причому число прошарків і число нейронів у кожному прошарку визначають складність функції.

Важливим при моделюванні мережі є визначення числа проміжних прошарків і числа нейронів в них. Більшість дослідників та інженерів використовують загальні правила, зокрема:

Кількість входів та виходів мережі визначаються кількістю вхідних та вихідних параметрів досліджуваного об'єкту, явища, процесу, тощо. На відміну від зовнішніх

прошарків, число нейронів прихованого прошарку $n_{прих}$ обирається емпіричним шляхом. В більшості випадків достатньою кількість нейронів буде $n_{прих} <= n_{вх} + n_{вих}$, де $n_{вх}$, $n_{вих}$ - кількість нейронів у вхідному і, відповідно, у вихідному прошарках.

Якщо складність у відношенні між отриманими та бажаними даними на виході збільшується, кількість нейронів прихованого прошарку повинна також збільшитись.

Якщо процес, що моделюється, може розділятись на багато етапів, потрібен додатковий прихований прошарок (прошарки). Якщо процес не розділяється на етапи, тоді додаткові прошарки можуть допустити перезапам'ятовування і, відповідно, невірне загальне рішення.

Після того, як визначено число прошарків і число нейронів в кожному з них, потрібно знайти значення для синаптичних ваг і порогів мережі, які спроможні мінімізувати похибку спродукованого результату. Саме для цього існують алгоритми навчання, де відбувається підгонка моделі мережі до наявних навчальних даних. Похибка для конкретної моделі мережі визначається шляхом проходження через мережу всіх навчальних прикладів і порівняння спродукованих вихідних значень з бажаними значеннями. Множина похибок створює функцію похибок, значення якої можна розглядати, як похибку мережі. В якості функції похибок найчастіше використовують суму квадратів похибок.

Для кращого розуміння алгоритму навчання мережі *Back Propagation* потрібно роз'яснити поняття поверхні станів. Кожному значенню синаптичних ваг і порогів мережі (вільних параметрів моделі кількістю N) відповідає один вимір в багатовимірному просторі. N+1-ий вимір відповідає похибці мережі. Для різноманітних сполучень ваг відповідну похибку мережі можна зобразити точкою в N+1-вимірному просторі, всі ці точки утворюють деяку поверхню - поверхню станів. Мета навчання нейромережі полягає в знаходженні на багатовимірній поверхні найнижчої точки.

Поверхня станів має складну будову і досить неприємні властивості, зокрема, наявність локальних мінімумів (точки, найнижчі в своєму певному околі, але вищі від глобального мінімуму), пласкі ділянки, сідлові точки і довгі вузькі яри. Аналітичними засобами неможливо визначити розташування глобального мінімуму на поверхні станів, тому навчання нейромережі по суті полягає в дослідженні цієї поверхні.

Відштовхуючись від початкової конфігурації ваг і порогів (від випадково обраної точки на поверхні), алгоритм навчання поступово відшукує глобальний мінімум. Обчислюється вектор градієнту поверхні похибок, який вказує напрямок найкоротшого спуску по поверхні з заданої точки. Якщо трошки просунутись по ньому, похибка зменшиться. Зрештою алгоритм зупиняється в нижній точці, що може виявитись лише локальним мінімумом (в ідеальному випадку - глобальним мінімумом).

Складність полягає у виборі довжини кроків. При великій довжині кроку збіжність буде швидшою, але є небезпека перестрибнути рішення, або піти в неправильному напрямку. При маленькому кроці, правильний напрямок буде виявлено, але зростає кількість ітерацій. На практиці розмір кроку береться пропорційним крутизні схилу з деякою константою - швидкістю навчання. Правильний вибір швидкості навчання залежить від конкретної задачі і здійснюється дослідним шляхом. Ця константа може також залежати від часу, зменшуючись по мірі просування алгоритму.

Алгоритм діє ітеративно, його кроки називаються епохами. На кожній епосі на вхід мережі по черзі подаються всі навчальні приклади, вихідні значення мережі порівнюються з бажаними значеннями і обчислюється похибка. Значення похибки, а також градієнту поверхні станів використовують для корекції ваг, і дії повторюються. Процес навчання припиняється або коли пройдена визначена кількість епох, або коли похибка досягає визначеного рівня малості, або коли похибка перестає зменшуватись (користувач переважно сам вибирає потрібний критерій зупинки).

Алгоритм навчання мережі

1. Ініціалізація мережі: вагові коефіцієнти і зсуви мережі приймають малі випадкові значення.

2. Визначення елемента навчальної множини: (вхід - вихід). Входи (x_1, x_2... x_N), повинні розрізнятися для всіх прикладів навчальної множини.

3. Обчислення вихідного сигналу:

$$S_{i_m} = \sum_{i_{m-1}}^{N_{m-1}} W_{i_m i_{m-1}} y_{i_{m-1}} - b_{i_m}$$

$$y_{im} = f(S_{jm})$$

$$i_m = 1, 2, ..., N_m,\ m = 1, 2, ..., L$$

де S - вихід суматора, w - вага зв'язку, y - вихід нейрона, b - зсув, i - номер нейрона, N - число нейронів у прошарку, m - номер прошарку, L - число прошарків, f- передатна функція.

4. Налаштування синаптичних ваг:

$$w_{ij}(t+1) = w_{ij}(t) + r\, g_j x'_i$$

де w_{ij} - вага від нейрона i або від елемента вхідного сигналу i до нейрона j у момент часу t, x_i' - вихід нейрона i, r - швидкість навчання, g_j - значення похибки для нейрона j.

Якщо нейрон з номером j належить останньому прошарку, тоді

$$g_j = y_j(1-y_j)(d_j-y_j)$$

де d_j - бажаний вихід нейрона j, y_j - поточний вихід нейрона j.

Якщо нейрон з номером j належить одному з прошарків з першого по передостанній, тоді

$$g_j = x'_j(1 - x'_j)\sum_k g_k w_{jk}$$

де k пробігає всі нейрони прошарку з номером на одиницю більше, ніж у того, котрому належить нейрон j.

Зовнішні зсуви нейронів b налаштовуються аналогічним образом.

Тип вхідних сигналів: цілі чи дійсні.

Тип вихідних сигналів: дійсні з інтервалу, заданого передатною функцією нейронів.

Тип передатної функції: сигмоїдальна. Сигмоїдальні функції є монотонно зростаючими і мають відмінні від нуля похідні по всій області визначення. Ці характеристики забезпечують правильне функціонування і навчання мережі.

Області застосування. Розпізнавання образів, класифікація, прогнозування.

Недоліки. Низька швидкість навчання.

Переваги. Ефективний та популярний алгоритм для вирішення численних практичних задач.

Модифікації. Модифікації алгоритму зворотного поширення зв'язані з використанням різних функцій похибки, різних процедур визначення напрямку і величини кроку.

3. Мережа Delta Bar Delta

Мережа Delta bar Delta була розроблена Робертом Джекобсом (Robert Jacobs), для поліпшення оцінки навчання стандартних мереж Feed Forward і є модифікацією мережі Back Propagation.

Процедура Back Propagation базується на підході крутого спуску, що мінімізує похибку мережі під час процесу зміни синаптичних ваг. Стандартні оцінки навчання застосовуються на базисі "шар за шаром" і значення моменту призначаються глобально. Моментом вважається фактор, що використовується для згладжування оцінки навчання. Момент додається до стандартної зміни ваги і пропорційний до попередньої зміни ваги.

Хоча цей метод успішний у розв'язанні багатьох задач, збіжність процедури занадто повільна для використання. Delta bar Delta має "неформальний" підхід до навчання штучних мереж, при якому кожна вага має свій власний самоадаптований фактор навчання і минулі значення похибки використовуються для обчислення майбутніх значень. Знання ймовірних похибок дозволяє мережі робити інтелектуальні кроки при зміні ваг, але процес ускладнюється тим, що кожна вага може мати зовсім різний вплив на загальну похибку. Джекобс запропонував поняття "здорового глузду", коли кожна вага з'єднання мережі повинна мати власну оцінку навчання, а розмір кроку, що призначений одній вазі з'єднання не застосовується для усіх ваг у шарі.

Оцінка навчання будь-якої ваги з'єднання змінюється на основі інформації про поточну похибку, знайденої зі стандартної Backpropagation. Якщо локальна похибка має однаковий знак для декількох послідовних часових кроків, оцінка навчання для цього з'єднання лінійно збільшується. Якщо локальна похибка часто змінює знак, оцінка навчання зменшується геометрично і це гарантує, що оцінки навчання з'єднання будуть завжди додатними. Оцінкам навчання дозволено мінятися в часі.

Призначення оцінки навчання до кожного з'єднання, дозвіл цій оцінці навчання неперервно мінятися з часом обумовлюють зменшення часу збіжності.

З розрішенням різних оцінок навчання для будь-якої ваги з'єднання в мережі, пошук крутого спуску може не виконуватися. Замість цього, ваги з'єднань змінюються на основі часток похідних похибок щодо самої ваги й оцінки "кривизни поверхні похибки" поблизу поточної точки. Зміни ваг відповідають обмеженню місцевості і вимагають інформацію від нейронів, з якими вони з'єднані.

Переваги. Парадигма Delta Bar Delta є спробою прискорити процес збіжності алгоритму зворотного поширення за рахунок використання додаткової інформації про зміну параметрів і ваг під час навчання.

Недоліки

• Навіть невелике лінійне збільшення коефіцієнта може привести до значного росту швидкості навчання, що викликає стрибки в просторі ваг.

• Геометричне зменшення коефіцієнта іноді буває недостатньо швидким.

4. Мережа Extended Delta Bar Delta

Елі Минай (Ali Minai) і Рон Вільямс (Ron Williams) розробили алгоритм роботи мережі Eextended Delta bar Delta, як природне продовження роботи Джекобса. В алгоритм вбудовується пам'ять з особливістю відновлення. Після кожного представлення навчальні дані епохи, оцінюється накопичена похибка. Якщо похибка менша за попередню мінімальну похибку, ваги зберігаються в пам'яті, як найкращі на цей час. Параметр допуску керує фазою відновлення. У випадку, якщо поточна похибка перевищує мінімальну попередню похибку, модифіковану параметром допуску, усі значення ваг з'єднань стохастично повертаються до збереженої в пам'яті найкращої множини ваг.

5. Мережа спрямованого випадкового пошуку

Мережа спрямованого випадкового пошуку (Directed Random Search), використовує стандартну архітектуру FeedForward, що не базується на алгоритмі BackProragation і

коректує ваги випадковим чином. Для забезпечення порядку в такому процесі, до випадкового кроку додається компонент напрямку, що гарантує напрямок ваг до попередньо успішного напрямку пошуку. Вплив на нейрони здійснюється окремо. Для збереження ваг усередині компактної області, де алгоритм працює добре, установлюють верхню границю величини ваг. Установлюючи границі ваг великими, мережа може продовжувати працювати, оскільки дійсний глобальний оптимум залишається невідомим.

Іншою особливістю правила навчання є початкова відмінність у випадковому розподілі ваг. У більшості комерційних пакетів існує рекомендоване розроблювачем число для параметра початкової відмінності. Парадигма випадкового пошуку має кілька важливих рис. Вона швидка і легка у використанні, найкращі результати виходять, коли початкові ваги знаходяться близько до найкращих ваг. Швидкою парадигма є завдяки тому, що для проміжних нейронів похибки не обчислюються, а обчислюється лише вихідна похибка. Алгоритм дає ефект лише в невеликій мережі, оскільки при збільшенні числа з'єднань, процес навчання стає довгим і важким. Існує *чотири ключових компоненти мережі* з випадковим пошуком. Це - **випадковий крок, крок реверсування, спрямований компонент і самокоригувальна відмінність**.

Випадковий крок. До будь-якої ваги додається випадкова величина. Уся навчальна множина пропускається через мережу, створюючи "похибку пророкування". Якщо нова загальна похибка навчальної множини менша ніж попередня найкраща похибка пророкування, поточні значення ваг, що включають випадковий крок, стають новою множиною "найкращих" ваг.

Поточна похибка пророкування зберігається як нова, найкраща похибка пророкування.

Крок реверсування. Якщо результати випадкового кроку гірші попередньої найкращої похибки, випадкова величина віднімається від початкового значення ваги. Це створює множину ваг, що знаходяться в протилежному напрямку до попереднього випадкового кроку. Якщо загальна " похибка пророкування" менше попередньої найкращої похибки, поточне значення ваг і поточна похибка пророкування зберігаються як найкращі. Якщо і прямий і зворотний кроки не поліпшують результат, до найкращих ваг додається цілком нова множина випадкових значень і процес починається спочатку.

Спрямований компонент. Для збіжності мережі створюється множина спрямованих компонентів, отриманих за результатами прямого і зворотного кроків. Спрямовані компоненти, що відображають ланцюг успіхів або невдач попередніх випадкових кроків, додаються до випадкових компонентів на кожнім кроці процедури і забезпечують елемент "здорового глузду" у пошуку.

Доведено, що додавання спрямованих компонентів забезпечує різке підвищення ефективності алгоритму.

Самокоригувальна відмінність. Визначається параметр початкової відмінності для керування початковим розміром випадкових кроків, що додається до ваг. Адаптивний механізм змінює параметр відмінності, що базується на поточній оцінці успіху або невдачі. Правило навчання припускає, що поточний розмір кроків для ваг у правильному напрямку збільшується для випадку декількох послідовних успіхів. Навпаки, якщо відбувається кілька послідовних невдач, відмінність зменшується для зменшення розміру кроку.

Переваги. Для невеликих і середніх нейромереж, спрямований випадковий пошук дає гарні результати за короткий час. Навчання автоматичне, вимагає невеликої взаємодії з користувачем.

Недоліки. Кількість ваг з'єднань накладає практичні обмеження на розмір задачі. Якщо мережа має більше чим 200 ваг з'єднань, спрямований випадковий пошук може вимагати збільшення часу навчання, але продукувати прийнятні рішення.

6. Нейрона мережа вищого порядку або функціонально - зв'язана нейрона мережа

Функціонально-зв’язані мережі були розроблені Йох-Хан Пао (Yoh-Han Pao) і детально описані в його книзі "Адаптивне розпізнавання образів і нейроні мережі" (Adaptive Pattern Recognition and Neural Networks). Нейромережа розширює стандартну архітектуру FeedForward BackPropagation модифікацією вузлів на вхідному шарі. Входи комбінуються математичним шляхом за допомогою функцій вищого порядку, таких як квадрати, куби або синуси і розширюють сприйняття мережею заданої проблеми. З назв цих функцій вищого порядку або функціонально-зв’язаних входів і випливає назва нейромереж.

Існує два основних способи додавання вхідних вузлів. У першому, у модель можуть додаватися перехресні добутки вхідних елементів, це називається вхідним добутком або тензорною моделлю, де кожен компонент вхідного образа перемножується з усіма компонентами вхідного вектора.

Наприклад, для мережі Backpropagation із трьома входами (A, B і C), перехресними добутками будуть AA, BB, CC, AB, AC і BC (елементи другого порядку). Також можуть додаватися елементи третього порядку, такі як ABC.

Другим методом для додавання вхідних вузлів є функціональне розширення базових входів. У випадку моделі з входами A, B і C, її можна перетворити в модель нейроної мережі вищого порядку з входами: A, B, C, SIN(A), COS(B), LOG(C), MAX(A,B,C) і ін. Повний ефект повинний забезпечити мережа з об'єднанням моделі тензорного і функціонального розширення. Ніякої нової інформації не додається, але розширене представлення входів робить мережу простішою для навчання. Існують обмеження для цієї моделі. Для перетворення початкових входів необхідна обробка більшої кількості вхідних вузлів, що впливають на швидкодію мережі, тому при розширенні входів повинно бути враховане об'єднання точного розв’язання і порівняно невеликого часу навчання.

7. Мережа Кохонена

Мережа розроблена Тойво Кохоненом на початку 1980-х рр. і принципово відрізняється від розглянутих вище мереж, оскільки використовує неконтрольоване навчання і навчальна множина складається лише із значень вхідних змінних.

Мережа розпізнає кластери в навчальних даних і розподіляє дані до відповідних кластерів. Якщо в подальшому мережа зустрічається з набором даних, несхожим ні з одним із відомих зразків, вона відносить його до нового кластеру. Якщо в даних містяться мітки класів, то мережа спроможна вирішувати задачі класифікації. Мережі Кохонена можна використовувати і в задачах, де класи є відомими - перевага буде у спроможності мережі виявляти подібність між різноманітними класами.

Мережа Кохонена має лише два прошарки: вхідний і вихідний (рис. 5.4.) її ще називають **самоорганізовуваною картою**. Елементи карти розташовуються в деякому просторі, як правило, двовимірному. Мережа Кохонена навчається методом послідовних наближень. У процесі навчання на входи подаються дані, але *мережа при цьому підлаштовується не під еталонне значення виходу, а під закономірності у вхідних даних*. Починається навчання з вибраного випадковим чином вихідного розташування центрів.

В процесі послідовної подачі на вхід мережі навчальних прикладів визначається найбільш схожий нейрон (той, у якого скалярний добуток ваг і поданого на вхід вектора є мінімальним). Цей нейрон оголошується переможцем і є центром при підлаштуванні ваг в сусідніх нейронів. Таке правило навчання передбачає "змагальне" (від «змагання») навчання з врахуванням відстані нейронів від "нейрона-переможця".

Навчання при цьому полягає не в мінімізації помилки, а в підлаштуванні ваг (внутрішніх параметрів нейронної мережі) для найбільшого збігу з вхідними даними.

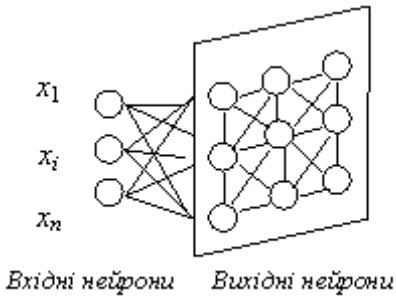

Рис. 5.4. Мережа Кохонена

Мережа Кохонена навчається методом послідовних наближень. Починаючи з випадковим чином обраного вихідного розташування центрів, алгоритм поступово поліпшується для кластеризації навчальних даних. Основний ітераційний алгоритм Кохонена послідовно проходить ряд епох, на кожній з яких обробляється один приклад з навчальної вибірки. Вхідні сигнали послідовно пред'являються мережі, при цьому бажані вихідні сигнали не визначаються. Після пред'явлення достатнього числа вхідних векторів синаптичні ваги мережі стають здатні визначити кластери. Ваги організуються так, що топологічно близькі вузли реагують до схожих вхідних сигналів.

В результаті роботи алгоритму центр кластера встановлюється в певній позиції, яка задовольняє кластеризовані приклади, для яких даний нейрон є "переможцем". В результаті навчання мережі необхідно визначити міру сусідства нейронів, тобто окіл нейрона-переможця, який представляє кілька нейронів, що оточують нейрон-переможець.

Для реалізації алгоритму необхідно визначити міру сусідства нейронів (околиця нейрона-переможця). На рис. 2.37 показані зони топологічного сусідства нейронів на карті ознак у різні моменти часу. NEj(t) - множина нейронів, що вважаються сусідами нейрона j у момент часу t. Зони сусідства зменшуються з часом.

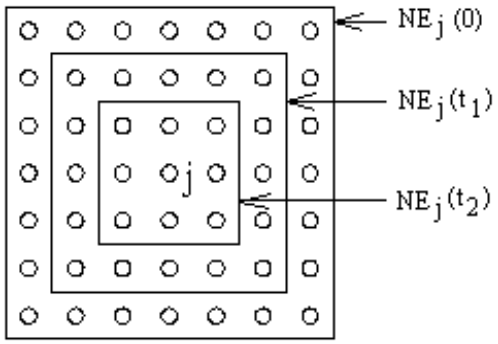

Рис. 5.5. Зони топологічного сусідства на карті ознак у різні моменти часу

Спочатку до околу належить велике число нейронів, далі її розмір поступово зменшується. Мережа формує топологічну структуру, в якій схожі приклади утворюють групи прикладів, які близько знаходяться на топологічної карті.

Алгоритм функціонування мережі Кохонена:

1. *Ініціалізація мережі*. Ваговим коефіцієнтам мережі надаються малі випадкові значення. Заються невеликі випадкові значення. Загальне число синаптичних ваг - M*N (див. рис. 5.4). Початкова зона сусідства показана на рис. 5.5.

2. *Пред'явлення мережі нового вхідного сигналу*.

3. *Обчислення відстані до всіх нейронів мережі*:
Відстані d_j від вхідного сигналу до кожного нейрона j визначаються за формулою:

$$d_j = \sum_{i=1}^{N} (x_i(t) \cdot w_{ij}(t))^2$$

де x_i - *i-ий* елемент вхідного сигналу в момент часу t, $w_{ij}(t)$ - вага зв'язку від *i-го* елемента вхідного сигналу до нейрона j у момент часу t.

4. *Вибір нейрона з найменшою відстанню*:
Вибирається нейрон-переможець j^*, для якого відстань d_j найменше.

5. *Налаштування ваг нейрона j^* і його сусідів*:
Робиться налаштування ваг для нейрона j^* і всіх нейронів з його околу NE. Нові значення ваг:

$$w_{ij}(t+1)=w_{ij}(t)+r(t)(x_i(t)-w_{ij}(t))$$

де $r(t)$ - швидкість навчання, що зменшується з часом (додатне число, менше одиниці).

6. *Повернення до кроку 2*.

В алгоритмі використовується коефіцієнт швидкості навчання, який поступово зменшується, для тонкішої корекції в новій епосі. В результаті позиція центру встановлюється в певній позиції. Вона задовільним чином кластеризує приклади, для яких даний нейрон є переможцем.

Властивість топологічної впорядкованості досягається в алгоритмі за допомогою використання поняття околу. **Окіл** - це декілька нейронів, що оточують нейрон-переможець. Відповідно до швидкості навчання, розмір околу поступово зменшується, так, що спочатку до нього належить досить велике число нейронів (можливо вся карта), на самих останніх етапах окіл стає нульовим і складається лише з нейрона-переможця.

В алгоритмі навчання корекція застосовується не тільки до нейрона-переможця, але і до всіх нейронів з його поточного околу. В результаті такої зміни околу, початкові доволі великі ділянки мережі мігрують в бік навчальних прикладів. Мережа формує грубу структуру топологічного порядку, при якій схожі приклади активують групи нейронів, що близько знаходяться на топологічній карті.

З кожною новою епохою швидкість навчання і розмір околу зменшуються, тим самим всередині ділянок карти виявляються більш тонкі розходження, що зрештою призводить до точнішого налаштування кожного нейрона.

Часто навчання зумисне розбивають на дві фази: більш коротку, з великою швидкістю навчання і великих околів, і більш тривалу з малою швидкістю навчання і нульовими або майже нульовими околами.

Після того, як мережа навчена розпізнавати структури даних, її можна використовувати як засіб візуалізації при аналізі даних.

Області застосування. Кластерний аналіз, розпізнавання образів, класифікація.

Недоліки. Мережа може бути використана для кластерного аналізу маючи заздалегідь відоме число кластерів.

Переваги. Мережа Кохонена здатна функціонувати в умовах завад, тому що число кластерів фіксоване, ваги модифікуються повільно, налаштування ваг закінчується після навчання.

8. Мережа квантування навчального вектора (Learning VectorQuantization)

Мережа була запропонована Тойво Кохоненом у середині 80-х рр., пізніше, ніж його робота із самоорганізованих карт. Мережа базується на шарі Кохонена, здатному до сортування прикладів у відповідні кластери і використовується як для проблем класифікації, так і для кластеризації зображень.

Мережа містить вхідний шар, самоорганізовану карту Кохонена і вихідний шар. Приклад мережі зображено на рис. 5.6. Вихідний шар має стільки нейронів, скільки є відмінних категорій або класів. Карта Кохонена має ряд нейронів, згрупованих для кожного з цих класів, кількість яких залежить від складності відношення "вхід-вихід". Звичайно, кожен клас буде мати однакову кількість елементів по всьому шарі. Шар Кохонена навчається класифікації за допомогою навчальної множини.

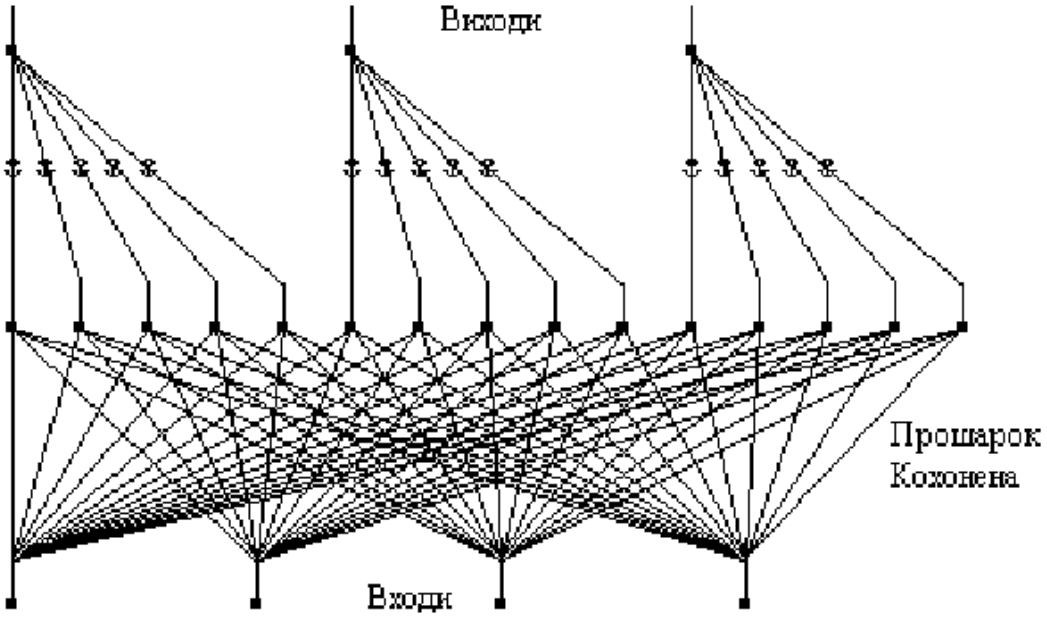

Рис. 5.6. Приклад мережі з квантуванням навчального вектора

Мережа використовує правила контрольованого навчання. Вхідний шар містить стільки нейронів, скільки є окремих вхідних параметрів. Квантування навчального вектора класифікує свої вхідні дані у визначені групування, тобто відображає *n*-мірний простір у *m*-мірний простір (бере *n* входів і створює *m* виходів).

Карти зберігають відношення між близькими сусідами в навчальній множині так, що вхідні образи, що не були попередньо вивчені, будуть розподілені по категоріях їхніх найближчих сусідів у навчальних даних.

У режимі навчання, контрольована мережа використовує шар Кохонена, де обчислюється відстань від навчального вектора до будь-якого нейрона і найближчий нейрон з'являється переможцем. Існує лише один переможець на весь шар.

Переможцеві дозволено збуджувати лише один вихідний нейрон, повідомляючи клас або кластер до якого належить вхідний вектор. Якщо нейрон-переможець знаходиться в очікуваному класі навчального вектора, його ваги підсилюються в напрямку навчального вектора. Якщо нейрон-переможець не знаходиться в класі навчального вектора, ваги з'єднань зменшуються. Ця остання операція згадується як відштовхування (repulsion). Під час навчання окремі нейрони, що приписані до часткового класу мігрують до області, зв'язаної з їхнім специфічним класом. Під час режиму функціонування, обчислюється

відстань від вхідного вектора до будь-якого нейрона і знову найближчий нейрон з'являється переможцем. Це у свою чергу генерує один вихід, визначаючи частковий клас, знайдений мережею.

Недоліки. Для складної класифікації схожих вхідних прикладів, мережа вимагає великої карти Кохонена з великою кількістю нейронів на клас. Це може бути переборено доцільним вибором навчальних прикладів або розширенням вхідного шару. Деякі нейрони мають тенденцію до перемоги занадто часто, тобто набудовують свої ваги дуже швидко, у той час як інші постійно залишаються незадіяними. Це часто трапляється, якщо їх ваги мають значення далекі від навчальних прикладів. Для усунення цього, нейрон, що перемагає занадто часто - штрафується, тобто зменшуються ваги його зв'язків з кожним вхідним нейроном. Це зменшення ваг пропорційно до різниці між частотою перемог нейрона і частотою перемог середнього нейрона.

Переваги. Алгоритм граничної корекції використовується для удосконалення рішення навіть якщо було знайдено відносно гарне рішення. Алгоритм здатний діяти, якщо нейрон-переможець знаходиться в неправильному класі, а другий найкращий нейрон у правильному класі. Навчальний вектор повинний бути близько від середньої точки простору, що з'єднує ці два нейрони. Неправильний нейрон-переможець зміщається з навчального вектора, а нейрон з іншого місця просувається до навчального вектора. Ця процедура робить чіткою границю між областями, де можлива невірна класифікація. На початку навчання бажано відключити відштовхування. Нейрон-переможець просувається до навчального вектора лише тоді, коли навчальний вектор і нейрон-переможець знаходяться в одному класі. Такий підхід доцільний, якщо нейрон повинний обійти область, яка має відмінний клас для досягнення необхідної області.