***ЛЕКЦІЯ 1***

***ОСНОВИ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ ДАНИХ***

**1. Структура сховища даних та оптимізація його обсягів**

Методи інтелектуального аналізу інформації часто розглядають як природний розвиток концепції сховищ даних. Головна відмінність сховища від бази даних полягає в тому, що їх створення і експлуатація переслідують різну мету. База даних відіграє роль помічника в оперативному управлінні організацією. Це щоденні задачі отримання актуальної інформації: бухгалтерські звітності, облік договорів, тощо. Сховище даних накопичує всі необхідні дані для здійснення задач стратегічного управління в середньостроковому і довгостроковому періоді. Наприклад, продаж товару і генерація рахунку проводяться з використанням бази даних, а аналіз динаміки продажів за декілька років, що дозволяє спланувати роботу з постачальниками - за допомогою сховища даних.

**Сховище даних (Data Warehouse) - це систематизована інформація з різнорідних джерел, яка є необхідною для обробки з метою ухвалення стратегічно важливих рішень**

Сховище будується на основі клієнт-серверної архітектури, СУБД і утиліт підтримки прийняття рішень. Дані, що надходять у сховище, стають доступні тільки для читання.

Властивості сховища даних;

o предметна орієнтація (інформацію організовано відповідно до основних аспектів діяльності);

o інтегрованість даних (дані в сховище надходять з різних джерел і відповідно агрегуються);

o стабільність, інваріантність у часі (записи в DW ніколи не змінюються, являючи собою відбитки даних, зроблені у певний час);

o мінімізація збитковості інформації (перед завантаженням у сховища дані фільтруються, зберігаються у певній послідовності, а також формується деяка підсумкова інформація).

В сховищах даних надмірність даних є мінімальною (приблизно 1%), оскільки:

o при завантаженні у сховище дані сортуються і фільтруються;

o інформація у сховищах зберігається в хронологічному порядку, що майже повністю виключає перекриття даних;

o при завантаженні у сховище дані зводяться до єдиного формату, включаючи обчислення підсумкових (агрегованих) показників.

Сервери багатовимірних баз даних можуть зберігати дані по-різному, крім агрегованих показників формується ще й додаткова інформація: поля часу, дати; адресні посилання, таблиці метаданих тощо. Це приводить до значного збільшення інформації. Вхідний масив розміром 200 Mb може розростись до об'єму 5 Gb. Сховище даних повинне бути оптимально організованою базою даних, яка забезпечує максимально швидкий і оперативний пошук інформації.

**Вітрина даних - це спрощений варіант сховища даних, що містить лише тематично орієнтовані, агреговані дані**

Глобальне сховище даних складається з трьох рівнів:

1) сховище агрегованих даних;

2) вітрини даних, які базуються на інформації зі сховища даних;

3) клієнтські робочі місця, на яких встановлено засоби оперативного аналізу даних.

У розпорядженні виробників прикладних програмних засобів є три різні технології роботи з базами даних:

o DAO (Data Access Objects) - доступ до локальних баз даних;

o RDO (Remote Data Objects) - доступ до віддалених баз даних;

o ADO (ActiveX Data Objects) - доступ до Widows-додатків через Інтернет. В основному використовується з міркувань безпеки.

Одним з перспективних напрямів удосконалення доступу до даних є гнучке конфігурування системи, коли розподіл між клієнтською і серверною частинами можливий за допомогою використання механізму віддалених

процедур.

Поряд з потоками даних існують і потоки метаданих, які розміщуються в депозитарії. Він дає змогу визначити семантичну структуру додатка у вигляді опису термінів предметної галузі, їхні взаємозв' язки й атрибути.

**Метадані - це дані про дані, які визначають джерело, приймач та алгоритм трансформації даних під час перенесення їх від джерела до приймача**

Метадані містять:

o описи структур даних та їхніх взаємозв'язків;

o інформацію про джерела даних і про ступінь їх вірогідності;

o інформацію про власників даних, права доступу;

o схему перетворення стовпців вхідних таблиць у стовпці кінцевих таблиць;

o правила підсумовування, консолідації та агрегування даних;

o інформацію про періодичність оновлення даних;

o каталог використаних таблиць, стовпців та ключів;

o фізичні атрибути стовпців;

o кількість табличних рядків та обсяг даних;

o часові ярлики (дата та час створення/модифікації записів);

o статистичні оцінки часу виконання запитів.

Контроль модифікації (versioning) полягає у властивості метаданих відслідковувати зміни в структурі даних та їх значення в часі.

Функціональна архітектура сховища даних містить наступні компоненти:

o сховище даних;

o клієнтська частина системи (дизайнери сховища, засоби розробки додатків, засоби адміністрування, інструменти аналізу даних, завантаження словника метаданих з XML-файлу у сховище і експорт його зі сховища в XML-файл;

o сервер обміну даними (Data Exchange Server) - набір програм імпорту/експорту даних зі сховища й каталогів для організації обміну даними із зовнішніми OLTP-системами;

o бібліотеки прикладних класів: ACL (Application Class Library), VCL (Visual Component Library), Win Lite.

Наповнення інформаційних сховищ відбувається в декілька етапів:

o екстракція (витяг) - імпорт даних у сховище з інформаційних підсистем, виробничих відділів та інших джерел;

o трансформація - консолідування, агрегування даних, розбиття їх на фракції, коригування та трансформування у відповідні формати;

o завантаження - у сховище, синхронізація з датою або зовнішніми подіями.

Обслуговування інформаційних сховищ полягає в: копіюванні баз даних, налаштуванні, тиражуванні, надсиланні застарілих баз даних до архіву, управлінні правами користувачів, створенні та редагуванні графічних діаграм баз даних, тощо.

Типи архівації у сховищах поділяють на:

o звичайна;

o копіювальна;

o додаткова;

o диференціальна;

o щоденна.

Архівні магнітні носії зберігають у вогнетривких сейфах або за межами обчислювального центру. Крім того, розробляється план архівації компонентів сервера баз даних. Сучасні сервери автоматично підтримують копію свого каталогу на кожному сервері вузла. Цей процес називається реплікацією каталогів (directory replication).

Звичайна архівація каталогів на всіх серверах здійснюється раз на тиждень у вихідні дні, а диференціальна - щодня в робочі .дні. У річному архіві, як правило, зберігаються дані останнього тижня місяця. Усі зміни в каталозі сервера, а також в особистих і загальних сховищах записуються у файли, які називаються журналами трансакцій (transaction log files).

Під час виконання додаткової архівації каталогу або інформаційного сховища архівуванню підлягають лише журнали трансакцій.

Для ефективної роботи зі сховищем даних, необхідно зібрати максимум інформації про процес. Наприклад, для прогнозування обсягів продажів можуть бути використані бази даних облікових систем компанії, маркетингові дані, відгуки клієнтів, дослідження конкурентів і т.п.

Необхідною для прогнозу є наступна інформація:

o хронологія продажів;

o стан складу на кожний день - якщо спад продажів буде пов'язаний із відсутністю товару на складі, а не через відсутність попиту;

o відомості про ціни конкурентів;

o зміни у законодавстві;

o загальний стан ринку;

o курс долара, інфляція;

o відомості про рекламу;

o відомості про відношення до продукції клієнтів;

o різного роду специфічну інформацію. Наприклад, для продавців морозива - температуру, а для фармакологічних складів -санітарно-епідеміологічний стан, тощо.

Проблема полягає в тому, що зазвичай в системах оперативного обліку більша частина цієї інформації відсутня, а наявна - неповна або спотворена. Кращим варіантом в цьому випадку буде створення сховища даних, куди б з певною заданою періодичністю надходила вся необхідна інформація, заздалегідь систематизована і очищена (рис.5.1).



Рис.5.1. Приклад сховища даних

Ефективна архітектура сховища даних організовується таким чином, щоб бути складовою частиною інформаційної системи управління підприємством.

Найбільш поширений випадок, коли сховище організовано за типом "зірка", де в центрі розміщуються факти і агрегатні дані, а "проміннями" є виміри. Кожна "зірка" описує певну дію, наприклад, продаж товару, його відвантаження, надходження коштів й інше:

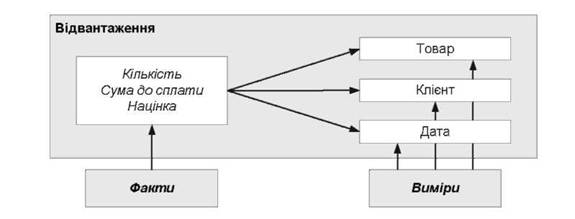


Рис.5.2. Схема організації сховища даних за типом "зірка"

Як правило, дані копіюються в сховище з оперативних баз даних і інших джерел відповідно до певних правил.

**2. Основні етапи та алгоритми інтелектуального аналізу даних**

Виокремимо два типи задач, розв’язуваних із різною ефективністю різними методами KDD (хоча, втім, реальні задачі дослідження даних можуть охоплювати обидва типи).

Задачі першого типу полягають у побудові на підставі наявних даних різних моделей, якими можна скористатися з метою прогнозування та ухвалення рішення в майбутньому, за схожої си-  
туації.

Задачі другого типу характерні тим, що наголос у них робиться на з’ясуванні сутності залежностей у множині даних, а також взаємовпливу, тобто на побудові емпіричних моделей різних систем, які легко може сприймати людина. При цьому не так уже й важливо, щоб система добре передбачала і працювала в майбутньому, а важливо зрозуміти взаємні впливи досліджуваних закономірностей (що і чим визначається в наявному масиві даних). І навіть якщо встановлені закономірності належатимуть до специфічних особливостей саме конкретних досліджуваних даних і більше ніде не траплятимуться, але нам усе одно потрібно їх з’ясувати.

Розглянемо головні етапи (кроки), характерні для будь-якого дослідження даних за допомогою методів KDD і становлять основний цикл пошуку нового знання та його оцінювання (рис. 5.1). Залежно від задачі кількість етапів, а також обсяг виконуваних на кожному з них дій можуть змінюватися, але загалом усі вони необхідні і так чи інакше мають належати процесу інтелектуального аналізу даних.

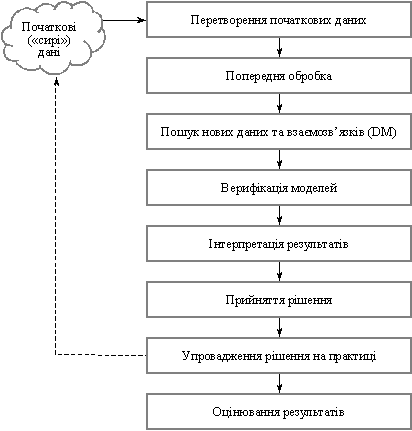


Рис. 5.1. Схема інтелектуального аналізу даних   
і оцінювання виявленого нового знання

Перший етап полягає у зведенні даних до форми, придатної для застосування конкретних реалізацій систем KDD. Нехай, скажімо, інформацію подано у вигляді текстів і потрібно побудувати автоматичний рубрикатор, класифікатор якихось анотацій, описів тощо. Вхідна інформація являє собою тексти в електронному вигляді, але практично жодна з наявних систем KDD не здатна працювати безпосередньо з текстами. Щоб працювати з певним текстом, ми маємо з вихідної текстової інформації заздалегідь дістати деякі похідні параметри (наприклад, частоту появи ключових слів, середню довжину речень, параметри, що характеризують сполучуваність тих чи інших слів у реченні тощо), тобто побудувати чіткий набір кількісних або якісних параметрів даного тексту. Ця задача найменш автоматизована в тому сенсі, що систему шуканих параметрів формує людина, хоча значення параметрів можуть обчислюватися автоматично в рамках відповідної технології первинної обробки даних. Вибравши параметри, дані можна подати у вигляді прямокутної таблиці, де кожний рядок характеризує окрему ознаку (стан, властивість) досліджуваного об’єкта, а кожний стовпець — ознаки (стани, властивості) всіх досліджуваних об’єктів. Рядки такої таблиці в теорії KDD, як і в теорії баз даних, називають записами, а стовпці — полями.

Практично всі наявні системи KDD працюють тільки зі щойно описаними прямокутними таблицями.

Здобута прямокутна таблиця — це лише «сировинний» матеріал для застосування методів KDD, і дані, що входять до неї, необхідно передусім обробити. По-перше, таблиця може містити параметри (ознаки об’єктів), що мають однакові значення в якомусь зі стовпців. Коли б досліджувані об’єкти мали тільки такі ознаки, усі вони були б абсолютно ідентичними. Звідси випливає, що відповідні ознаки жодним чином не характеризували б досліджуваних об’єктів, а отже, їх потрібно вилучити з аналізу. Можлива й така ситуація, що деяка категоріальна ознака в усіх її записах має різні значення, через що відповідне поле не придатне для аналізу даних і його також доведеться вилучити. Нарешті може статися так, що полів буде дуже багато, і якщо ми всі їх намагатимемося досліджувати, то надто відчутно збільшиться час розрахунків, оскільки практично для всіх методів KDD характерна сильна (не менш ніж квадратична, а нерідко й експоненціальна) залежність часу розрахунків від кількості параметрів, тоді як залежність часу розрахунків від кількості записів лінійна або близька до неї.

Тому у процесі попередньої обробки даних необхідно, по-перше, розглянути множину всіх ознак, що стосуються шуканої залежності, вилучити з неї ті, які явно не придатні для подальшого дослідження, та виокремити ті, що найімовірніше ввійдуть у шукану залежність. Для цього, як правило, застосовують статистичні методи, що ґрунтуються на застосуванні кореляційного аналізу, лінійних регресій, тобто методи, що дають змогу швидко, хоча й наближено оцінити вплив одного параметра на інші.

Третій етап — безпосереднє застосування методів KDD за різними сценаріями, що містять складні комбінації тих методів, які   
допомагають аналізувати дані з різних поглядів. Власне, цей етап дослідження і називають Data Mining (добування даних).

Четвертий етап — верифікація та перевірка результатів, найчастіше здійснювані в такий спосіб. Усі наявні дані, що мають бути проаналізовані, розбивають на дві (як правило, не однакові за розміром) групи. У більшій групі даних за допомогою тих чи інших методів KDD дістають моделі й залежності, а в меншій виконують їх перевірку. Далі за різницею в точності між результатами, здобутими для обох груп, доходять висновку щодо адекватності й статистичної значущості побудованої моделі. Існує багато інших, складніших способів верифікації (перехресна перевірка, бутстреп-аналіз тощо), які дають змогу оцінити значущість побудованих моделей без розбиття даних на дві групи.

Нарешті, на п’ятому етапі знання, що їх здобула людина, автоматично інтерпретуються з метою їх використання для прий-  
няття рішень та внесення сформульованих правил і залежностей до баз знань тощо. Цей етап часто передбачає застосування методів, що є проміжними між технологією KDD і технологією експертних систем. Від того, наскільки ефективним він буде, значною мірою залежить успіх розв’язання поставленої задачі.

Цим етапом і закінчується цикл KDD. Остаточне оцінювання вагомості здобутого нового знання виходить за рамки аналізу, автоматизованого чи традиційного, і стає можливим тільки після впровадження на практиці рішення, прийнятого на основі такого знання. Дослідженням практичних результатів, досягнутих за допомогою здобутого засобами KDD нового знання, завершується його оцінювання (див. рис. 5.1).

|  |
| --- |
|  |

|  |  |  |
| --- | --- | --- |
| **3. Методи інтелектуального аналізу даних**  Технології аналізу даних, що базуються на застосуванні класичних статистичних підходів, мають низку недоліків. Відповідні методи ґрунтуються на використанні усереднених показників, на підставі яких важко з’ясувати справжній стан справ у досліджуваній сфері (наприклад, середня зарплата по країні не відбиває її розміру у великих містах та в селах). Методи математичної статистики виявилися корисними насамперед для перевірки заздалегідь сформульованих гіпотез та «грубого» розвідницького аналізу, що становить основу оперативної аналітичної обробки даних (OLAP).  Наприклад, дослідження спеціалістів Гарвардського інституту показують, що на основі наявної інформації за допомогою стандартних статистичних методів не можна було передбачити великої депресії кінця 1920-х років.  Окрім того, стандартні статистичні методи відкидають (нехтують) нетипові спостереження — так звані піки та сплески. Проте окремі нетипові значення можуть становити самостійний інтерес для дослідження, характеризуючи деякі виняткові, але важливі явища. Навіть сама ідентифікація цих спостережень, не говорячи про їх подальший аналіз і докладний розгляд, може бути корисною для розуміння сутності досліджуваних об’єктів чи явищ. Як показують сучасні дослідження, саме такі події можуть стати вирішальними щодо майбутнього поводження та розвитку складних систем.  Ці недоліки статистичних методів спонукали до розвитку нових методів дослідження складних систем, що базуються на нелінійній динаміці, теорії катастроф, фрактальній геометрії тощо (див. розд. 5).  Водночас постала нагальна потреба в такій технології, яка автоматично видобувала б із даних нові нетривіальні знання у формі моделей, залежностей, законів тощо, гарантуючи при цьому їхню статистичну значущість. Новітні підходи, спрямовані на розв’язання цих проблем, дістали назву технологій інтелектуального аналізу даних.  В основу цих технологій покладено концепцію шаблонів (патернів), що відбивають певні фрагменти багатоаспектних зв’язків у множині даних, характеризуючи закономірності, притаманні підвибіркам даних, які можна компактно подати у зрозумілій людині формі. Шаблони відшукують методами, що виходять за межі апріорних припущень стосовно структури вибірки та вигляду роз- поділів значень аналізованих показників. Важлива особливість цієї технології полягає в нетривіальності відшукуваних шаблонів. Це означає, що вони мають відбивати неочевидні, несподівані регулярності у множині даних, складові так званого прихованого знання. Адже сукупність первинних («сирих») даних може містити й глибинні шари знань.  Knowledge Discovery in Databases (дослівно: «виявлення знань у базах даних» — KDD) — аналітичний процес дослідження значних обсягів інформації із залученням засобів автоматизації, що має на меті виявити приховані у множині даних структури, залежності й взаємозв’язки. При цьому передбачається повна чи часткова відсутність апріорних уявлень про характер прихованих структур та залежностей. KDD передбачає, що людина попередньо осмислює задачу й подає неповне (у термінах цільових змінних) її формулювання, перетворює дані до формату придатного для їх автоматизованого аналізу й попередньої обробки, виявляє засобами автоматичного дослідження даних приховані структури й залежності, апробовує виявлені моделі на нових даних, не використовуваних для побудови моделей, та інтерпретує виявлені моделі й результати.  Отже, KDD — це синтетична технологія, що поєднує в собі останні досягнення штучного інтелекту, чисельних математичних методів, статистики й евристичних підходів. Методи KDD особливо стрімко розвиваються протягом останніх 20 років, а раніше задачі комп’ютерного аналізу баз даних виконувалися переважно за допомогою різного роду стандартних статистичних методів.  Data Mining (дослівно: «Розробка, добування даних» — DM) — дослідження «сирих» даних і виявлення в них за допомогою «машини» (алгоритмів, засобів штучного інтелекту) прихованих нетривіальних структур і залежностей, які раніше не були відомі й мають практичну цінність та придатні для того, щоб їх інтерпретувала людина.  Розглянемо відмінності між засобами Data Mining і OLAP. Технологія OLAP спрямована на підтримання процесу прийняття управлінських рішень і використовується з метою пошуку відповіді на запитання: чому деякі речі є такими, якими вони є насправді? При цьому користувач сам формує модель-гіпотезу про дані чи відношення між даними, а далі, застосовуючи серію запитів до бази даних, підтверджує чи відхиляє висунуті гіпотези. Засоби Data Mining відрізняються від засобів OLAP тим, що замість перевірки передбачуваних користувачем взаємозалежностей вони на основі наявних даних самі можуть будувати моделі, які дають змогу кількісно та якісно оцінювати ступінь впливу різних досліджуваних факторів на задану властивість об’єкта. Крім того, засоби DM дають змогу формулювати нові гіпотези про характер досі невідомих, але таких, що реально існують, залежностей між даними.  Засоби OLAP застосовуються на ранніх стадіях процесу KDD, оскільки вони дають змогу краще зрозуміти дані, що, у свою чергу, забезпечує ефективніший результат процесу KDD.  Головна мета технології KDD — побудова моделей і відношень, прихованих у базі даних, тобто таких, які не можна знайти звичайними методами. Варто зазначити, що на комп’ютери перекладаються не лише рутинні операції (скажімо, перевірка статистичної значущості гіпотез), а й операції, що донедавна були аж ніяк не рутинними (вироблення нових гіпотез). KDD дає змогу побачити такі відношення між даними, що залишалися поза увагою дослідників.  Будуючи моделі, ми встановлюємо кількісні зв’язки між характеристиками досліджуваного явища. Щодо призначення можна виокремити моделі двох типів: прогнозні та описові (дескриптивні). Моделі першого типу використовують набори даних із відомими результатами для побудови моделей, що явно прогнозують результати для інших наборів даних, а моделі другого типу описують залежності в наявних даних. Обидва типи моделей використовуються для прийняття управлінських рішень.  Технологія KDD дає змогу не лише підтверджувати (відкидати) емпіричні висновки, а й будувати нові, невідомі раніше моделі. Знайдена модель не зможе здебільшого претендувати на абсолютне знання, але вона надає аналітикові деякі переваги вже завдяки самому факту виявлення альтернативної статистично значущої моделі, а також, можливо, стає приводом для пошуку відповіді на запитання: чи справді існує виявлений взаємозв’язок і чи є він причинним? А це, у свою чергу, стимулює поглиблені дослідження, сприяючи глибшому розумінню досліджуваного явища.  Отже, найважливіша мета застосування технології KDD до дослідження реальних систем — це поліпшення розуміння суті їх функціонування.  Відзначимо, що процес виявлення знань не є цілком автоматизованим — він вимагає участі користувача (експерта, особи що приймає рішення). Користувач має чітко усвідомлювати, що він шукає, ґрунтуючись на власних гіпотезах. Зрештою замість того, щоб підтверджувати наявну гіпотезу, процес пошуку часто сприяє появі ряду нових гіпотез. Усе це позначається терміном «discovery-driven data mining» (DDDM), і терміни Data Mining, Knowledge Discovery у загальному випадку стосуються до технології DDDM.   |  | | --- | |  |   **4. Огляд алгоритмів та ІС Data Mining**  Data Mining — це сукупність багатьох різних методів здобування знань. Вибір методу часто залежить від типу наявних даних і від того, яку інформацію потрібно дістати.  До найпоширеніших методів можна віднести такі:  об’єднання (association; іноді вживають термін affinity, що означає подібність, структурну близькість) — виокремлення структур, що повторюються в часовій послідовності. Цей метод визначає правила, за якими можна встановити, що один набір елементів корелює з іншим. Користуючись ним, аналізують ринковий кошик пакетів продуктів, розробляють каталоги, здійснюють перехресний маркетинг тощо;  аналіз часових рядів (sequence-based analysis, або sequential association) дає змогу відшукувати часові закономірності між даними (трансакціями). Наприклад, можна відповісти на запитання: купівля яких товарів передує купівлі даного виду продукції? Метод застосовується, коли йдеться про аналіз цільових ринків, керування гнучкістю цін або циклом роботи із замовником (Customer Lifecycle Management);  кластеризація (clustering) — групування записів, що мають однакові характеристики, наприклад за близькістю значень полів у БД. Використовується для сегментування ринку та замовників. Можуть залучатися статистичні методи або нейромережі. Кластеризація часто розглядається як перший необхідний крок для подальшого аналізу даних;  класифікація (classification) — віднесення запису до одного із заздалегідь визначених класів, наприклад під час оцінюваня ризиків, пов’язаних із видачею кредиту;  оцінювання (estimation);  нечітка логіка (fuzzy logic);  статистичні методи, що дають змогу знаходити криву, найближче розміщену до набору точок даних;  генетичні алгоритми (genetic algorithms);  фрактальні перетворення (fractal-based transforms);  нейронні мережі (neural networks) — дані пропускаються через шари вузлів, «навчених» розпізнавати ті чи інші структури — використовуються для аналізу переваг і цільових ринків,  а також для приваблювання замовників.  До DM можна віднести ще візуалізацію даних — побудову графічного образу даних, що допомагає у процесі загального аналізу даних вбачати аномалії, структури, тренди. Частково до DM примикають дерева рішень і паралельні бази даних.  DM тісно пов’язана (інтегрована) зі сховищами даних (Data Warehousing, DW), які, можна сказати, забезпечують роботу Data Mining.  Data Mining — міждисциплінарна технологія, що виникла й розвивається на базі досягнень прикладної статистики, розпізнавання образів, методів штучного інтелекту, теорії баз даних тощо (рис. 5.2). Звідси й численні методи та алгоритми, реалізовані в різних дійових системах Data Mining. Багато з таких систем інтегрують у собі відра- зу кілька підходів. Проте, як правило, у кожній системі присутній певний ключовий компонент, на який робиться головна ставка.  http://buklib.net/msohtml1/678/clip_image002.gif  Рис. 5.2. Data Mining — міждисциплінарна галузь  Предметно-орієнтовані аналітичні системи. Такі системи дуже різноманітні. Найширший їх підклас, що набув поширення у сфері дослідження фінансових ринків, дістав назву «технічний аналіз». Він містить кілька десятків методів прогнозування динаміки цін і вибору оптимальної структури інвестиційного портфеля, які ґрунтуються на різних емпіричних моделях динаміки ринку. Зазначені методи, застосовуючи здебільшого нескладний статистичний апарат, максимально враховують специфіку своєї предметної галузі (професійна мова, системи різних індексів тощо). На ринку пропонується багато відповідних програм.  Статистичні пакети. Новітні версії майже всіх відомих статистичних пакетів поряд із традиційними статистичними методами містять також елементи Data Mining. Проте основна увага приділяється в них класичним підходам — кореляційному, регресійному, факторному аналізу та іншим. Недоліком відповідних систем можна вважати вимоги щодо спеціальної підготовки користувача. Існує, однак, і принциповий недолік статистичних пакетів, що обмежує їх застосування в Data Mining: більшість методів, що входять до складу пакетів, спираються на усереднені характеристики вибірки, які в разі дослідження складних життєвих явищ часто є фіктивними. І все ж деякі сучасні пакети пропонують модулі для інтелектуального аналізу. Наприклад, STATISTICA містить модуль Data Miner, що дає змогу будувати дерева рішень, нейронні мережі, виявляти IF THEN правила тощо.  До найпотужніших і найчастіше застосовуваних статистичних пакетів належать SAS (компанія SAS Institute), SPSS (SPSS), STATGRAPICS (Manugistics), STATISTICA, STADIA, Eviews тощо.  Нейронні мережі. Це великий клас систем, архітектура яких певною мірою аналогічна побудові нервової тканини з нейронів. В одній із найпоширеніших архітектур — багатошаровому перцептроні зі зворотним зв’язком помилки, імітується робота нейронів у складі ієрархічної мережі, де кожний нейрон вищого рівня з’єднаний своїми входами з виходами нейронів нижчого шару. На нейрони найнижчого шару подаються значення вхідних параметрів, на підставі яких потрібно приймати якісь рішення, прогнозувати розвиток ситуації тощо. Ці значення розглядаються як сигнали, що передаються в наступний шар, послаблюючи чи підсилюючи його залежно від числових значень (ваг), приписуваних міжнейронним зв’язкам.  У результаті на виході нейрона найвищого шару виробляється деяке значення, що розглядається як відповідь (реакція) всієї мережі на значення вхідних параметрів. Для того щоб мережу можна було використовувати надалі, її потрібно «навчити» на базі здобутих раніше даних, для яких відомі значення вхідних параметрів і правильні відповіді на них. Тренування полягає в доборі ваг міжнейронних зв’язків, що забезпечують найбільшу близькість відповідей мережі до відомих правильних відповідей.  Основним недоліком нейромережної технології є те, що вона потребує дуже великого обсягу навчальної вибірки. Ще один істотний недолік такий: навіть натренована нейронна мережа — це «чорна скринька». Знання, зафіксовані як ваги кількох сотень міжнейронних зв’язків, людина не в змозі проаналізувати й інтерпретувати.  До нейромережних систем належить, скажімо, BrainMaker (CSS), NeuroShell (Ward Systems Group), OWL (HyperLogic).  Системи міркувань на основі аналогічних випадків. Для того щоб зробити деякий прогноз або вибрати правильне рішення, зазначені системи (case based reasoning — CBR) відшукують у минулому близькі аналоги наявної ситуації, вибираючи ті самі відповіді, що були для них правильними. Тому цей метод ще називають методом «найближчого сусіда» (nearest neighbour). Останнім часом набув поширення також термін «memory based reasoning», який акцентує увагу на тому, що рішення приймається на підставі всієї інформації, нагромадженої в пам’яті.  Системи CBR забезпечують добрі результати в найрізноманітніших задачах. Головний їхній недолік полягає в тому, що вони взагалі не створюють будь-яких моделей чи правил, які узагальнюють попередній досвід, а ґрунтуються у виборі рішення на всьому масиві доступних історичних даних. Саме через це не можна встановити, на яких конкретно засадах системи CBR будують свої відповіді.  Інший недолік — певне «свавілля», що його припускаються такі системи, вибираючи міру «близькості», від якої залежить обсяг множини прецедентів, збережуваних у пам’яті з метою досягнення задовільної класифікації або прогнозу.  З-поміж систем CBR назвемо, наприклад, KATE tools (Acknosoft, Франція), Pattern Recognition Workbench (Unica, США).  Дерева рішень (decision trees). Дерева рішень є одним із найпопулярніших підходів до розв’язання задач Data Mining. Вони створюють ієрархічну структуру правил, класифікованих за схемою «ЯКЩО... ТО...» (if-then), яка має вигляд дерева. Для ухвалення рішення про те, до якого класу варто віднести деякий об’єкт (ситуацію, потрібно відповісти на запитання, що містяться у вузлах  цього дерева, починаючи з його кореня. Запитання можуть бути, наприклад, такі: «Значення параметра а більше за x?». Якщо відповідь ствердна, відбувається перехід до правого вузла наступного рівня, якщо заперечна — до лівого вузла. Далі знову ставиться запитання, пов’язане з відповідним вузлом.  Популярність цього підходу зумовлюється наочністю та зро- зумілістю. Але дерева рішень принципово не здатні знаходити «кращі» (найбільш повні і точні) правила в даних. Вони реалізують принцип послідовного перегляду ознак і збирають фактично уламки наявних закономірностей, створюючи лише ілюзію логічного висновку.  Проте більшість систем діють саме за цим методом. До таких належать, наприклад, See5/З5.0 (RuleQuest, Австралія), Clementine (Integral Solutions, Великобританія), SIPINA (University of Lyon, Франція), IDIS (Information Discovery, США), KnowledgeSeeker (ANGOSS, Канада).  Еволюційне програмування. Сучасний його стан схарактеризуємо, розглянувши систему PolyAnalyst, в якій гіпотези про вигляд залежності цільової змінної від інших змінних формулюються у вигляді програм, що подаються деякою внутрішньою мовою програмування. Процес побудови програм розгортається еволюційно в комплексі програм (на кшталт генетичних алгоритмів). Коли система відшукує програму, що більш-менш задовільно виражає шукану залежність, вона починає вносити до неї невеликі модифікації і добирає серед побудованих дочірніх програм ті, які підвищують точність. У такий спосіб система «вирощує» кілька генетичних ліній програм, що конкурують між собою стосовно точності вираження шуканої залежності. Спеціальний модуль системи PolyAnalyst перекладає знайдені залежності з внутрішньої мови системи зрозумілою користувачеві мовою (математичні формули, таблиці тощо).  Інший напрямок еволюційного програмування пов’язаний із пошуком залежності цільових змінних від решти у формі функцій певного вигляду. Наприклад, один із найбільш вдалих алгоритмів цього типу — метод групового врахування аргументів (МГВА) передбачає відшукання залежності у формі поліномів.  Генетичні алгоритми. Data Mining не є головною сферою застосування генетичних алгоритмів. Їх варто розглядати радше як могутній засіб розв’язання різноманітних комбінаторних задач та задач оптимізації. Проте генетичні алгоритми становлять нині стандартний інструментарій методів Data Mining.  Перший крок під час побудови генетичних алгоритмів — це кодування вихідних логічних закономірностей у базі даних, що їх іменують хромосомами, а весь набір таких закономірностей називають популяцією хромосом. Далі для реалізації концепції вибору вводиться спосіб зіставлення різних хромосом. Популяція обробляється за допомогою процедур репродукції, мінливості (мутацій), генетичної композиції. Ці процедури імітують біологічні процеси. Найважливіші з них такі:  випадкові мутації даних в індивідуальних хромосомах, переходи і рекомбінації генетичного матеріалу, що міститься в індивідуальних батьківських хромосомах, і міграцію генів;  у процесі виконання процедур на кожній стадії еволюції виходять популяції з дедалі досконалішими індивідами.  Генетичні алгоритми зручні тим, що їх легко розпаралелити. Наприклад, можна розбити покоління на кілька груп і працювати з кожною з них незалежно, змінюючи час від часу кілька хромосом. Існують також інші методи розпаралелювання генетичних алгоритмів.  Генетичні алгоритми мають і низку недоліків. Критерій добору хромосом і використовуваних процедур є евристичним і зовсім не гарантує відшукання «найкращого» рішення. Як і в реальному житті, еволюцію може «заклинити» на якій-небудь непродуктивній галузці. І, навпаки, може статися, що безперспективні батьки, яких вилучить з еволюції генетичний алгоритм, будуть здатні породити високоефективного нащадка. Це особливо стає помітним під час розв’язування багатовимірних задач зі складними внутрішніми зв’язками.  Як приклад можна згадати систему GeneHunter фірми Ward Systems Group.  Алгоритми обмеженого перебору. Ці алгоритми обчислюють частоти комбінацій простих логічних подій у підгрупах даних. Приклади простих логічних подій: x = a; x < a; x > a; a < x < b  тощо, де x — деякий параметр, a та b — константи. На підставі аналізу обчислених частот робиться висновок про корисність тієї чи іншої комбінації стосовно встановлення асоціацій у даних, класифікації, прогнозування і т. ін.  Найбільш виразним сучасним представником цього підходу є система WizWhy підприємства WizSoft.  Системи для візуалізації багатовимірних даних. Засоби графічного відображення даних тією чи іншою мірою підтримуються всіма системами Data Mining. Проте дуже значну частку ринку становлять системи, що спеціалізуються винятково на цій функції. Одна з них — програма DataMiner 3D словацької фірми Dimension 5.  У таких системах основна увага сконцентрована на дружньому користувальницькому інтерфейсі, що дає змогу асоціювати з аналізованими показниками різні параметри діаграми розсіювання об’єктів (записів) бази даних. До зазначених параметрів належать колір, форма, орієнтація щодо власної осі, розміри й інші властивості графічних елементів зображення. Крім того, системи візуалізації даних є зручними засобами для масштабування зображень. |  |

Тема 1. Перспективні дослідження і розробки по інтелектуальних системах

Призначення аналітичних технологій

Традиційні технології

Детерміновані технології

Імовірнісні технології

Приклади реальних задач

Оптимальний розподіл інвестицій

Прогнозування курсу акцій

Недоліки традиційних технологій

Нові технології

Комп'ютерні системи стають банальністю. Дійсно, вони майже повсюдні. Вони є найважливим компонентом у функціонуванні бізнесу, урядового, військового, навколишнього середовища, установах охорони здоров'я і є частиною багатьох освітніх програм навчання. Ці комп'ютерні системи, все більш і більш впливаючи на наше життя мають бути спроможними швидко адаптуватися, змінюватись та допомагати нам і нашим установам справлятися з непередбаченими можливостями світу.

Національна конкурентноздатність залежить від можливостей доступу, обробки та аналізу інформації. Аналіз і передача даних за допомогою комп'ютера надали нам велику кількість інформації. Однак, щоб досягти повного співробітництва, комп'ютерні системи повинні вміти більше, ніж обробляти інформацію, але і мати інтелект. Вони повинні кваліфіковано зберігати й використовувати великі обсяги інформації й ефективно допомагати людям знаходити нові шляхи рішення проблем, використовуючи більш природні засоби комунікації.

Щоб перебороти обмеження існуючих систем, потрібно зрозуміти шляхи і способи взаємодії людей між собою і зі світом, розробити методи для з'єднання людського інтелекту і комп'ютерних систем. Для вирішення цього призначені аналітичні технології.

Аналітичні технології - це методики, які на основі певних моделей, алгоритмів, математичних теорем дозволяють по відомих даних оцінити значення невідомих характеристик і параметрів. Найпростіший приклад аналітичної технології - теорема Піфагора, що дозволяє по довжинах сторін прямокутника визначити довжину його діагоналі. Ця технологія заснована на відомій формулі с2=а2+b2.

Іншим прикладом аналітичної технології є способи, за допомогою яких людський мозок обробляє інформацію. Навіть мозок дитини може вирішувати задачі, непідвласні сучасним комп'ютерам, такі як розпізнавання знайомих облич у юрбі, чи ефективне керування декількома десятками м'язів при грі у футбол. Унікальність мозку полягає в тому, що він здатен навчатися рішенню нових задач - грі в шахи, водінню автомобіля і т.д. Проте, мозок погано пристосований до обробки великих обсягів числової інформації - людина не зможе знайти навіть квадратний корінь з числа 463761, не використовуючи калькулятора або алгоритму обчислення в стовпчик. На практиці ж часто зустрічаються задачі про числа, набагато більш складні, ніж витяг кореня. Таким чином, людині для рішення таких задач необхідні додаткові методики й інструменти.

Призначення аналітичних технологій

Аналітичні технології потрібні в першу чергу людям, що приймають важливі рішення - керівникам, аналітикам, експертам, консультантам. Дохід компанії у великому ступені визначається якістю цих рішень - точністю прогнозів, оптимальністю обраних стратегій.

Прогнозування:

курсів валют;

цін на сировину;

попиту;

доходу компанії;

рівня безробіття;

числа страхових випадків.

Оптимізація:

розкладів;

маршрутів;

плану закупівель;

плану інвестицій;

стратегії розвитку.

Як правило, для реальних задач бізнесу й виробництва не існує чітких алгоритмів рішення. Зазвичай керівники й експерти вирішували такі задачі тільки на основі особистого досвіду. Часто класичні методики виявляються малоефективними в багатьох практичних задачах. Це пояснюється тим, що неможливо досить точно описати реальність за допомогою невеликого числа параметрів моделі, або розрахунок моделі займає занадто багато часу та обчислювальних ресурсів, а за допомогою аналітичних технологій будуються системи, що дозволяють істотно підвищити ефективність рішень.

Традиційні технології

Детерміновані технології

Аналітичні технології типу теореми Піфагора використовуються людиною вже багато століть. За цей час була створена величезна кількість формул, теорем і алгоритмів для рішення класичних задач - визначення об'ємів, рішення систем лінійних рівнянь, пошуку коренів багаточленів. Розроблено складні й ефективні методи для рішення задач оптимального керування, рішення диференційних рівнянь і т.д. Всі ці методи діють по однієї і тій же схемі.



Для застосування алгоритму необхідно, щоб дана задача цілком описувалася визначеною детермінованою моделлю (деяким набором відомих функцій і параметрів). У такому випадку алгоритм дає точну відповідь. Наприклад, для застосування теореми Піфагора потрібно перевірити, що трикутник - прямокутний.



Імовірнісні технології

На практиці часто зустрічаються задачі, пов'язані зі спостереженням випадкових величин - наприклад, задача прогнозування курсу акцій. Для подібних задач не можна побудувати детерміновані моделі, тому застосовується принципово інший, імовірнісний підхід.

Параметри імовірнісних моделей - це розподіли випадкових величин, їхні середні значення, дисперсії і т.д. Як правило, ці параметри заздалегідь невідомі, а для їхньої оцінки використовуються статистичні методи, що застосовуються до вибірок спостережених значень (історичних даних).



Такого роду методи припускають, що відомо деяка імовірнісна модель задачі. Наприклад, у задачі прогнозування курсу можна припустити, що завтрашній курс акцій залежить тільки від курсу за останні 2 дні (авторегресійна модель). Якщо це вірно, то спостереження курсу протягом декількох місяців дозволяють досить точно оцінити коефіцієнти цієї залежності і прогнозувати курс у майбутньому.



Приклади реальних задач

Оптимальний розподіл інвестицій

Задача. Є інвестиційний капітал, який потрібно розподілити серед 10 проектів. Для кожного проекту задана функція залежності прибутку від обсягу вкладення. Потрібно знайти найбільш прибутковий варіант розподілу капіталу, за умови, що задано мінімальний і максимальний обсяг інвестицій для кожного проекту.

Традиційне рішення. Переважно рішення в даному випадку приймає керівник, ґрунтуючись тільки на особистих враженнях про проекти. Розміри упущеної вигоди при цьому не підраховують, і неоптимальність рішення може залишитися непоміченої.

У випадку, якщо керівник доручає аналітикам вибрати найбільш прибутковий варіант, застосовуються математичні методи оптимізації. Якщо всі дані функції лінійні, то можна застосувати методи лінійного програмування (симплекс-метод). Якщо хоча б одна з функцій нелінійна, то можна використовувати метод градієнтного спуску чи повного перебору.

Прогнозування курсу акцій

Задача. Трейдеру фондового ринку потрібно щоденний прогноз поведінки курсу акцій енергетичного підприємства.

Дано. Значення котувань і різних ринкових індикаторів за останній рік. Також відомі котування нафтових і вугільних компаній, а також міських компаній-енергоспоживачів, що тісно зв'язані з курсом акцій ЄЕС.

Традиційне рішення. Використовуються методики технічного аналізу, кореляційного аналізу, статистика.

Недоліки традиційних технологій

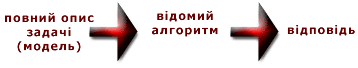
На жаль, класичні методики виявляються малоефективними в багатьох практичних задачах. Це зв'язано з тим, що неможливо досить повно описати реальність за допомогою невеликого числа параметрів моделі, або розрахунок моделі вимагає занадто багато часу й обчислювальних ресурсів. Зокрема, розглянемо проблеми, що виникають при рішенні задачі оптимального розподілу інвестицій.

У реальній задачі жодна з функцій не відома точно - відомі лише приблизні або очікувані значення прибутку. Для того щоб позбутися від невизначеності, ми змушені зафіксувати функції, втрачаючи при цьому точність опису задачі.

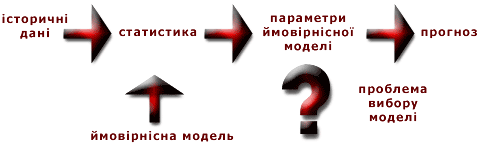
Детермінований алгоритм для пошуку оптимального рішення (симплекс-метод) застосовуємо тільки в тому випадку, якщо всі дані функції лінійні. У реальних задачах бізнесу ця умова не виконується. Хоча дані функції можна апроксимувати лінійними, рішення в цьому випадку буде далеким від оптимального.

Якщо одна з функцій нелінійна, то симплекс-метод не застосовується, і залишається два традиційних шляхи рішення цієї задачі.

Перший шлях - використовувати метод градієнтного спуску для пошуку максимуму прибутку. У даному випадку область визначення функції прибутку має складну форму, а сама функція - кілька локальних максимумів, тому градієнтний метод може привести до неоптимального рішення. Другий шлях - провести повний перебір варіантів інвестування. Якщо кожна з 10 функцій задана в 100 точках, то прийдеться перевірити близько 1020 варіантів, що займе не менш декількох місяців роботи сучасного комп'ютера.



Ймовірнісні технології мають істотні недоліки при рішенні практичних задач. Ми проілюстрували роботу імовірнісного підходу на прикладі простої лінійної авторегресійної моделі, однак залежності, що зустрічаються на практиці, переважно нелінійні. Навіть якщо й існує проста залежність, то її вигляд заздалегідь невідомий. Відзначимо також, що статистичні методи добре розроблені тільки для одновимірних випадкових величин. Якщо ж ми хочемо враховувати для прогнозування курсу акцій кілька взаємозалежних факторів (наприклад, обсяг угод, курс долара і т.д.), то прийдеться звернутися до побудови багатовимірної статистичної моделі. Однак, такі моделі або припускають гаусівський розподіл спостережень (що не виконується на практиці), або не обґрунтовані теоретично. У багатовимірній статистиці через брак кращого нерідко застосовують малообгрунтовані евристичні методи, що по своїй суті дуже близькі до технології нейронних мереж. Про це буде більш докладно розказано нижче.



Нові технології

В останні 10 років іде активний розвиток аналітичних систем нового типу. В їх основі - технології штучного інтелекту, що імітують природні процеси, наприклад, такі як діяльність нейронів мозку процес або природного відбору.

Інтелектуальні аналітичні системи містять у собі:

здатність міркування відносно задачі і знання, засновані на здоровому глузді;

міркування відносно спільного процесу і знання можливостей інших систем і людей, що приймають участь у взаємодії;

зв'язок з користувачами за допомогою розуміння природної мови, малюнків, зображень, і знаків;

системи повинні відчувати середовище;

координувати прийняття, планування, і дії;

навчання на попередньому досвіді й адаптацію до поведінки.

Розуміння цих можливостей у людях і втілення їх при розробці програм є центральним у створеннях новітніх аналітичних технологій, що здатні здобувати знання та керувати ними.

Національна конкурентноздатність залежить від зростання потужностей для проведення інформаційного аналізу, прийняття рішення, гнучкого проектування та виробництва. Зусилля в цих областях були обмежені недостатніми даними, відсутністю обчислювальної потужності або неадекватними контролюючими механізмами. Багато з цих обмежень можуть бути усунені тільки при додаванні інтелекту до систем.

**Тема 2. Технології інтелектуальних обчислень - стан проблеми, нові рішення**

* [Визначення Data Mining](http://www.victoria.lviv.ua/html/oio/html/theme2.htm#2_1_1)
  + [Можливості інтелектуального аналізу](http://www.victoria.lviv.ua/html/oio/html/theme2.htm#2_1_2)
  + [Недоліки технології інтелектуального аналізу даних](http://www.victoria.lviv.ua/html/oio/html/theme2.htm#2_1_3)
  + [Data Mining і OLAP](http://www.victoria.lviv.ua/html/oio/html/theme2.htm#2_1_4)
  + [Data Mining і сховища даних](http://www.victoria.lviv.ua/html/oio/html/theme2.htm#2_1_5)
  + [Технології інтелектуальних обчислень і апаратне забезпечення](http://www.victoria.lviv.ua/html/oio/html/theme2.htm#2_1_6)
* [Сфера застосування технологій інтелектуальних обчислень](http://www.victoria.lviv.ua/html/oio/html/theme2.htm#2_2)
  + [Бізнес-застосування Data Mining](http://www.victoria.lviv.ua/html/oio/html/theme2.htm#2_2_1)
* [Технології інтелектуальних обчислень та український ринок](http://www.victoria.lviv.ua/html/oio/html/theme2.htm#2_3)

**Представлення нової технології інтелектуального аналізу даних**

Комп'ютерні технології з організацією інтелектуальних обчислень переживають свій розквіт. Це пов'язано, головним чином, з потоком нових ідей, що виходять з галузі комп'ютерних наук, яка утворилась на перетині штучного інтелекту, статистики та теорії баз даних. Зараз відбувається стрімкий зріст числа програмних продуктів, що використовують нові технології, а також типів задач, де їх застосування надає значного економічного ефекту. Елементи автоматичної обробки і аналізу даних, що називають Data Mining (знаходження знань) стають невід'ємною частиною концепції електронних сховищ даних та організації інтелектуальних обчислень. Простий доступ користувача до сховища даних забезпечує тільки отримання відповідей на питання, що були задані, в той час як технологія data mining дозволяє побачити ("знайти") приховані правила і закономірності у наборах даних, які користувач не може передбачити, і застосування яких може сприяти збільшенню прибутків підприємства.

**Визначення Data Mining**

Data Mining переводиться як "видобуток" чи "розкопка даних". Нерідко поруч з Data Mining зустрічаються слова "інтелектуальний аналіз даних". Справа в тому, що людський розум сам по собі не пристосований для сприйняття великих масивів різнорідної інформації. Людина до того ж не здатна уловлювати більш двох-трьох взаємозв'язків навіть у невеликих вибірках. Але і традиційна математична статистика, яка довгий час претендувала на роль основного інструмента аналізу даних, також нерідко пасує при рішенні задач з реального складного життя. Вона оперує усередненими характеристиками вибірки, що часто є фіктивними величинами (типу середньої температури пацієнтів по лікарні, середньої висоти будинку на вулиці, що складається з палаців і халуп і т.п.). Тому методи математичної статистики виявляються корисними, головним чином, для перевірки заздалегідь сформульованих гіпотез.

**Можливості інтелектуального аналізу**

Більшість організацій накопичують під час своєї діяльності величезні обсяги даних, але єдине, що вони хочуть від них одержати - це корисна інформація. Як можна довідатися з даних про те, що потрібно найбільш вигідним для організації клієнтам, як розмістити ресурси найбільш ефективним чином або як мінімізувати втрати? Для вирішення цих проблем призначені новітні технології інтелектуального аналізу. Вони використовують складний статистичний аналіз і моделювання для знаходження моделей і відношень, прихованих у базі даних - таких моделей, що не можуть бути знайдені звичайними методами.

Модель, як і карта - це абстрактне представлення реальності. Карта може вказувати на шлях від аеропорту до будинку, але вона не може показати аварію, що створила пробку, або ремонтні роботи, які ведуться в даний момент і вимагають об'їзду. Доти поки модель не відповідає існуючим реально відношенням, неможливо отримати успішні результати.  
Існують два види моделей: передбачувані й описові. Перші використовують один набір даних з відомими результатами для побудови моделей, що явно передбачають результати для інших наборів, а другі описують залежності в існуючих даних, що у свою чергу використовуються для прийняття керівних рішень чи дій.

Звичайно ж, компанія, що довго знаходиться на ринку і знає своїх клієнтів вже інформована про багато моделей, які спостерігалися протягом декількох останніх періодів. Технології інтелектуального аналізу можуть не тільки підтвердити ці емпіричні спостереження, але і знайти нові, невідомі раніше моделі. Спочатку це може дати користувачеві лише невелику перевагу. Але така перевага, якщо її об'єднати по кожному товару і кожному клієнту, дає істотний відрив від тих, хто не використовує технології Data Mining. З іншого боку, за допомогою методів data mining можна знайти таку модель, що приведе до радикального поліпшення у фінансовому і ринковому становищі компанії.

**Недоліки технології інтелектуального аналізу даних**

Data Mining - це набір засобів, а не чарівна паличка. Він не знаходиться в базі даних і не посилає електронну пошту, коли бачить цікаву модель. Він не виключає необхідності знання Вашого бізнесу і розуміння самих даних чи аналітичних методів. Цей набір засобів допомагає аналітикам у знаходженні моделей і відношень у даних, але він не говорить про цінність цих моделей для організації. Кожна модель повинна перевірятися в реальному середовищі.

Хоча інструментарій інтелектуального аналізу і звільнює користувача від можливих складностей у застосуванні статистичних методів, він все-таки потребує від нього розуміння роботи цього інструментарію й алгоритмів, на яких він базується. Крім цього, технологія знаходження нового знання в базі даних не може дати відповіді на ті питання, що не були задані. Вона не заміняє аналітиків чи менеджерів, а дає їм сучасний, могутній інструмент для поліпшення роботи, яку вони виконують.

**Data Mining і OLAP**

У професіоналів обробки даних часто виникає питання про різницю між засобами інтелектуального аналізу і засобами OLAP (On-Line Analytical Processing) - засобами оперативної аналітичної обробки.

OLAP - це частина технологій, скерованих на підтримку ухвалення рішення. Звичайні засоби формування запитів і звітів описують саму базу даних. Технологія OLAP використовується для відповіді на задані питання. При цьому користувач сам формує гіпотезу про дані чи відношення між даними і після цього використовує серію запитів до бази даних для підтвердження чи відхилення цих гіпотез. Засоби Data Mining відрізняються від засобів OLAP тим, що замість перевірки передбачуваних взаємозалежностей, вони на основі наявних даних можуть робити моделі, що дозволяють кількісно оцінити ступінь впливу досліджуваних факторів. Крім того, засоби інтелектуального аналізу дозволяють робити нові гіпотези про характер невідомих, але реально існуючих відношень у даних.

Сучасні технології інтелектуального аналізу перелопачують інформацію з метою автоматичного пошуку шаблонів (патернів), характерних для яких-небудь фрагментів неоднорідних багатомірних даних. На відміну від оперативної аналітичної обробки даних у Data Mining тягар формулювання гіпотез і виявлення незвичайних шаблонів перекладено з людини на комп'ютер.

**Приклади формулювань задач при використанні методів OLAP і Data Mining**

|  |  |
| --- | --- |
| OLAP | Data Mining |
| Які середні показники травматизму для людей, що палять і не палять? | Які фактори найкраще передбачають нещасливі випадки? |
| Які середні розміри телефонних рахунків існуючих клієнтів у порівнянні з рахунками колишніх клієнтів (що відмовилися від послуг телефонної компанії)? | Які характеристики відрізняють клієнтів, що, цілком ймовірно, збираються відмовитися від послуг телефонної компанії? |
| Яка середня величина щоденних покупок по вкрадених і не вкрадених кредитних картках? | Які схеми покупок характерні для шахрайства з кредитними картками? |

У принципі немає нічого нового в постановці задачі Data Mining. Фахівці протягом декількох останніх десятків років вирішували подібні задачі ("пошук емпіричних закономірностей", "евристичний пошук у складних середовищах", "індуктивний висновок" і т.п.). Але тільки зараз суспільство в цілому дозріло для розуміння практичної важливості і широти цих задач. По-перше, у зв'язку з розвитком технологій запису і збереження даних сьогодні на людей вилились колосальні потоки інформаційної руди у всіляких областях, що без продуктивної переробки грозять перетворитися в нікому не потрібні смітники. І, по-друге, засоби і методи обробки даних стали доступними і зручними, а їхні результати зрозумілими для будь-якої людини.

**Data Mining і сховища даних**

Для успішного проведення всього процесу знаходження нового знання необхідною умовою є наявність сховища даних. Принципи побудови сховищ - це дуже велика тема, що заслуговує окремого курсу лекцій. Обмежимося лише основними принципами побудови сховища даних.

Отже, **сховище даних** - це предметно-орієнтований, інтегрований, прив'язаний до часу, незмінний збір даних для підтримки процесу прийняття керівних рішень. Предметна орієнтація означає, що дані об'єднані в категорії і зберігаються відповідно до тих областей, що вони описують, а не до застосувань, що їх використовують. Інтегрованість означає, що дані задовольняють вимогам усього підприємства (у його розвитку), а не єдиної функції бізнесу. Тим самим сховище даних гарантує, що однакові звіти, згенеровані для різних аналітиків, будуть містити однакові результати. Прив'язка до часу означає, що сховище можна розглядати як сукупність "історичних" даних: можна відновити картину на будь-який момент часу. Атрибут часу завжди є явно присутнім у структурах сховища даних. Незмінність означає, що, потрапивши один раз у сховище, дані вже не змінюються на відміну від оперативних систем, де дані зобов'язані бути присутніми тільки в останній версії, оскільки постійно змінюються. У сховище дані лише долучаються.

Для рішення переліченого ряду задач, що неминуче виникають при організації й експлуатації інформаційного сховища, повинно існувати спеціалізоване програмне забезпечення. Сучасні засоби адміністрування сховища даних мають забезпечити ефективну взаємодію з інструментарієм знаходження нового знання.

**Технології інтелектуальних обчислень і апаратне забезпечення**

Ключовою можливістю застосування новітніх технологій стало величезне падіння ціни за останні кілька років на пристрої збереження інформації з десятків доларів за збереження мегабайта інформації, до десятків центів. Це істотно здешевіло і збільшило можливості збору і збереження великих обсягів інформації.

Падіння цін на процесори з одночасним збільшенням їхньої швидкодії сприяло розвитку технологій зв'язаних з обробкою величезних масивів інформації. У результаті цього була переборена множина бар'єрів, що знаходяться на шляху знаходження нового знання в сховищах інформації.

Клієнт-серверна архітектура також є необхідним атрибутом технології інтелектуального аналізу даних. Такий підхід надає можливості виконувати найбільш трудомісткі процедури обробки даних на високопродуктивному сервері як розробникам проектів, так і користувачам. На цьому ж сервері можуть зберігатися і по запитах клієнтів виконуватися корпоративні проекти.

**Сфера застосування технологій інтелектуальних обчислень**

Сфера застосування технологій інтелектуальних обчислень нічим не обмежена - вона скрізь, де є які-небудь дані. Але в першу чергу методи Data Mining сьогодні, м'яко говорячи, заінтригували комерційні підприємства, що розгортають проекти на основі інформаційних сховищ даних. Досвід багатьох таких підприємств показує, що віддача від використання технологій інтелектуального аналізу даних може досягати 1000%. Наведено в InterNet приклад - річна економія 700 тис. дол. за рахунок упровадження Data Mining у мережі універсамів у Великобританії. Технології Data Mining надають велику допомогу для керівників і аналітиків у їхній повсякденній діяльності. Ділові люди усвідомили, що за допомогою методів Data Mining вони можуть одержати відчутні переваги в конкурентній боротьбі. Коротко охарактеризуємо деякі можливі бізнес-застосування технологій інтелектуального аналізу даних та обчислень.

**Бізнес-застосування Data Mining**

Застосування інтелектуальних технологій поширено в широкому спектрі індустрій. Методи Data Mining поширені в багатьох організаціях через те, що вони можуть зробити істотний внесок у збільшення доходів. Ці методи можуть використовуватися для керування взаємовідносинами з клієнтами. Визначаючи характеристики клієнтів, що можуть звернутись до конкурентів, компанія може починати дії для їхнього утримання, тому що зберегти клієнта завжди дешевше, ніж придбати нового.

Маркетинг даних - область, в якій активно застосовується інтелектуальні технології. Визначаючи за допомогою методів Data Mining коло кандидатів для розсилання цільової реклами можна збільшити продаж, при цьому зменшивши витрати на проведення такої реклами.

Телекомунікаційні компанії і компанії, що випускають кредитні картки, є лідерами в застосуванні цих технологій для визначення можливих втрат клієнтів.

Страхові компанії і фондові біржі застосовують ці технології для визначення втрат клієнтів.

Ще одна область застосування методів інтелектуального аналізу даних - медицина; тут передбачається ефективність застосування медикаментів, хірургічних процедур і медичних тестів.

Компанії, що діють на фінансовому ринку, визначають ринкові і галузеві характеристики для передбачення індивідуальних і фондових переваг у найближчому майбутньому.

Супермаркети визначають, які продукти продавати і як їх розташувати всередині магазина для досягнення найбільшої кількості продажів.

Фармацевтичні фірми використовують сховища даних по хімічних сполуках для знаходження комбінацій цих з'єднань, що надалі можна буде використовувати як ліки для лікування різних захворювань.

Ключем до успішного застосування методів інтелектуальних обчислень служить не просто вибір алгоритму, а майстерність людини, що проводить побудову моделі, і можливості програми проводити процес моделювання. Інформативність реалізованого проекту залежить від цих факторів у більшому ступені, ніж від алгоритмів.

Існують дві сторони успіху в пошуку даних. По-перше - це чітке і ясне формулювання задачі, що підлягає рішенню. По-друге - це використання правильних даних. Після вибору даних із усіх доступних джерел (чи навіть придбання даних із зовнішніх джерел) необхідно їх перетворити або згрупувати у визначеному порядку.

Чим більше аналітик може "грати" з даними, будувати моделі, оцінювати результати (тобто більше працювати з даними за одиницю часу), тим краще може бути результат. Робота з даними стає більш ефективної, коли можлива інтеграція наступних компонентів: візуалізація, графічний інструментарій, засоби формування запитів, оперативна аналітична обробка, що дозволяють зрозуміти дані й інтерпретувати результати, і, нарешті, самі алгоритми, що будують моделі.

**Технології інтелектуальних обчислень та український ринок**

На українському ринку технології інтелектуальних обчислень роблять лише перші кроки. Це можна пояснити їх високою вартістю, але, як показує історія розвитку інших галузей комп'ютерного ринку України, сам по собі цей фактор навряд чи є визначальним. Скоріше тут виявляється дія деяких специфічних для України негативних факторів, що різко зменшують ефективність застосування аналітичних технологій. Постараємося визначити ці фактори, проаналізувати ступінь притаманних їм різних класів систем інтелектуального аналізу даних та обчислень, а також виділити властивості таких систем, що полегшують українським покупцям їхнє застосування.

Почнемо з характеристики української специфіки. Комп'ютерні системи підтримки прийняття рішень, у принципі, можуть ґрунтуватися на двох підходах. Перший, більш традиційний, полягає в тому, що в системі фіксується досвід експерта, який використовується для вироблення, оптимального в даній ситуації, рішення. Системи інтелектуальних обчислень в основному реалізують другий підхід. Вони намагаються знайти рішення на основі аналізу історичних даних, що описують поведінку досліджуваного об'єкта, прийняті в минулому рішення, їхні результати і т.д. Усі ці дані можуть включати, наприклад, часові ряди цін на різні фінансові послуги, результати фінансово-господарської діяльності підприємства, статистику продажів тієї чи іншої продукції. Зрозуміло, щоб застосування цих систем у практиці виявилось виправданим, необхідно мати досить вагому множину цих даних - інакше прийняті на їхній основі рішення будуть безпідставними.

З цією очевидною обставиною зв'язані головні труднощі просування технологій інтелектуальних обчислень в України: відмінною рисою більшості вітчизняних підприємств є порівняно невеликий термін існування. Характерний "вік" накопичених ними баз даних складає 2-3 роки, і, як показує досвід, інформації, що міститься в цих базах даних, виявляється недостатньо для вироблення на її основі ефективної стратегії прийняття рішень за допомогою новітніх аналітичних систем. Небезпека тут складається не стільки в неможливості виявлення цікавлячих взаємозв'язків у нечисленних даних і побудови моделей на їхній основі, скільки в одержанні статистично незначущих моделей і прийнятті на їхній основі невірних рішень. Якщо даних мало, а їхня описова модель складна, то завжди можна підігнати цю модель під дані, навіть якщо це цілком випадкові числа. Той факт, що метод відмінно працює, коли потрібно пояснити те, що було в минулому, але зовсім непридатний для прийняття рішень "на майбутнє", народжує сумніву в здатності систем інтелектуальних обчислень вирішувати реальні задачі зі сфери бізнесу і фінансів. Таким чином, головна проблема застосування систем видобутку знань для України - це нечисленність аналізованих даних, а одне з головних вимог до цих систем - наявність твердого контролю статистичної значимості одержуваних результатів.

Іншою відмітною рисою української економіки, як на макрорівні, так і на рівні окремих підприємств, є її нестабільність; крім того, вона знаходиться під впливом дії численних, зненацька виникаючих факторів. У той час як на Заході підприємства в основному працюють у рамках вже устояної законодавчої бази, у сформованих структурах товарних, фінансових і інформаційних потоків, українські підприємства змушені підбудовуватися під правила гри, що постійно змінюються. Це ж стосується українських фінансових ринків, де приблизно раз у півроку відбувається істотне корегування правил роботи. Отже, людина повинна обов'язково контролювати й аналізувати результати, спродуковані інтелектуальними системами. Це потрібно, щоб гарантувати облік усіх факторів, що впливають на рішення. Як наслідок, побудовані моделі повинні бути прозорі і допускати інтерпретацію.

Нарешті, ще одна обставина впливає на застосування систем інтелектуальних обчислень в українських умовах. Воно зв'язано з тим, що люди, відповідальні за прийняття рішень у бізнесі і фінансах, звичайно не є фахівцями з статистиці і штучному інтелекті і тому не можуть безпосередньо використовувати системи інтелектуального аналізу даних, що вимагають складного налаштування чи спеціальної підготовки даних. Якщо така система поставляється як складова частина загальної технології електронних сховищ даних, реалізованої на підприємстві (що стає самою розповсюдженою практикою в розвинутих країнах), то це не створює проблеми - всі налаштування і передпроцесорна обробка здійснюються автоматично. Однак українські підприємства, що використовують сховища даних з елементами інтелектуального аналізу, сьогодні вкрай нечисленні. Тому важливими факторами, що визначають комерційний успіх систем інтелектуального аналізу даних в України, є простота у використанні і високому ступені автоматизму.

Засоби інтелектуальних обчислень знаходження нового знання та аналізу даних припускають надання допомоги організаціям у знаходженні прихованих залежностей у даних. Отримані моделі можна використовувати як для передбачення майбутніх значень, так і для опису поточного стану. Однак, засоби інтелектуальних обчислень не можуть працювати без супроводу користувачів, що добре розуміють ділову область, самі дані і загальний характер використовуваних аналітичних методів. Результат застосування методів знаходження нового знання може виявлятися в широкому спектрі, від збільшення доходів, до зменшення витрат.

Побудова моделі, це тільки один крок у процесі знаходження нового знання. Для отримання коректних результатів необхідно зібрати і підготувати дані і перевірити модель у реальному світі. Найкращу модель можна знайти після побудови моделей різних типів по різних технологіях.

Просунуті підприємці усвідомили, що засоби інтелектуальних обчислень - це реальний спосіб підвищення ефективності роботи. Питання не в тому, чи потрібні нові технології, а в тому, як їх застосувати в кожному конкретному випадку. Витрати на постановку задачі і супровід інтелектуальних систем можуть на порядок перевищувати вартість окремого пакета програм. Очевидно, що варто витратити частину грошей на навчання фахівців - у підсумку вийде дешевше й ефективніше. Зростає роль спеціалізованих консалтингових фірм, що здійснюють комплексний супровід проектів, включаючи діагностику задачі, аналіз методів рішення, вироблення рекомендацій, реалізацію обраного підходу, супровід, оптимізацію.

|  |  |
| --- | --- |
| |  | | --- | | **Тема 3. Основні моделі та методи технологій інтелектуальних обчислень**   * [Види моделей інтелектуальних обчислень](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_1)    + [Класифікація](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_1_1)   + [Регресійний аналіз](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_1_2)   + [Прогнозування часових послідовностей](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_1_3)   + [Кластеризація](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_1_4)   + [Асоціація](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_1_5)   + [Послідовність](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_1_6) * [Методи інтелектуальних обчислень](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_2)   + [Нейронні мережі](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_2_1)   + [Дерева рішень](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_2_2)   + [Системи міркування на основі аналогічних випадків](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_2_3)   + [Алгоритми виявлення асоціацій](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_2_4)   + [Нечітка логіка](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_2_5)   + [Генетичні алгоритми](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_2_6)   + [Еволюційне програмування](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_2_7)   + [Комбіновані методи](http://www.victoria.lviv.ua/html/oio/html/theme3.htm#3_2_8)   **Види моделей інтелектуальних обчислень**  Розглянемо основні види моделей, що використовуються для знаходження нового знання на основі даних інформаційного сховища. Метою інтелектуальних технології є знаходження нового знання, що користувач може надалі застосувати для поліпшення результатів своєї діяльності. Результат моделювання - це виявлені відношення в даних.  Можна виділити принаймні шість методів виявлення й аналізу знань:   * класифікація, * регресія, * прогнозування часових послідовностей (рядів), * кластеризація, * асоціація, * послідовність.   Перші три використовуються головним чином для передбачення, у той час як останні зручні для опису існуючих закономірностей в даних.  **Класифікація** є найбільш розповсюдженою операцією інтелектуального аналізу даних. З її допомогою виявляються ознаки, що характеризують групу, до якої належить той чи інший об'єкт. Це робиться за допомогою аналізу вже класифікованих об'єктів і формулювання деякого набору правил. В багатьох видах бізнесу проблемою є втрата постійних клієнтів. У різних сферах (таких, як мобільний телефонний зв'язок, фармацевтичний бізнес чи діяльність, пов'язана із кредитними картками) її позначають різними термінами - "зміною моди", "виснаженням попиту" чи "купівельною зрадою", - але суть при цьому одна.  Класифікація допоможе вам виявити характеристики "хитливих" покупців і створити модель, здатну передбачати, хто саме схильний піти до іншого постачальника. Використовуючи її, можна визначити самі ефективні види знижок і інших вигідних пропозицій, що будуть найбільш діючими для тих чи інших типів покупців. Завдяки цьому можна втримати клієнтів, витративши рівно стільки грошей, скільки необхідно. Один раз визначений ефективний класифікатор використовується для класифікації нових записів у базі даних у вже існуючі класи й в цьому випадку він здобуває характер прогнозу.  Наприклад, класифікатор, що вміє ідентифікувати ризик віддачі позики, може бути використаний для прийняття рішення, чи великий ризик надання позики визначеному клієнту. Тобто класифікатор використовується для прогнозування ймовірності повернення позики.  **Регресійний аналіз** використовується в тому випадку, якщо відношення між змінними можуть бути виражені кількісно у виді деякої комбінації цих змінних. Отримана комбінація використовується для передбачення значення, що може приймати цільова (залежна) змінна, яка обчислюється на заданому наборі значень вхідних (незалежних) змінних. У найпростішому випадку для цього використовуються стандартні статистичні методи, такі як лінійна регресія. На жаль, більшість реальних моделей не вкладаються в рамки лінійної регресії. Наприклад, розміри продажів чи фондові ціни дуже складні для передбачення, оскільки можуть залежати від комплексу взаємин множин змінних. Таким чином, необхідні комплексні методи для передбачення майбутніх значень.  **Прогнозування часових послідовностей**. Основою для всіляких систем прогнозування служить історична інформація, що зберігається в інформаційних сховищах у виді часових рядів. Якщо можна побудувати математичну модель і знайти шаблони, що адекватно відбивають цю динаміку, є імовірність, що з їх допомогою можна передбачати і поведінку системи в майбутньому. Прогнозування часових послідовностей дозволяє на основі аналізу поведінки часових рядів оцінити майбутні значення прогнозованих змінних. Звичайно, ці моделі повинні містити в собі особливі ознаки часу: ієрархія періодів (місяць-квартал-рік), особливі відрізки часу (п'яти- шести чи семиденний робочий тиждень), сезонність, свята й ін.  **Кластеризація** відрізняється від класифікації тим, що самі групи заздалегідь не задані. За допомогою моделі кластеризації засоби інтелектуальних обчислень самостійно виділяють різні однорідні групи даних.  **Асоціація** має місце в тому випадку, якщо кілька подій зв'язані між собою і адресована до класу проблем аналізу структури. Класичний приклад аналізу структури покупок відноситься до представлення придбання якої-небудь кількості товарів як одиночної економічної операції (транзакції). Оскільки велика кількість покупок відбувається в супермаркетах, а покупці для зручності використовують кошики, куди і складається весь товар, то найбільш відомим прикладом знаходження асоціацій є аналіз вмісту кошику.  Метою цього підходу є знаходження трендів (однакових ділянок) серед великого числа транзакцій, які можна використовувати для пояснення поведінки покупців. Така інформація може бути використана для регулювання запасів, зміни розміщення товарів на території магазину й прийняття рішення по проведенню рекламної кампанії для збільшення всіх продажів або для просування визначеного виду продукції, хоча цей підхід прийшов винятково з роздрібної торгівлі, він може також добре застосовуватися у фінансовій сфері для аналізу портфеля цінних паперів і знаходження наборів фінансових послуг, які клієнти часто придбають разом. Це може використовуватися для створення деякого набору послуг, як частини кампанії по стимулюванню продажів. Наприклад, дослідження, проведене в супермаркеті, може показати, що 65% купуючих картопляні чіпси, беруть також і "кока-колу", а при наявності знижки за такий комплект "колу" здобувають у 85% випадків. Маючи такі дані, менеджерам легко оцінити, наскільки діюча надана знижка.  **Послідовність** має місце, коли існує ланцюжок зв'язаних у часі подій. Традиційний аналіз структури покупок має справу з набором товарів, що представляють одну транзакцію. Варіант такого аналізу зустрічається, коли існує додаткова інформація (номер кредитної карти клієнта чи номер його банківського рахунка) для зв'язування різних покупок у єдину часову серію. В такій ситуації важливо не тільки співіснування даних всередині однієї транзакції, але і порядок, у якому ці дані з'являються в різних транзакціях і час між цими транзакціями. Правила, що встановлюють ці відношення, можуть бути використані для визначення типового набору попередніх продажів, що можуть повести за собою наступні продажі визначеного товару. Після покупки будинку в 45% випадків протягом місяця купується і нова кухонна плита, а в межах двох тижнів 60% новоселів обзаводяться холодильником.  Ці основні типи моделей використовуються для знаходження нового знання в сховищах даних.  Звернемося тепер до методів (алгоритмів), що використовуються для проведення інтелектуальних обчислень.  **Методи інтелектуальних обчислень**  Засоби інтелектуальних обчислень використовують наступні основні методи:   * нейронні мережі; * дерева рішень; * системи міркування на основі аналогічних випадків; * алгоритми визначення асоціацій і послідовностей; * нечітка логіка; * генетичні алгоритми; * еволюційне програмування; * візуалізація даних.   Іноді застосовується комбінація перерахованих методів.  **Нейронні мережі** відносяться до класу нелінійних адаптивних систем з архітектурою, що умовно імітує нервову тканину з нейронів. Математична модель нейрона являє собою деякий універсальний нелінійний елемент із можливістю широкої зміни і налаштування його характеристик. В одній з найбільш розповсюджених нейромережевих архитектур - багатошаровому перцептроні зі зворотним поширенням похибки - моделюється робота нейронів у складі ієрархічної мережі, де кожен нейрон прошарку з'єднаний своїми виходами з входами нейронів наступного прошарку. На нейрони вхідного прошарку подаються значення вхідних параметрів, на основі яких виробляються обчислення, необхідні для прийняття рішень, прогнозування розвитку ситуації і т.п. Ці значення розглядаються як сигнали, що передаються в наступний прошарок. Величина послаблення або підсилення сигналу залежить від числових значень (ваг), приписуваних міжнейронним зв'язкам. У результаті цього на виході нейрона вихідного прошарку продукується значення, що розглядається як відповідь, реакція всієї мережі на введені значення вхідних параметрів. Для того щоб мережу можна було застосовувати надалі, її треба "навчити" на прикладах, для яких відомо і значення вхідних параметрів, і правильні відповіді на них.  Процес "навчання" складається в підборі ваг межнейронних зв'язків і модифікації внутрішніх параметрів передатної функції нейронів. Для кожного сполучення навчальних даних на виході мережі вихідні значення порівнюються з відомим результатом. Якщо вони розрізняються, то обчислюється похибка, що враховується при обробці у вузлах мережі. Зазначені кроки повторюються, поки не виконається умова останову, наприклад необхідна похибка не буде перевищувати заданої величини.  Отже, нейронні мережі уявляють собою сукупність зв'язаних між собою вузлів, що отримують вхідні дані, здійснюють їх обробку і генерують на виході результат. Між вузлами видимих вхідного і вихідного прошарків може знаходитися певне число прихованих прошарків обробки. Нейронні мережі реалізують непрозорий процес. Це означає, що побудована модель, як правило, не має чіткої інтерпретації. Багато пакетів, що реалізують алгоритми нейронних мереж, застосовуються не лише в сфері обробки комерційної інформації, без них важко обійтися при рішенні більш загальних задач розпізнавання образів, скажемо розшифровки рукописного тексту чи інтерпретації кардіограм.  Апаратні або програмні реалізації алгоритмів нейромереж називаються нейрокомп'ютером.   * Нейрокомп'ютери дають стандартний спосіб рішення багатьох нестандартних задач. І неважливо, що спеціалізована машина краще вирішує один клас задач. Важливіше, що один нейрокомп'ютер вирішить і цю задачу, і другу, і третю і не треба щораз проектувати спеціалізовану ЕОМ, нейрокомп'ютер зробить все сам і майже не гірше. * Замість програмування навчання. Нейрокомп'ютер вчиться, потрібно лише формувати навчальні множини. Праця програміста заміняється новою працею вчителя. Краще це чи гірше? Ні те, ні інше. Програміст наказує машині всі деталі роботи, вчитель створює "навчальне середовище", до якого пристосовується нейрокомп'ютер. З'являються нові можливості для роботи. * Нейрокомп'ютери ефективні там, де потрібний аналог людської інтуїції, зокрема, для розпізнавання образів, читання рукописних текстів, підготовки аналітичних прогнозів, перекладу з однієї природної мови на іншу і т.п. Саме для таких задач звичайно важко скласти явний алгоритм. * Нейронні мережі дозволяють створити ефективне програмне та математичне забезпечення для комп'ютерів з високим ступенем розпаралелювання обробки. * Нейрокомп'ютери "демократичні", вони також дружні, як текстові процесори, тому з ними може працювати будь-який, навіть зовсім недосвідчений користувач.   **Дерева рішень** - це метод, придатний не тільки для рішення задач класифікації, але і для обчислень і тому досить широко застосовується в області фінансів і бізнесу, де частіше зустрічаються задачі чисельного прогнозу. В результаті застосування цього методу для навчальної вибірки даних створюється ієрархічна структура правил класифікації типу, "ЯКЩО... ТОДІ...", що має вид дерева. Для того щоб вирішити, до якого класу віднести деякий об'єкт або ситуацію, ми відповідаємо на питання, що стоять у вузлах цього дерева, починаючи з його кореня. Питання можуть мати вид "Значення параметра A більше Х ? " або виду "Значення змінної В належить підмножині ознак С ? ". Якщо відповідь позитивна, ми переходимо до правого вузла наступного рівня, якщо негативна - то до лівого вузла; потім знову відповідаємо на запитання, зв'язані з відповідним вузлом. Таким чином ми, зрештою, доходимо до одного з кінцевих вузлів - листів, де знаходиться вказівка, до якого класу треба віднести розглянутий об'єкт. Цей метод добрий тим, що таке представлення правил наочно і його легко зрозуміти.  Сьогодні спостерігається підйом інтересу до продуктів, що застосовують дерева рішень. В основному це пояснюється тим, що багато комерційних проблем розв'язується ними швидше, ніж алгоритмами нейронних мереж. До того ж вони більш прості і зрозумілі для користувачів.  У той же час не можна сказати, що дерева рішень завжди діють безвідмовно: для визначених типів даних вони можуть виявитися неприйнятними. Для дерев рішень дуже гостро постає проблема значимості. Справа в тому, що окремим вузлам на кожному новому побудованому рівні дерева відповідає все менше і менше число записів даних - дерево може сегментувати дані на велику кількість окремих випадків. Чим більше цих окремих випадків, чим менше навчальних прикладів попадає в кожен такий окремий випадок, тим менш надійної стає їх класифікація. Якщо побудоване дерево занадто "рунисте" - складається з невиправдано великого числа дрібних гілочок - воно не буде давати статистично обґрунтованих відповідей. Як показує практика, у більшості систем, що використовують дерева рішень, ця проблема не знаходить задовільного рішення.  **Системи міркування на основі аналогічних випадків**. Ідея алгоритму вкрай проста. Для того щоб зробити прогноз на майбутнє або вибрати правильне рішення, ці системи знаходять у минулому близькі аналоги наявної ситуації і вибирають ту ж відповідь, що була для них правильною. Тому цей метод ще називають методом "найближчого сусіда". Системи міркування на основі аналогічних випадків показують добрі результати в найрізноманітніших задачах. Головний їх мінус полягає в тому, що вони взагалі не створюють яких-небудь моделей або правил, що узагальнюють попередній досвід, - у виборі рішення вони ґрунтуються на всьому масиві доступних історичних даних, тому неможливо сказати, на основі яких конкретно факторів ці системи будують свої відповіді.  **Алгоритми виявлення асоціацій** знаходять правила про окремі предмети, що з'являються разом в одній економічній операції, наприклад в одній покупці. Послідовність - це теж асоціація, але залежна від часу.  Асоціація записується як А->Б, де А називається лівою частиною або передумовою, Б - правою частиною або наслідком.  Частота появи кожного окремого предмета або групи предметів, визначається дуже просто - підраховується кількість появи цього предмета у всіх подіях (покупках) і ділиться на загальну кількість подій. Ця величина вимірюється у відсотках і зветься "поширеність". Низький рівень поширеності (менш одного тисячної відсотка) говорить про те, що така асоціація не істотна.  Для визначення важливості кожного отриманого асоціативного правила необхідно одержати величину, що зветься "довірчість А до Б" (взаємозв'язок А і Б). Ця величина показує як часто з появою А з'являється Б и розраховується як відношення частоти появи (поширеності) А і Б разом до поширеності А. Тобто якщо довірчість А до Б дорівнює 20%, то це означає, що при покупці товару А в кожному п'ятому випадку придбають і товар Б.  Необхідно відзначити, що якщо поширеність А не дорівнює поширеності Б, то і довірчість А до Б не дорівнює довірчості Б к А. Справді, покупка комп'ютера частіше веде до покупки дискет, ніж покупка дискети до покупки комп'ютера.  Ще однією важливою характеристикою асоціації є потужність асоціації. Чим більше потужність, тим сильніше вплив, який поява А робить на появу Б. Потужність розраховується по формулі: (довірчість А до Б) / (поширеність Б).  Деякі алгоритми пошуку асоціацій спочатку сортують дані і тільки після цього визначають взаємозв'язок і поширеність. Єдиним розходженням таких алгоритмів є швидкість або ефективність знаходження асоціацій. Це важливо через величезну кількість комбінацій, які необхідно перебрати для знаходження найбільш значимих правил. Алгоритми пошуку асоціацій можуть створювати свої бази даних поширеності, довірчості і потужності, до яких можна звертатися при запиті. Наприклад: "Знайти всі асоціації, в яких для товару Х довірчість більш 50% і поширеність не менш 2,5%"  При знаходженні послідовностей додається змінна часу, що дозволяє працювати із серією подій для знаходження послідовних асоціацій протягом деякого періоду часу.  Підводячи підсумки цьому методу аналізу, необхідно сказати, що випадково може виникнути така ситуація, коли товари в супермаркеті будуть згруповані за допомогою знайдених моделей, але це, замість очікуваного прибутку, дасть зворотний ефект. Це може відбутись через те, що клієнт не буде довго ходити по магазині в пошуках бажаного товару, купуючи при цьому ще щось, що потрапляється на очі, і те, що він ніколи не планував придбати.  **Нечітка логіка** застосовується для таких наборів даних, де належність даних до якої-небудь групи є імовірністю в інтервалі від 0 до 1. Чітка логіка маніпулює результатами, що можуть бути або істиною, або не істиною. Нечітка логіка застосовується в тих випадках, коли необхідно маніпулювати ступенем "може бути" у доповненні до "так" або "ні". Областю впровадження алгоритмів нечіткої логіки є всілякі аналітичні системи, у тому числі:   * нелінійний контроль за процесами (виробництво); * вдосконалення стратегій керування і координації дій, наприклад складне промислове виробництво; * самонавчаючі системи (або класифікатори); * дослідження ризикових і критичних ситуацій; * розпізнавання образів; * фінансовий аналіз (ринки цінних паперів); * дослідження даних (корпоративні сховища).   У Японії цей напрямок переживає дійсний бум. Тут функціонує спеціально створена лабораторія Laboratory for International Fuzzy Engineering Research ( LIFE ). Програмою цієї організації є створення більш близьких людині обчислювальних пристроїв. LIFE поєднує 48 компаній у числі яких знаходяться: Hitachi, Mitsubishi, NEC, Sharp, Sony, Honda, Mazda, Toyota. З закордонних (не Японських) учасників LIFE можна виділити: IBM, Fuji Xerox, до діяльності LIFE також виявляє інтерес NASA.  Потужність і інтуїтивна простота нечіткої логіки як методології вирішення проблем гарантує її успішне використання у вбудованих системах контролю й аналізу інформації. При цьому відбувається підключення людської інтуїції і досвіду оператора.  На відміну від традиційної математики, що вимагає на кожному кроці моделювання точних і однозначних формулювань закономірностей, нечітка логіка пропонує зовсім інший рівень мислення, завдяки якому творчий процес моделювання відбувається на найвищому рівні абстракцій, при якому постулюється лише мінімальний набір закономірностей.  **Недоліками нечітких систем є:**   * відсутність стандартної методики конструювання нечітких систем; * неможливість математичного аналізу нечітких систем існуючими методами;   **Генетичні алгоритми** є могутнім засобом рішення різноманітних комбінаторних задач і задач оптимізації. Проте, генетичні алгоритми ввійшли зараз у стандартний інструментарій методів інтелектуальних обчислень. Цей метод названий так тому, що в якійсь степені імітує процес природного відбору в природі. Нехай нам треба знайти рішення задачі, найбільш оптимальне з погляду деякого критерію, де кожне рішення цілком описується певним набором чисел чи величин нечислової природи. Скажемо, якщо нам треба вибрати сукупність фіксованого числа параметрів ринку, найбільш виражено впливаючих на його динаміку, це буде набір імен цих параметрів. Про цей набір можна говорити як про сукупність хромосом, що визначають якості індивіда - даного рішення поставленої задачі. Значення параметрів, що визначають рішення, будуть тоді називатися генами. Пошук оптимального рішення при цьому схожий на еволюцію популяції індивідів, представлених їхніми наборами хромосом. У цій еволюції діють три механізми: по-перше, відбір найсильніших - наборів хромосом, яким відповідають найбільш оптимальні рішення; по-друге, схрещування - виробництво нових індивідів за допомогою змішування хромосомних наборів відібраних індивідів; і, по-третє, мутації - випадкові зміни генів у деяких індивідів популяції. У результаті зміни поколінь виробляється таке рішення поставленої задачі, що вже не може бути далі поліпшено.  Генетичні алгоритми мають два слабких місця. По-перше, сама постановка задачі в їхніх термінах не дає можливості проаналізувати статистичну значимість одержуваного з їх допомогою рішення і, по-друге, ефективно сформулювати задачу, визначити критерій відбору хромосом під силу тільки фахівцю. У силу цих факторів сьогодні генетичні алгоритми треба розглядати скоріше як інструмент наукового дослідження, ніж засіб аналізу даних для практичного застосування в бізнесі і фінансах.  **Еволюційне програмування** сьогодні наймолодша ділянка інтелектуальних обчислень. Суть методу в тому, що гіпотези про вид залежності цільової змінної від інших змінних формулюються системою у виді програм на деякій внутрішній мові програмування. Якщо це універсальна мова, то теоретично на ній можна виразити залежність будь-якого виду. Процес побудови цих програм будується як еволюція у світі програм (цим метод небагато схожий на генетичні алгоритми). Коли система знаходить програму, що досить точно виражає шукану залежність, вона починає вносити в неї невеликі модифікації і відбирає серед побудованих таким чином дочірніх програм ті, які підвищують точність. Отже, система "вирощує" кілька генетичних ліній програм, що конкурують між собою в точності вираження шуканої залежності. Спеціальний транслюючий модуль, переводить знайдені залежності з внутрішньої мови системи на зрозумілу користувачу мову (математичні формули, таблиці й ін.), роблячи їх легкодоступними. Для того щоб зробити отримані результати ще зрозумілішими для користувача-нематематика, є великий арсенал різноманітних засобів візуалізації виявлених залежностей.  Пошук залежності цільових змінних від інших проводиться у формі функцій якогось визначеного виду. Наприклад, в одному з найбільш вдалих алгоритмів цього типу - методі групового урахування аргументів (МГУА) залежність шукають у формі поліномів. Причому складні поліноми заміняються декількома більш простими, враховуючі тільки деякі ознаки (групи аргументів). Звичайно використовуються попарні об'єднання ознак. Цей метод не має істотних переваг у порівнянні з нейронними мережами з їх готовим набором стандартних нелінійних функцій, але, отримані формули залежності, у принципі, піддаються аналізу й інтерпретації (хоча на практиці все-таки буває занадто складна для цього).  **Комбіновані методи.** Часто виробники з'єднують зазначені підходи. Об'єднання в собі алгоритми нейронних мереж і технології дерев рішень повинно сприяти побудові більш точної моделі і підвищенню її швидкодії. Програми візуалізації даних у певному сенсі не є засобом аналізу інформації, оскільки вони тільки представляють її користувачу. Проте, візуальне представлення, скажемо, відразу чотирьох змінних досить виразно узагальнює надвеликі обсяги даних. Отже виробники розуміють, що для рішення кожної проблеми варто застосовувати оптимальний метод. | |

|  |  |
| --- | --- |
| |  | | --- | | **Тема 4. Процес знаходження нового знання**   * [Визначення проблеми (постановка задачі);](http://www.victoria.lviv.ua/html/oio/html/theme4.htm#4_1) * [Збір та підготовка даних:](http://www.victoria.lviv.ua/html/oio/html/theme4.htm#4_2)    + [Оцінка даних;](http://www.victoria.lviv.ua/html/oio/html/theme4.htm#4_2_1)   + [Об'єднання й очищення даних;](http://www.victoria.lviv.ua/html/oio/html/theme4.htm#4_2_2)   + [Відбір даних;](http://www.victoria.lviv.ua/html/oio/html/theme4.htm#4_2_3)   + [Перетворення;](http://www.victoria.lviv.ua/html/oio/html/theme4.htm#4_2_4) * [Побудова моделі;](http://www.victoria.lviv.ua/html/oio/html/theme4.htm#4_3)    + [Оцінка й інтерпретація;](http://www.victoria.lviv.ua/html/oio/html/theme4.htm#4_3_1)   + [Зовнішня перевірка;](http://www.victoria.lviv.ua/html/oio/html/theme4.htm#4_3_2) * [Використання моделі;](http://www.victoria.lviv.ua/html/oio/html/theme4.htm#4_4) * [Спостереження за моделлю;](http://www.victoria.lviv.ua/html/oio/html/theme4.htm#4_5)   Для того щоб знайти нове знання на основі даних великого сховища недостатньо просто взяти алгоритми Data Mining, запустити їх і чекати появи цікавих результатів. Знаходження нового знання - це процес, що містить у собі кілька кроків, кожний з яких необхідний для ефективного застосування засобів інтелектуальних обчислень. Основні етапи цього процесу наступні:   1. визначення проблеми (постановка задачі); 2. збір та підготовка даних:    * оцінка даних;    * об'єднання й очищення даних;    * відбір даних;    * перетворення; 3. побудова моделі;    * оцінка й інтерпретація;    * зовнішня перевірка; 4. використання моделі; 5. спостереження за моделлю;   Зупинимося докладніше на кожному з цих етапів:  **1. Визначення проблеми.** Для того щоб найбільш повно використати всі переваги інтелектуальних технологій необхідно ясно представити мету майбутнього аналізу. Побудова моделі проводиться в залежності від мети. Якщо необхідно збільшити прибуток торгової організації, то для цілей: "збільшення кількості продажів" і "збільшення ефективності реклами" необхідно будувати різні моделі. На цьому ж етапі визначаються способи оцінки результатів майбутнього проекту і можливі витрати на його реалізацію.  **2. Збір та підготовка даних.** Самий довготривалий етап: може займати від 50% до 85% часу всього процесу знаходження нового знання. На цьому етапі необхідно визначити джерела отримання даних. Це можуть бути дані, накопичені самою організацією або зовнішні дані від загальнодоступних джерел (відомості про погоду чи перепис населення) або приватних джерел (різні архівні дані, бази нотаріальних контор і ін.).  **а. оцінка даних.** При побудові моделі необхідно пам'ятати одне правило, що стосується коректності вхідних даних: "Якщо на вхід задачі надходить "сміття", то і результатом теж буде "сміття". Вхідні дані можуть знаходитися в одній базі або в декількох. Перед "завантаженням" ' даних у сховище необхідно врахувати, що різні джерела даних можуть бути спроектовані під визначені задачі і, відповідно, виникають проблеми, пов'язані з об'єднанням даних: різні формати представлення числових даних (наприклад, цілі або дійсні); різне кодування даних (наприклад, різний формат дати); різні способи збереження даних; різні одиниці виміру (дюйми й сантиметри); а також частота збору даних і дата останнього оновлення.  Навіть, якщо дані знаходяться в одній базі, то все одне треба звертати пильну увагу на пропущені значення і значення, нереальної величини, так звані "викиди".  Аналітик повинен завжди знати, як, де і при яких умовах збираються дані, і бути впевненим, що всі дані, які використовуються для проведення аналізу виміряні однаковим способом.  **б. об'єднання й очищення даних.** На цьому етапі відбувається побудова сховища даних, що буде піддаватися подальшій обробці, тобто, відбувається наповнення сховища чи долучення до нього тих даних, що були відібрані на попередніх етапах. У цей же час відбувається очищення, тобто виправлення всіх виявлених помилок. Існують різні аспекти очищення даних. Усі вони спрямовані на знаходження і виправлення помилок, що були допущені ще на етапі збору інформації. Помилкою в даних можуть вважатися:   * пропущене значення; * неможлива подія (невірно набране значення - "викид").   Корекція відбувається на основі здорового глузду, використання правил і/або із залученням експерта, що добре знає предметну область. Тобто запис у базі даних, в якому є помилка, може бути виправлений чи, у спірних випадках, виключений з подальшого розгляду.  Після перевірки даних, вони перетворюються і форматуються відповідно до результатів оцінки. Це робиться для більшої зручності спостереження за даними. Дані дискретних подій перетворюються в спеціально розроблену чи стандартну форму, в якій відбиваються час і опис подій. Коли користувачі будуть легко розбиратися в цій формі, вони зможуть швидко вивчити події, що були в основі побудови цієї форми. Може здатися, що цей крок дублює етап збору даних, але насправді це два зовсім різні етапи. На першому з них відбувається відбір даних для прискорення машинної обробки інформації без втрати якості, на другому дані приводяться до виду, зручного для візуального контролю користувача.  Тепер людина, що проводить аналіз може найбільш повно уявити собі вхідні дані. Це буває необхідно для різного роду звітів, коли необхідно коротко охарактеризувати вхідні дані, що застосовуються для аналізу.  **в. відбір даних.** Коли сформовано сховище і визначено типи моделей, які будуть побудовані для рішення задачі, відбувається відбір даних необхідних саме для цих моделей. Мається на увазі не тільки зменшення кількості записів у базі по визначеній умові, але також і зміна кількості полів, злиття різних таблиць в одну чи навпаки створення на основі однієї таблиці декількох. Тобто перетворення відбувається у "трьох вимірах": по кількості записів, по кількості полів і за структурою.  **г. перетворення даних** має за мету збагачення отриманої бази, тобто додавання різних відношень на основі існуючих полів (не просто "ціна" і "кількість", а їхній добуток - "загальна сума", не борг і дохід, а відношення боргу до доходу), додавання інтервалів (по номеру місяця можна поставити номер кварталу, а відсоток виконання плану можна доповнити характеристиками "добре", "задовільно"), додавання критичних значень (максимум, середнє, мінімум).  **3. Побудова моделі.** Найважливіше, про що завжди потрібно пам'ятати - це те, що побудова будь-якої моделі представляє собою ітераційний процес. Тобто необхідно побудувати ряд моделей для знаходження однієї, найбільш задовольняючої поставленим цілям.  Моделі можна розділити на дві групи: контрольовані (моделі класифікації, регресії, прогнозування часових послідовностей) і неконтрольовані (кластеризація, асоціація і послідовність). Після того, як визначено тип моделі, необхідно вибрати алгоритм побудови моделі, чи технологію знаходження знання.  Сутність процесу побудови контрольованої моделі зводиться до знаходження залежностей на одній частині даних ("навчання моделі") і перевірки цих залежностей на іншій частині даних (оцінка точності). Модель вважається побудованою, коли завершується цикл "навчання" і перевірок. Якщо точність моделі при чергових ітераціях не поліпшується, то це говорить про завершення побудови моделі.  Оскільки "навчальні" і тестові дані знаходяться в одній базі даних, то часто виникає необхідність у третьому наборі даних - контрольному, який вибирається з таких даних, що не перетинаються з "навчальними" та тестовими. Він потрібен для незалежного оцінювання точності моделі. Як правило всі три набори даних належать тій самій множині даних, необхідної для реалізації визначеного проекту.  Найбільш відомий тестовий метод - називається простою оцінкою. У цьому випадку розподіл даних на два набори відбувається випадковим чином. Відношення кількості тестових даних до кількості даних, на яких відбувається побудова моделі повинен бути в межах від 5% до 33%. Після побудови моделі, її використовують для передбачення значень на тестовому наборі. Мірою точності моделі вважають відношення кількості вдалих результатів до загальної кількості прикладів у тестовому наборі (можна використовувати таку змінну, як міра неточності, що дорівнює 1 - "міра точності").  Якщо для побудови моделі використовується не дуже велика база даних, то застосовується так звана перехресна оцінка точності. У цьому випадку дані випадковим образом поділяються на дві приблизно рівні частини. Після цього модель будуватиметься на одній з них, а інша використовується для визначення міри точності. Потім частини бази міняються ролями. Отримані дві незалежні оцінки точності поєднуються (як середнє арифметичне чи іншим способом) для найкращої оцінки міри точності моделі, побудованої на всій базі.  Для ще менших баз, у кілька тисяч записів, використовується n-перехресна оцінка точності. У цьому випадку база поділяється на n приблизно рівних груп, що не перетинаються. Далі перша з цих груп стає тестовим набором, а інші групи поєднуються, і на їхній основі відбувається побудова моделі. Отримана модель використовується для передбачення значень для тестового набору і таким чином виходить перша міра точності. Аналогічним образом розраховуються всі n незалежні міри точності. Середнє з них є мірою точності всієї моделі.  Ще один спосіб використовується для знаходження міри точності в малих базах даних. У цьому випадку модель будуватиметься на основі даних усієї бази. Після цього випадковим чином із записів бази створюється множина тестових наборів (мінімум 200, а іноді навіть більше 1000). Один запис може бути присутнім у різних тестових наборах. Для кожного з них визначається міра точності. Знову ж середнє з них є мірою точності всієї моделі.  Після того як побудова моделі завершена, можна побудувати модель, використовуючи інші параметри, чи навіть змінити алгоритм побудови моделі, тому що ніколи не можна сказати, який алгоритм, яка технологія знаходження знання дасть найкращі результати. Не можна бути впевненим, що визначена технологія буде працювати найкраще. Найчастіше доводиться будувати велику кількість моделей і для кожної проводити процедуру оцінки для знаходження найкращої. Крім цього, для різних моделей необхідна різна підготовка даних, отже, неминуче повторення кроків. Все це збільшує час знаходження кращої моделі, тому необхідно застосовувати технології паралельних обчислень.  **а. оцінка й інтерпретація.** Після побудови моделі необхідно оцінити результати і пояснити (інтерпретувати) їхню значимість. При оцінці моделі обчислюється міра точності, але треба пам'ятати, що це значення вірне лише до тих даних, на яких модель побудована і бути готовим, що нові дані, до яких надалі буде застосовуватися модель, можуть відрізнятися від вихідних невідомим чином.  **б. зовнішня перевірка.** Висока міра точності моделі не є гарантією того, що модель правильно відбиває реальне середовище. Однією з причин для цього є існування так званих неявних припущень у моделі. Тобто сам по собі коефіцієнт інфляції не може бути частиною моделі, що пояснює схильність покупців до покупки того чи іншого товару, але різка зміна цього коефіцієнта з 3% до 20% уже, напевно, може пояснити таку поведінку.  Інша причина - це існування неминучих проблем з даними, що можуть привести до некоректності моделі, тому дуже важливо перевірити модель у реальному середовищі. Наприклад, якщо модель використовується для відбору кандидатів для цільової реклами, то можна зробити тестове розсилання для перевірки моделі на невеликому обсязі даних. Якщо модель використовується для передбачення ризику неповернення кредиту, то варто буде випробувати цю модель на невеликій кількості претендентів на позичку. Чим більше ризик, зв'язаний з некоректністю моделі, тим більш важливо провести попередні експерименти для перевірки моделі перед початком її повної експлуатації.  **4. Використання моделі.** Після побудови й оцінки моделі вона може бути використана різними способами. Наприклад, аналітик може подивитися групи, що визначила модель кластеризації, графіки ефективності моделі чи отримані правила. Іноді аналітик може використовувати модель для вибору деяких записів з бази даних, щоб провести додатковий аналіз.  Ґрунтуючись на результатах такого використання моделі, аналітик може рекомендувати дії, які можна почати в діловій сфері. Однак, часто технології інтелектуальних обчислень - це частина автоматизованої системи (наприклад, знаходження кредитних ризиків, визначення можливості втрати клієнтів і ін.), тобто модель вбудовується в систему, яку аналітик або менеджер може застосовувати для прийняття рішення. З іншої сторони модель можна включати в систему, що генерує деяку дію (наказ), якщо прогнозована величина починає виходити за межі якихось значень.  В єдиному застосуванні, методи інтелектуальних обчислень, це невелика, хоча і важлива частина кінцевого програмного продукту. Процедура знаходження знання за допомогою таких методів може об'єднуватися зі знаннями експертів і застосовуватися до даних у базі.  **5. Спостереження за моделлю.** Коли модель починає працювати в реальному середовищі, то необхідно виміряти міру точності моделі на реальних даних. Однак, навіть якщо модель працює добре, і можна вважати, що робота на цьому закінчується, те все ж таки необхідно продовжувати спостереження за моделлю. Всі системи мають властивість розвиватися, і отримані дані (їхня структура, точність, періодичність) теж міняються. Зовнішні змінні, такі як коефіцієнт інфляції, своєю зміною теж можуть впливати на поведінку людей і на фактори, що впливають на цю зміну. Таким чином, час від часу модель необхідно піддавати процедурі повторного тестування, і навіть перебудовування.  Найпростішим способом спостереження за результатами діяльності моделі є графіки розходжень між передбаченими величинами і реальними значеннями. Вони прості для побудови і розуміння і можуть вбудовуватися в програмні продукти, отже, така автоматизована система може стежити сама за собою й оповіщати користувача, якщо величина цих розходжень починає виходити за визначений граничний рівень. | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Тема 5. Штучні нейронні мережі**   * [Історія нейронних мереж](http://www.victoria.lviv.ua/html/oio/html/theme5.htm#5_1) * [Аналогія з мозком](http://www.victoria.lviv.ua/html/oio/html/theme5.htm#5_2)    + [Біологічний нейрон](http://www.victoria.lviv.ua/html/oio/html/theme5.htm#5_2_1)   + [Штучний нейрон](http://www.victoria.lviv.ua/html/oio/html/theme5.htm#5_2_2)   + [Штучні нейронні мережі](http://www.victoria.lviv.ua/html/oio/html/theme5.htm#5_2_3) * [Навчання штучної нейронної мережі](http://www.victoria.lviv.ua/html/oio/html/theme5.htm#5_3) * [Обґрунтованість застосування нейромереж](http://www.victoria.lviv.ua/html/oio/html/theme5.htm#5_4)   Інтелектуальні системи на основі штучних нейронних мереж дозволяють з успіхом вирішувати проблеми розпізнавання образів, виконання прогнозів, оптимізації, асоціативної пам'яті і керування. Традиційні підходи до рішення цих проблем не завжди надають необхідної гнучкість і багато застосувань виграють від використання нейромереж.  Штучні нейромережі є електронними моделями нейронної структури мозку, який, головним чином, навчається з досвіду. Природній аналог доводить, що множина проблем, які поки що  не підвладні розв'язуванню наявними комп'ютерами, можуть бути ефективно вирішені блоками  нейромереж.  Тривалий період еволюції додав мозку людини багато якостей, що відсутні в сучасних  комп'ютерах з архітектурою фон Неймана. До них відносяться:   * розподілене представлення інформації і паралельні обчислення; * здатність до навчання й узагальнення; * адаптивність; * толерантність до помилок * низьке енергоспоживання.   Прилади, побудовані на принципах біологічних нейронів, мають перелічен  і характеристики, що можна вважати суттєвим здобутком у індустрії обробки даних.  Досягнення в галузі нейрофізіології надають початкове розуміння механізму  природного мислення, де збереження інформації відбувається у вигляді образів, деякі з  яких є складними. Процес зберігання інформації як образів, використання образів і вирішення поставленої проблеми визначають нову галузь в обробці даних, яка, не використовуючи традиційного програмування, забезпечує створення паралельних мереж та їх навчання. В лексиконі розробників та користувачів нейромереж присутні слова, дуже відмінні від традиційної обробки даних, зокрема, "вести себе", "реагувати", "самоорганізовувати", "навчати", "узагальнювати" та "забувати".  **Історія нейронних мереж**  Вивченню людського мозку - тисячі років. З появою сучасної електроніки, почались спроби  апаратного відтворення процесу мислення. Перший крок був зроблений у 1943 р.  з виходом статті нейрофізіолога Уоррена Маккалоха (Warren McCulloch) і  математика Уолтера Піттса (Walter Pitts) про роботу штучних нейронів і представлення  моделі нейронної мережі на електричних схемах.   * В 1949 р. опублікована книга Дональда Хебба (Donald Hebb) * "Організація поведінки". В ній досліджена проблематика налаштування синаптичних зв'язків. * В 1950-х рр. з'являються програмні моделі штучних нейромереж.   Перші роботи провів Натаніел Рочестер (Nathanial Rochester) з дослідної лабораторії  IBM. І хоча пізніші реалізації були успішними, його модель зазнала невдачі,  оскільки бурхливий зріст традиційних обчислень залишив у затінку нейронні дослідження.   * В 1956 р. Дартмутський дослідний проект з штучного інтелекту забезпечив   підйом штучного інтелекту, зокрема, нейронних мереж. Стимулювання  досліджень штучного інтелекту розгалузилось у двох напрямках:  промислові застосування систем штучного інтелекту (експертні системи) та моделювання мозку.   * В 1958 р. Джон фон Нейман (John fon Neumann) запропонував імітацію простих функцій нейронів із використанням вакуумних трубок. * У 1959 р. Бернард Відров (Bernard Widrow) та Марсіан Хофф (Marcian Hoff) розробили моделі ADALINE та MADALINE (Множинні Адаптивні Лінійні Елементи (Multiple ADAptive LINear Elements)). MADALINE діяла, як адаптивний фільтр, що усував відлуння   на телефонних лініях. Ця нейромережа досі в комерційному використанні.   * Нейробіолог Френк Розенблатт (Frank Rosenblatt) почав роботу над перцептроном.   Одношаровий перцептрон був збудований апаратно і вважається класичною нейромережею. На той час перцептрон використовувався у класифікації множини вхідних сигналів у один  з двох класів. На жаль, одношаровий перцептрон був обмеженим і зазнав критиці у 1969 р., у книзі Марвіна Мінскі (Marvin Minsky) та Сеймура Пейперта (Seymour Papert) "Перцептрони".  Ранні успіхи, були підставою того, що люди перебільшили потенціал нейронних мереж,  зокрема в світлі обмеженої на ті часи електроніки. Надмірне сподівання,  яке квітнуло у академічному та технічному світах, заразило загальну літературу  цього часу. Побоювання у тому, як ефект "мислячої машини" відіб'ється на людині весь час підігрівався письменниками, зокрема, серія книг Азімова про роботів показала  наслідки на моральних цінностях людини, у випадку спроможності інтелектуальних роботів виконувати функції людини.  Ці побоювання, об'єднані з невиконаними обіцянками, викликали множину розчарувань фахівців, які критикували дослідження нейронних мереж. Результатом було припинення більшості фінансування. Цей період спаду продовжувався до 80-х років.   * У 1982 р. відновлення інтересу спричинило декілька подій. Джон Хопфілд (John Hopfield) представив статтю до національної Академії Наук США. Підхід Хопфілда створював коренево нові підходи до моделювання. * У той самий час у Кіото (Японія) відбулась Об'єднана американо-японська конференція по нейронних мережах, які оголосили досягненням п'ятої генерації. Американські періодичні видання підняли цю історію, акцентуючи, що США можуть залишитись позаду, що привело до зросту фінансування в галузі нейромереж. * З 1985 р. Американський Інститут Фізики розпочав щорічні зустрічі - "Нейронні мережі для обчислень". * В 1989 р. на зустрічі "Нейронні мережі для оборони" Бернард Відров повідомив аудиторії про початок четвертої світової війни, де полем бою є світові ринки та виробництва. * У 1990 р. Департамент програм інноваційних досліджень захисту малого бізнесу   назвав 16 основних та 13 додаткових тем, де потрібне та можливе використання  нейронних мереж.  Сьогодні, обговорення нейронних мереж відбуваються скрізь. Перспектива їх використання видається досить яскравою, в світлі вирішення нетрадиційних проблем і є ключем до цілої технології.  На даний час більшість розробок нейронних мереж принципово працюючі,  але можуть існувати процесорні обмеження. Дослідження скеровані на програмні та  апаратні реалізації нейромереж. Компанії працюють над створенням трьох типів нейрочіпів:  цифрових, аналогових та оптичних, що обіцяють бути хвилею близького майбутнього.  **Аналогія з мозком**  Точна робота мозку людини - все ще таємниця. Проте деякі аспекти цього дивовижного  процесора відомі. Базовим елементом мозку людини є специфічні клітини, відомі як нейрони,  що здатні запам'ятовувати, думати і застосовувати попередній досвід до кожної дії,  що коренево відрізняє їх від решта клітин тіла.  Кора головного мозку людини є протяжною, утвореною нейронами поверхнею товщиною від 2  до 3 мм із площею близько 2200 см2, що вдвічі перевищує площу поверхні стандартної  клавіатури. Кора головного мозку містить близько 1011 нейронів, що приблизно дорівнює  числу зірок Чумацького шляху. Кожен нейрон зв'язаний з 103 - 104 іншими нейронами.  У цілому мозок людини містить приблизно від 1014 до 1015 взаємозв'язків.  Сила людського розуму залежить від числа базових компонент, різноманіття з'єднань між ними,  а також від генетичного програмування й навчання.  Індивідуальний нейрон є складним, має свої складові, підсистеми та механізми керування і передає інформацію через велику кількість електрохімічних зв'язків. Налічують біля сотні різних  класів нейронів. Разом нейрони та з'єднання між ними формують недвійковий, нестійкий та  несинхронний процес, що різниться від процесу обчислень традиційних комп'ютерів.  Штучні нейромережі моделюють лише найголовніші елементи складного мозку,  що надихає науковців та розробників до нових шляхів розв'язування проблеми.  **Біологічний нейрон**  Нейрон (нервова клітка) складається з тіла клітини - соми (soma), і двох типів зовнішніх деревоподібних відгалужень: аксона (axon) і дендритів (dendrites). Тіло клітини вміщує ядро (nucleus), що містить інформацію про властивості нейрона, і плазму, яка продукує необхідні для нейрона матеріали. Нейрон отримує сигнали (імпульси) від інших нейронів через дендрити (приймачі) і передає сигнали,  згенеровані тілом клітки, вздовж аксона (передавач), що наприкінці розгалужується на волокна (strands). На закінченнях волокон знаходяться синапси (synapses).  http://www.victoria.lviv.ua/html/oio/images/theme5/rys1.gif  Рис. 1. Схема біологічного нейрона  Синапс є функціональним вузлом між двома нейронами (волокно аксона одного нейрона і  дендрит іншого). Коли імпульс досягає синаптичного закінчення, продукуються хімічні речовини, названі нейротрансмітерами. Нейротрансмітери проходять через синаптичну щілину,  збуджуючи або гальмуючи, у залежності від типу синапсу,  здатність нейрона-приймача генерувати електричні імпульси. Результативність синапсу  налаштовується минаючими через нього сигналами, тому синапси навчаються в  залежності від активності процесів, у яких вони приймають участь. Нейрони взаємодіють  за допомогою короткої серії імпульсів. Повідомлення передається за допомогою  частотно-імпульсної модуляції.  Останні експериментальні дослідження доводять, що біологічні нейрони  структурно складніші, ніж спрощене пояснення, наведене вище і значно складніші,  ніж існуючі штучні нейрони, які є елементами сучасних штучних нейронних мереж.  Оскільки нейрофізіологія надає науковцям розширене розуміння дії нейронів,  а технологія обчислень постійно вдосконалюється, розробники мереж мають  необмежений простір для вдосконалення моделей біологічного мозку.  **Штучний нейрон**  Базовий модуль нейронних мереж штучний нейрон моделює основні функції  природного нейрона (рис. 2).  http://www.victoria.lviv.ua/html/oio/images/theme5/rys2.gif  Рис. 2. Базовий штучний нейрон  Вхідні сигнали xn зважені ваговими коефіцієнтами з'єднання wn додаються, проходять  через передатну функцію, генерують результат і виводяться. У наявних на цей час  пакетах програм штучні нейрони називаються "елементами обробки" і мають набагато  більше можливостей, ніж простий штучний нейрон, описаний вище. На рис. 3  зображена детальна схема спрощеного штучного нейрону.  http://www.victoria.lviv.ua/html/oio/images/theme5/rys3.gif  Рис. 3. Модель "елементу обробки"  Модифіковані входи передаються на функцію сумування, яка переважно тільки сумує добутки.  Проте можна обрати багато різних операцій, такі як середнє, найбільше, найменше, OR, AND,  тощо, які могли б виробляти деяку кількість різних значень. Окрім того, більшість  комерційних програм дозволяють інженерам-програмістам створювати власні функції  суматора за допомогою підпрограм, закодованих на мові високого рівня (C, С++, TurboPascal). Інколи функція сумування ускладнюється додаванням функції активації, яка дозволяє функції  сумування оперувати в часі.  В любому з цих випадків, вихід функції сумування надсилається у передатну функцію і скеровує  весь ряд на дійсний вихід (0 або 1, -1 або 1, або яке-небудь інше число) за допомогою  певного алгоритму. В існуючих нейромережах в якості передатних функцій можуть бути використані сигмоїда, синус, гіперболічний тангенс та ін. Приклад того, як працює передатна функція показаний на рис. 4.  http://www.victoria.lviv.ua/html/oio/images/theme5/rys4.gif  Рис. 4. Сигмоїдна передаточна функція  Після обробки сигналу, нейрон на виході має результат передатної функції, який надходить на  входи інших нейронів або до зовнішнього з'єднання, як це передбачається структурою нейромережі.  Всі штучні нейромережі конструюються з базового формуючого блоку - штучного  нейрону. Існуючі різноманітності і фундаментальні відмінності, є підставою  мистецтва талановитих розробників для реалізації ефективних нейромереж.  **Штучні нейронні мережі**  Інша частина створення і використання нейромереж стосується нескінченої кількості зв'язків,  що пов'язують окремі нейрони. Групування у мозку людини відбувається так, що  інформація обробляється динамічним, інтерактивним та самоорганізуючим шляхом.  Біологічні нейронні мережі створені у тривимірному просторі з мікроскопічних  компонент і здатні до різноманітних з'єднань. Але для створеної людиною мережі існують фізичні обмеження.  Існуючі на даний час, нейромережі є групуванням штучних нейронів.  Це групування обумовлено створенням з'єднанних між собою прошарків.  http://www.victoria.lviv.ua/html/oio/images/theme5/rys5.gif  Рис. 5. Діаграма простої нейронної мережі  На рис. 5 показана типова структура штучних нейромереж. Хоча існують мережі,  які містять лише один прошарок, або навіть один елемент,  більшість застосувань вимагають мережі, які містять як мінімум три нормальних типи  прошарків - вхідний, прихований та вихідний. Прошарок вхідних нейронів отримує дані  або з вхідних файлів, або безпосередньо з електронних давачів.  Вихідний прошарок пересилає інформацію безпосередньо до зовнішнього середовища,  до вторинного комп'ютерного процесу, або до інших пристроїв.  Між цими двома прошарками може бути багато прихованих прошарків, які містять  багато нейронів у різноманітних зв'язаних структурах. Входи та виходи кожного з  прихованих нейронів просто йдуть до інших нейронів.  Напрямок зв'язку від одного нейрону до іншого є важливим аспектом нейромереж. У більшості  мереж кожен нейрон прихованого прошарку отримує сигнали від всіх нейронів  попереднього прошарку та звичайно від нейронів вхідного прошарку. Після  виконання операцій над сигналами, нейрон передає свій вихід до всіх нейронів наступних  прошарків, забезпечуючи шлях передачі вперед (feedforward) на вихід.  При зворотньому зв'язку, вихід нейронів прошарку скеровується до нейронів  попереднього прошарку (рис. 6).  http://www.victoria.lviv.ua/html/oio/images/theme5/rys6.gif  Шлях, яким нейрони з'єднуються між собою має значний вплив на роботу мережі. Більшість пакетів професіональної розробки програмного забезпечення дозволяють користувачу  додавати, вилучати та керувати з'єднаннями як завгодно. Постійно коректуючі параметри,  зв'язки можна робити як збуджуючими так і гальмуючими.  **Навчання штучної нейронної мережі**  Здатність до навчання є фундаментальною властивістю мозку. Процес навчання  може розглядатися як визначення архітектури мережі і налаштування ваг зв'язків  для ефективного виконання спеціальної задачі. Нейромережа налаштовує ваги зв'язків  по наявній навчальній множині. Властивість мережі навчатися на прикладах робить їх більш привабливими в порівнянні із системами, які функціонують згідно визначеній системі правил, сформульованої  експертами.  Для процесу навчання необхідно мати модель зовнішнього середовища, у якій функціонує  нейронна мережа - потрібну для вирішення задачі інформацію. По-друге, необхідно  визначити, як модифікувати вагові параметри мережі. Алгоритм навчання означає  процедуру, в якій використовуються правила навчання для налаштування ваг.  Існують три загальні парадигми навчання: "з вчителем", "без вчителя" (самонавчання) і змішана.  У першому випадку нейромережа має у своєму розпорядженні правильні відповіді  (виходи мережі) на кожен вхідний приклад. Ваги налаштовуються так, щоб мережа  виробляла відповіді як можна більш близькі до відомих правильних відповідей.  Навчання без вчителя не вимагає знання правильних відповідей на кожен приклад навчальної  вибірки. У цьому випадку розкривається внутрішня структура даних та кореляція між зразками  в навчальній множині, що дозволяє розподілити зразки по категоріях.  При змішаному навчанні частина ваг визначається за допомогою навчання зі вчителем, у той час як інша визначається за допомогою самонавчання.  **Обґрунтованість застосування нейромереж**  Нейромережі не можна вважати доцільним рішенням для всіх обчислювальних проблем.  Традиційні комп'ютери та обчислювальні методи є ідеальними для багатьох застосувань.  Сучасні цифрові обчислювальні машини перевершують людину по здатності робити числові  й символьні обчислення. Однак людина може без зусиль вирішувати складні задачі  сприйняття зовнішніх даних (наприклад, впізнавання людини в юрбі по його обличчю)  з такою швидкістю і точністю, що наймогутніший у світі комп'ютер у порівнянні з ним здається  безнадійним тугодумом. У чому причина настільки значного розходження в їхній продуктивності?  **Машина фон Неймана у порівнянні з біологічної нейроною системою**   |  |  |  | | --- | --- | --- | | \*\* | Машина фон Неймана | Біологічна нейрона система | | Процесор | Складний  Високошвидкісний  Один чи декілька | Простий  Низькошвидкісний  Велика кількість | | Пам'ять | Відділена від процесора Локалізована  Адресація за адресою | Інтегрована в процесор Розподілена  Адресація по змісту | | Обчислення | Централізовані  Послідовні Збережені програми | Розподілені  Паралельні  Самонавчання | | Надійність | Висока вразливість | Живучість | | Спеціалізація | Числові й символьні операції | Проблеми сприйняття | | Середовище функціонування | Строго визначене  Строго обмежене | Погано визначене  Без обмежень | | Функції | Логічно,  через правила,  концепції, обчислення | Через зображення,  рисунки, керування | | Метод навчання | За правилами (дидактично) | За прикладами (сократично) | | Застосування | Числова та символьна обробка інформації | Розпізнавання мови, розпізнавання образів, розпізнавання текстів |   Представимо деякі проблеми, розв'язувані в контексті нейромоделювання, які представляють інтерес для вчених і інженерів.  **Класифікація образів.** Завдання полягає у визначенні приналежності вхідного образа (наприклад, мовного сигналу чи рукописного символу), представленого вектором ознак,  одному чи декільком попередньо визначеним класам. До відомих застосувань  відносяться розпізнавання букв, розпізнавання мови, класифікація сигналу електрокардіограми, класифікація кліток крові.  **Кластеризація/категоризація.** При рішенні задачі кластеризації, що відома також  як класифікація образів "без вчителя", навчальна множина з визначеними класами відсутня.  Алгоритм кластеризації заснований на подобі образів і розміщує близькі образи в один кластер. Відомі випадки застосування кластеризації для видобутку знань, стиснення даних  і дослідження властивостей даних.  **Апроксимація функцій.** Припустимо, що є навчальна вибірка ((x1,y1), (x2,y2)..., (xn,yn))  (пари даних вхід-вихід), яка генерується невідомою функцією F, спотвореної шумом.  Завдання апроксимації полягає в знаходженні невідомої функції F. Апроксимація  функцій необхідна при рішенні численних інженерних і наукових задач моделювання.  **Передбачення/прогноз.** Нехай задані n дискретних відліків {y(t1), y(t2), ..., y(tn)} у  послідовні моменти часу t1, t2,..., tn . Завдання полягає в передбаченні значення y(tn+1) у  деякий майбутній момент часу tn+1. Передбачення/прогноз мають значний вплив  на прийняття рішень у бізнесі, науці й техніці (передбачення цін на фондовій біржі, прогноз погоди).  **Оптимізація.** Численні проблеми в математиці, статистиці, техніці, науці, медицині й економіці можуть розглядатися як проблеми оптимізації. Задачею алгоритму оптимізації є знаходження  такого рішення, що задовольняє системі обмежень і максимізує чи мінімізує цільову функцію.  **Пам'ять, що адресується за змістом.** В традиційних комп'ютерах звертання до пам'яті  доступно тільки за допомогою адреси, що не залежить від змісту пам'яті. Більш того,  якщо допущена помилка в обчисленні адреси, то може бути знайдена зовсім інша інформація.  Асоціативна пам'ять, чи пам'ять, що адресується за змістом, доступна за вказівкою заданого змісту. Вміст пам'яті може бути викликано навіть по частковому входу чи спотвореному змісту.  Асоціативна пам'ять надзвичайно бажана при створенні мультимедійних інформаційних баз даних.  **Керування.** Розглянемо динамічну систему, задану сукупністю {u(t), y(t)}, де u(t) є вхідним  керуючим впливом, а y(t) - виходом системи в момент часу t. В системах керування з  еталонною моделлю метою керування є розрахунок такого вхідного впливу u(t), при якому  система діє по бажаній траєкторії, заданою еталонною моделлю. Прикладом є оптимальне  керування двигуном.  Але, незважаючи на переваги нейронних мереж в часткових галузях над традиційними  обчисленнями, існуючі нейромережі є не досконалими рішеннями. Вони навчаються і можуть робити "помилки". Окрім того, не можна гарантувати, що розроблена мережа є оптимальною  мережею. Застосування нейромереж вимагає від розробника виконання ряду умов.  Ці умови включають:   * множину даних, що включає інформацію, яка може характеризувати проблему; * відповідно встановлену за розміром множину даних для навчання й тестування мережі; * розуміння базової природи проблеми, яка буде вирішена; * вибір функції суматора, передатної функції та методів навчання; * розуміння інструментальних засобів розробника; * відповідна потужність обробки.   Новий шлях обчислень вимагає вмінь розробника поза межами традиційних обчислень.  Спочатку, обчислення були лише апаратними й інженери робили його працюючим. Потім,  були спеціалісти з програмного забезпечення: програмісти, системні інженери, спеціалісти  по базах даних та проектувальники. Тепер є нейронні архітектори. Новий професіонал  повинен мати кваліфікацію, відмінну від його попередників. Наприклад,  він повинен знати статистику для вибору і оцінювання навчальних і тестувальних  множин. Логічне мислення сучасних інженерів програмного забезпечення, їх емпіричне вміння  та інтуїтивне відчуття гарантує створення ефективних нейромереж. | |

**Тема 6. Детальний опис компонентів та роботи нейронних мереж**

* [Розширена модель штучного нейрона](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_1)
* [Компоненти штучного нейрона](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_2)
  + [Вагові коефіцієнти](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_2_1)
  + [Функція суматора](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_2_2)
  + [Передатна функція](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_2_3)
  + [Масштабування](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_2_4)
  + [Вихідна функція (змагання)](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_2_5)
  + [Функція похибки та поширюване назад значення](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_2_6)
  + [Функція навчання](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_2_7)
* [Архітектура з'єднань штучних нейронів](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_3)
* [Навчання штучної нейронної мережі](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_4)
  + [Контрольоване навчання](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_4_1)
  + [Неконтрольоване навчання](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_4_2)
  + [Оцінки навчання](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_4_3)
* [Правила навчання](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_5)
  + [Правило Хеба](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_5_1)
  + [Правило Хопфілда](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_5_2)
  + [Правило "дельта"](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_5_3)
  + [Правило градієнтного спуску](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_5_4)
  + [Навчання методом змагання](http://www.victoria.lviv.ua/html/oio/html/theme6.htm#6_5_5)

Сучасні дослідження фізіології мозку відкривають лише обмежене розуміння роботи нейронів і процесу мислення. Дослідники працюють як в біологічній, так і в інженерній галузях для розшифрування ключових механізмів нейронної обробки, що допомагає створювати потужніші і більш стислі нейромережі.

**Розширена модель штучного нейрона**

Поглиблені уявлення щодо будови біологічного дозволяють представити модель технічного нейрона в розширеному вигляді, деталізована структура якого наведена на рис. 1, де

1. суматор, який моделює функції тіла біологічного нейрона;
2. функціональний перетворювач, що виконує роль аксонного горбка;
3. збуджуючий синапс;
4. гальмуючий синапс;
5. вхідний сигнал;
6. дихотомічне галуження вхідного сигналу;
7. вихідний сигнал;
8. дихотомічне галуження вихідного сигналу;
9. прямий зв'язок, що відповідає аксодендритному зв'язку між біологічними нейронами;
10. зворотній (аксосоматичний) зв'язок.

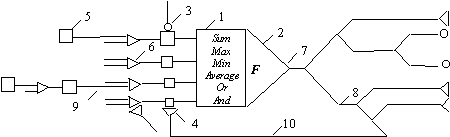


Рис. 1. Розширена модель нейронного елемента

Підставою для деталізації моделі нейронного елемента можна вважати встановлення нових фактів в області нейрофізіології, зокрема:

1. Наявність декількох місць синаптичного контакту.
2. Дихотомічне галуження дендритів різних порядків, що відповідає логічним операціям "І", "АБО", "Виключаюче АБО", виділення максимального або мінімального сигналу в технічних аналогах.
3. Різні діаметри стовбурових дендритів, гілок, що безпосередньо прилягають до тіла нейрона, причому величина діаметра визначає степінь важливості інформації, яка проходить через дендрит.
4. Наявність "доріжок" на поверхні соми, які проходять від головних стовбурних дендритів до аксона, що обумовлює наявність паралельних шляхів обробки інформації і надає можливість застосування логічних операцій над сигналами, які надходять від різних стовбурових дендритів.
5. Особливості функціонування аксонного горбка; власне аксонний горбок встановлює передатну функцію нейрона, яка має набагато складнішу форму від прийнятих в нейромережевих технологіях сигмоїдальних або лінійних передатних функцій.
6. Наявність дихотомічного галуження аксона; у вузлах галуження відбувається керування проходженням сигналу, що залежить від співвідношення діаметрів різних гілок аксона; при математичному моделюванні ці особливості можна реалізувати за допомогою логічних операцій.
7. Наявність зворотного аксосоматичного зв'язку, що вже знайшло свою реалізацію при побудові рекурентних нейромереж.

Поглиблені знання відносно будови біологічного нейрона, як ефективного перетворюючого інструмента, можна розглядати як джерело базових ідей та концепцій по створенню нових парадигм нейромереж не лише на даний час, але і на віддалену перспективу.

**Компоненти штучного нейрона**

Незалежно від розташування та функціонального призначення, всі штучні нейронні елементи мають спільні компоненти. Розглянемо сім основних компонент штучного нейрона.

**Компонента 1. Вагові коефіцієнти**

При функціонуванні нейрон одночасно отримує багато вхідних сигналів. Кожен вхід має свою власну синаптичну вагу, яка надає входу вплив, необхідний для функції суматора елемента обробки. Ваги є мірою сили вхідних зв'язків і моделюють різноманітні синаптичні сили біологічних нейронів. Ваги суттєвого входу підсилюються і, навпаки, вага несуттєвого входу примусово зменшується, що визначає інтенсивність вхідного сигналу. Ваги можуть змінюватись відповідно до навчальних прикладів, топології мережі та навчальних правил.

**Компонента 2. Функція суматора**

Першим кроком дії нейрону є обчислення зваженої суми всіх входів. Математично, вхідні сигнали та відповідні їм ваги представлені векторами (х10, х20 ... хn0) та (w10, w20 . . . wn0). Добуток цих векторів є загальним вхідним сигналом. Спрощеною функцією суматора є множення кожної компоненти вектора х на відповідну компоненту вектора w: вхід1 = х10 \* w10, вхід2 = х20 \* w20, і знаходження суми всіх добутків: вхід1 + вхід2 + . . . + вхідn. Результатом є єдине число, а не багатоелементний вектор.

Функція суматора може бути складнішою, наприклад, вибір мінімуму, максимуму, середнього арифметичного, добутку або виконувати інший нормалізуючий алгоритм. Вхідні сигнали та вагові коефіцієнти можуть комбінуватись багатьма способами перед надходженням до передатної функції. Особливі алгоритми для комбінування входів нейронів визначаються обраними мережною архітектурою та парадигмою.

В деяких нейромережах функції суматора виконують додаткову обробку, так звану функцію активації, яка зміщує вихід функції суматора відносно часу. На жаль, функції активації на теперішній час обмежено досліджені і більшість сучасних нейронних реалізацій використовують функцію активації "тотожності", яка еквівалентна її відсутності. Цю функцію доцільніше використовувати як компоненту мережі в цілому, ніж як компоненту окремого нейрона.

**Компонента 3. Передатна функція**

Результат функції суматора є зваженою сумою вхідних сигналів, що перетворюється у вихідний сигнал через алгоритмічний процес відомий як передатна функція. У передатній функції для визначення виходу нейрона загальна сума порівнюється з деяким порогом. Якщо сума є більшою за значення порога, елемент обробки генерує сигнал, в противному випадку сигнал не генерується або генерується гальмуючий сигнал.

Переважно застосовують нелінійну передатну функцію, оскільки лінійні (прямолінійні) функції обмежені і вихід є просто пропорційним до входу. Застосування лінійних передатних функцій було проблемою у ранніх моделях мереж, і їх обмеженість та недоцільність була доведена в книзі Мінскі та Пейперта "Перцептрони".  
На рис. 2 зображені типові передатні функції.

|  |  |
| --- | --- |
| http://www.victoria.lviv.ua/html/oio/images/theme6/ris2.gif | http://www.victoria.lviv.ua/html/oio/images/theme6/ris3.gif |
| http://www.victoria.lviv.ua/html/oio/images/theme6/ris4.gif | http://www.victoria.lviv.ua/html/oio/images/theme6/ris5.gif |

Для простої передатної функції нейромережа може видавати 0 та 1, 1 та -1 або інші числові комбінації. Передатна функція в таких випадках є "жорстким обмежувачем" або пороговою функцією (рис. 2а).

Інший тип передатної функції лінійна з насиченням віддзеркалює вхід всередині заданого діапазону і діє як жорсткий обмежувач за межами цього діапазону. Це лінійна функція, яка відсікається до мінімальних та максимальних значень, роблячи її нелінійною (рис. 2б).

Наступним вибором є сигмоїда або S-подібна крива, яка наближує мінімальне та максимальне значення у асимптотах і називається сигмоїдою (рис. 2в), коли її діапазон [0, 1], або гіперболічним тангенсом (рис. 2г), при діапазоні [-1, 1]. Важливою рисою цих кривих є неперервність функцій та їх похідних. Застосування сигмоїдних функцій надає добрі результати і має широке застосування.

Зрештою, для різних нейромереж можуть вибиратись інші передатні функції.

Перед надходженням до передатної функції до вхідного сигналу деколи додають однорідно розподілений випадковий шум, джерело та кількість якого визначається режимом навчання. В літературі цей шум, згадується як "температура" штучних нейронів, яка надає математичній моделі елемент реальності.

**Компонента 4. Масштабування**

Після передатної функції вихідний сигнал проходить додаткову обробку масштабування, тобто результат передатної функції множиться на масштабуючий коефіцієнт і додається зміщення.

**Компонента 5. Вихідна функція (змагання)**

По аналогії з біологічним нейроном, кожний штучний нейрон має один вихідний сигнал, який передається до сотень інших нейронів. Переважно, вихід прямо пропорційний результату передатної функції. В деяких мережевих архітектурах результати передатної функції змінюються для створення змагання між сусідніми нейронами. Нейронам дозволяється змагатися між собою, блокуючи дії нейронів, що мають слабий сигнал. Змагання (конкуренція) може відбуватись між нейронами, які знаходяться на одному або різних прошарках. По-перше, конкуренція визначає, який штучний нейрон буде активним і забезпечить вихідний сигнал. По-друге, конкуруючі виходи допомагають визначити, який нейрон візьме участь у процесі навчання.

**Компонента 6. Функція похибки та поширюване назад значення**

У більшості мереж, що застосовують контрольоване навчання обчислюється різниця між спродукованим та бажаним виходом. Похибка відхилення (біжуча похибка) перетворюється функцією похибки відповідно заданій мережній архітектурі. В базових архітектурах похибка відхилення використовується безпосередньо, в деяких парадигмах використовується квадрат або куб похибки зі збереженням знаку.

Після проходження всіх прошарків біжуча похибка поширюється назад до попереднього прошарку і може бути безпосередньо похибкою або похибкою, масштабованою певним чином залежно від типу мережі (наприклад, похідною від передаточної функції). Це поширюване назад значення враховується в наступному циклі навчання.

**Компонента 7. Функція навчання**

Метою функції навчання є налаштування змінних ваг з'єднань на входах кожного елемента обробки відповідно до певного алгоритму навчання для досягнення бажаного результату. Існує два типи навчання: контрольоване та неконтрольоване. Контрольоване навчання вимагає навчальної множини даних або спостерігача, що ранжує ефективність результатів мережі. У випадку неконтрольованого навчання система самоорганізовується за внутрішнім критерієм, закладеним в алгоритм навчання.

**Архітектура з'єднань штучних нейронів**

Об'єднуючись у мережі, нейрони утворюють системи обробки інформації, які забезпечують ефективну адаптацію моделі до постійних змін з боку зовнішнього середовища. В процесі функціонування мережі відбувається перетворення вхідного вектора сигналів у вихідний. Конкретний вид перетворення визначається як архітектурою нейромережі так і характеристиками нейронних елементів, засобами керування та синхронізації інформаційних потоків між нейронами. Важливим фактором ефективності мережі є встановлення оптимальної кількості нейронів та типів зв'язків між ними.

При описі нейромереж використовують кілька усталених термінів, які в різних джерелах можуть мати різне трактування, зокрема:

* структура нейромережі - спосіб зв'язків нейронів у нейромережі;
* архітектура нейромережі - структура нейромережі та типи нейронів;
* парадигма нейромережі - спосіб навчання та використання; іноді вміщує і поняття архітектури.

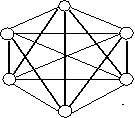
На основі однієї архітектури можуть бути реалізовані різні парадигми нейромережі і навпаки.

Серед відомих архітектурних рішень виділяють групу слабозв'язаних нейронних мереж, у випадку, коли кожний нейрон мережі зв'язаний лише із сусідніми.

|  |  |
| --- | --- |
| http://www.victoria.lviv.ua/html/oio/images/theme6/ris6.gif | http://www.victoria.lviv.ua/html/oio/images/theme6/ris7.gif |

Рис. 3. Слабозв'язані нейромережі

Навпаки, якщо входи кожного нейрона зв'язані з виходами усіх решта нейронів, тоді мова йде про повнозв'язані нейромережі.



Зрозуміло, що такий поділ носить дещо теоретичний характер. Аналізуючи найбільш відомі на даний час розробки нейромереж, слід зазначити, що самим поширеним варіантом архітектури є багатошарові мережі. Нейрони в даному випадку об'єднуються у прошарки з єдиним вектором сигналів входів. Зовнішній вхідний вектор подається на вхідний прошарок нейронної мережі (рецептори). Виходами нейронної мережі є вихідні сигнали останнього прошарку (ефектори). Окрім вхідного та вихідного прошарків, нейромережа має один або декілька прихованих прошарків нейронів, які не мають контактів із зовнішнім середовищем.

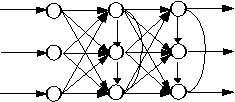


Рис. 5. Багатошаровий тип з'єднання нейронів

* Зв'язки між нейронами різних прошарків називають проективними.
* Зв'язки скеровані від вхідних прошарків до вихідних називаються аферентними,
* в інакшому випадку, при зворотному напрямку зв'язків - еферентними.
* Зв'язки між нейронами одного прошарку відносять до бічних (латеральних).

Фактично, по архітектурі зв'язків, більшість відомих нейромереж, що знайшли практичне застосування, можна згрупувати у два великих класи:

1. Мережі прямого поширення (з односкерованими послідовними зв'язками).
2. Мережі зворотного поширення (з рекурентними зв'язками).

На схемі (рис. 6) представлені назви найбільш типових архітектур мереж, що в свою чергу мають багато модифікацій та можуть бути складниками у інших мережах.

Мережі прямого поширення відносять до статичних, так як на задані входи нейронів надходить не залежний від попереднього стану мережі вектор вхідних сигналів.

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

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

Рис. 6. Найбільш відомі архітектури нейронних мереж

Оригінальність нейромереж, як аналога біологічного мозку, полягає у здібності до навчання за прикладами, що складають навчальну множину. Процес навчання нейромереж розглядається як налаштування архітектури та вагових коефіцієнтів синаптичних зв'язків відповідно до даних навчальної множини так, щоб ефективно вирішити поставлену задачу. Виділяють варіанти контрольованого та неконтрольованого навчання.

**Навчання штучної нейронної мережі**

**Контрольоване навчання**

Величезна більшість рішень отримана від нейромереж з контрольованим навчанням, де біжучий вихід постійно порівнюється з бажаним виходом. Ваги на початку встановлюються випадково, але під час наступних ітерацій коректуються для досягнення близької відповідності між бажаним та біжучим виходом. Створені методи навчання націлені на мінімізації біжучих похибок всіх елементів обробки, яке створюється за якийсь час неперервною зміною синаптичних ваг до досягнення прийнятної точності мережі.

Перед використанням, нейромережа з контрольованим навчанням повинна бути навченою. Фаза навчання може тривати багато часу, зокрема, у прототипах систем, з невідповідною процесорною потужністю навчання може займати декілька годин. Навчання вважається закінченим при досягненні нейромережею визначеного користувачем рівня ефективності. Цей рівень означає, що мережа досягла бажаної статистичної точності, оскільки вона видає бажані виходи для заданої послідовності входів. Після навчання ваги з'єднань фіксуються для подальшого застосування. Деякі типи мереж дозволяють під час використання неперервне навчання, з набагато повільнішою оцінкою навчання, що допомагає мережі адаптуватись умов, що повільно змінюються.

Навчальні множини повинні бути досить великими, щоб містити всю необхідну інформацію для виявлення важливих особливостей і зв'язків. Але і навчальні приклади повинні містити широке різноманіття даних. Якщо мережа навчається лише для одного прикладу, ваги старанно встановлені для цього прикладу, радикально змінюються у навчанні для наступного прикладу. Попередні приклади при навчанні наступних просто забуваються. В результаті система повинна навчатись всьому разом, знаходячи найкращі вагові коефіцієнти для загальної множини прикладів. Наприклад, у навчанні системи розпізнавання піксельних образів для десяти цифр, які представлені двадцятьма прикладами кожної цифри, всі приклади цифри "сім" не доцільно представляти послідовно. Краще надати мережі спочатку один тип представлення всіх цифр, потім другий тип і так далі.

Головною компонентою для успішної роботи мережі є представлення і кодування вхідних і вихідних даних. Штучні мережі працюють лише з числовими вхідними даними, отже, необроблені дані, що надходять із зовнішнього середовища повинні перетворюватись. Додатково необхідне масштабування, тобто нормалізація даних відповідно до діапазону всіх значень. Нормалізація виконується шляхом ділення кожної компоненти вхідного вектора на довжину вектора, що перетворює вхідний вектор в одиничний. Попередня обробка зовнішніх даних, отриманих за допомогою сенсорів, у машинний формат спільна для стандартних комп'ютерів і є легко доступною.

Якщо після контрольованого навчання нейромережа ефективно опрацьовує дані навчальної множини, важливим стає її ефективність при роботі з даними, які не використовувались для навчання. У випадку отримання незадовільних результатів для тестової множини, навчання продовжується. Тестування використовується для забезпечення запам'ятовування не лише даних заданої навчальної множини, але і створення загальних образів, що можуть міститись в даних.

**Неконтрольоване навчання**

Неконтрольоване навчання може бути великим надбанням у майбутньому. Воно проголошує, що комп'ютери можуть самонавчатись у справжньому роботизованому сенсі. На даний час, неконтрольоване навчання використовується мережах відомих, як самоорганізовані карти (self organizing maps), що знаходяться в досить обмеженому користуванні, але доводячи перспективність самоконтрольованого навчання. Мережі не використовують зовнішніх впливів для коректування своїх ваг і внутрішньо контролюють свою ефективність, шукаючи регулярність або тенденції у вхідних сигналах та роблять адаптацію згідно навчальної функції. Навіть без повідомлення правильності чи неправильності дій, мережа повинна мати інформацію відносно власної організації, яка закладена у топологію мережі та навчальні правила.

Алгоритм неконтрольованого навчання скерований на знаходження близькості між групами нейронів, які працюють разом. Якщо зовнішній сигнал активує будь-який вузол в групі нейронів, дія всієї групи в цілому збільшується. Аналогічно, якщо зовнішній сигнал в групі зменшується, це приводить до гальмуючого ефекту на всю групу.

Конкуренція між нейронами формує основу для навчання. Навчання конкуруючих нейронів підсилює відгуки певних груп на певні сигнали. Це пов'язує групи між собою та відгуком. При конкуренції змінюються ваги лише нейрона-переможця.

**Оцінки навчання**

Оцінка ефективності навчання нейромережі залежить від декількох керованих факторів. Теорія навчання розглядає три фундаментальні властивості, пов'язані з навчанням: ємність, складність зразків і обчислювальна складність. Під ємністю розуміють, скільки зразків може запам'ятати мережа, і які межі прийняття рішень можуть бути на ній сформовані. Складність зразків визначає число навчальних прикладів, необхідних для досягнення здатності мережі до узагальнення. Обчислювальна складність напряму пов'язана з потужністю процесора ЕОМ.

**Правила навчання**

У загальному використанні є багато правил навчання, але більшість з цих правил є деякою зміною відомого та найстаршого правила навчання, правила Хеба. Дослідження різних правил навчання триває, і нові ідеї регулярно публікуються в наукових та комерційних виданнях. Представимо декілька основних правил навчання.

**Правило Хеба**

Опис правила з'явився у його книзі "Організація поведінки" у 1949 р. "Якщо нейрон отримує вхідний сигнал від іншого нейрону і обидва є високо активними (математично мають такий самий знак), вага між нейронами повинна бути підсилена". При збудженні одночасно двох нейронів з виходами (хj, уі) на t-тому кроці навчання вага синаптичного з'єднання між ними зростає, в інакшому випадку - зменшується, тобто

 *Wij*(*k*)=*r xj* (*k*) *yi* (*k*),

де r - коефіцієнт швидкості навчання.

Може застосовуватись при навчанні "з вчителем" і "без вчителя".

**Правило Хопфілда**

Є подібним до правила Хеба за винятком того, що воно визначає величину підсилення або послаблення. "Якщо одночасно вихідний та вхідний сигнал нейрона є активними або неактивними, збільшуємо вагу з'єднання оцінкою навчання, інакше зменшуємо вагу оцінкою навчання".

**Правило "дельта"**

Це правило є подальшою зміною правила Хеба і є одним із найбільш загально використовуваних. Це правило базується на простій ідеї неперервної зміни синаптичних ваг для зменшення різниці ("дельта") між значенням бажаного та біжучого вихідного сигналу нейрона.

*Wij*= *xj* (*di* - *yi*).

За цим правилом мінімізується середньоквадратична похибка мережі. Це правило також згадується як правило навчання Відрова-Хофа та правило навчання найменших середніх квадратів.

У правилі "дельта" похибка отримана у вихідному прошарку перетворюється похідною передатної функції і послідовно пошарово поширюється назад на попередні прошарки для корекції синаптичних ваг. Процес зворотного поширення похибок мережі триває до досягнення першого прошарку. Від цього методу обчислення похибки успадкувала своє ім'я відома парадигма FeedForward BackPropagation.

При використанні правила "дельта" важливим є невпорядкованість множини вхідних даних. При добре впорядкованому або структурованому представленні навчальної множини результат мережі може не збігтися до бажаної точності і мережа буде вважатись нездатною до навчання.

**Правило градієнтного спуску**

Це правило подібне до правила "дельта" використанням похідної від передатної функції для змінювання похибки "дельта" перед тим, як застосувати її до ваг з'єднань. До кінцевого коефіцієнта зміни, що діє на вагу, додається пропорційна константа, яка пов'язана з оцінкою навчання. І хоча процес навчання збігається до точки стабільності дуже повільно, це правило поширене і є загально використовуване.

Доведено, що різні оцінки навчання для різних прошарків мережі допомагає процесу навчання збігатись швидше. Оцінки навчання для прошарків, близьких до виходу, встановлюються меншими, ніж для рівнів, ближчих до входу.

**Навчання методом змагання**

На відміну від навчання Хеба, у якому множина вихідних нейронів може збуджуватись одночасно, при навчанні методом змагання вихідні нейрони змагаються між собою за активізацію. Це явище, відоме як правило "переможець отримує все". Подібне навчання має місце в біологічних нейронних мережах. Навчання за допомогою змагання дозволяє кластеризувати вхідні дані: подібні приклади групуються мережею відповідно до кореляцій і представляються одним елементом.

При навчанні модифікуються синаптичні ваги нейрона-переможця. Ефект цього правила досягається за рахунок такої зміни збереженого в мережі зразка (вектора синаптичних ваг нейрона-переможця), при якому він стає подібним до вхідного приклада. Нейрон з найбільшим вихідним сигналом оголошується переможцем і має можливість гальмувати своїх конкурентів і збуджувати сусідів. Використовується вихідний сигнал нейрона-переможця і тільки йому та його сусідам дозволяється коректувати свої ваги з'єднань.

*Wij* (*k*+1)= *Wij*(*k*)+*r* [*xj* - *Wij*(*k*)].

Розмір області сусідства може змінюватись під час періоду навчання. Звичайна парадигма повинна починатись з великої області визначення сусідства і зменшуватись під час процесу навчання. Оскільки елемент-переможець визначається по найвищій відповідності до вхідного зразку, мережі Коxонена моделюють розподіл входів. Це правило використовується в самоорганізованих картах.

**Тема 7. Класифікація відомих нейромереж по основних категоріях застосування**

* [Перцептрон Розенбалата](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#Perceptron)
* [Нейромережа зворотного поширення похибки](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#BackPropagation)
* [Delta Bar Delta](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#Delta_Bar_Delta)
* [Extended Delta Bar Delta](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#Extended_Delta_Bar_Delta)
* [Скерований випадковий пошук](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#Skerovanyj_Poschuk)
* [Нейронна мережа вищого порядку або функціонально-пов'язана нейронна мережа](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#Vyshij_Porjadok)
* [Мережа Кохонена](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#Cohonen)
* [Квантування навчального вектора](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#LearningVectorQuantization)
* [Мережа зустрічного поширення CounterРropagation](http://www.victoria.lviv.ua/html/oio/html/theme7_1.htm#Counter%D0%A0ropagation)
* [Імовірнісна нейронна мережа](http://www.victoria.lviv.ua/html/oio/html/theme7_1.htm#ProbabilisticNeuralNetworks)
* [Мережа Хопфілда](http://www.victoria.lviv.ua/html/oio/html/theme7_1.htm#Hopfild)
* [Машина Больцмана](http://www.victoria.lviv.ua/html/oio/html/theme7_1.htm#BoltzmannMashine)
* [Мережа Хемінга](http://www.victoria.lviv.ua/html/oio/html/theme7_1.htm#Hamming)
* [Двоскерована асоціативна пам'ять](http://www.victoria.lviv.ua/html/oio/html/theme7_1.htm#AsociatyveMemory)
* [Мережа адаптивної резонансної теорії](http://www.victoria.lviv.ua/html/oio/html/theme7_1.htm#AdaptiveResonanceTheory)

Оскільки всі штучні нейронні мережі базуються на концепції нейронів, з'єднань та передатних функцій, існує подібність між різними структурами або архітектурами нейронних мереж. Більшість змін походить з різних правил навчання. В цій лекції розглянемо деякі з найвідоміших штучних нейромереж, які організовані в певні категорії застосувань.

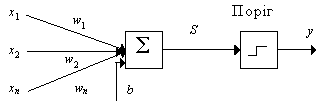
**Перцептрон Розенбалата**

Першою моделлю нейромереж вважають перцептрон Розенбалата. 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 елементів вхідного сигналу, 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 класу A,
* -1 - вхідний сигнал належить класу B.

На рис. 1 показана схема нейронів, використовуваних в одношарових перцептронах, графік передатної функції і схема вирішальних областей, створених у багатовимірному просторі вхідних сигналів. Вирішальні області визначають, які вхідні образи будуть віднесені до класу A, які - до класу B. Перцептрон, що складається з одного нейрона, формує дві вирішальні області, розділені гіперплощиною. На рисунку показаний випадок, коли розмірність вихідного сигналу дорівнює 2. При цьому поділяюча поверхня уявляє собою пряму лінію на площині. Рівняння, що задає поділяючу пряму, залежить від значень синаптичних ваг і зсуву.



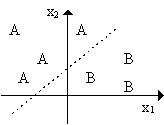
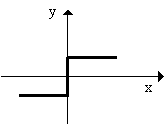


Рис. 1. Схема нейрона, графік передатної функції і поділяюча поверхня

**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,..., *xn*) пред'являється нейрону разом з бажаним вихідним сигналом *d*.
3. Обчислення вихідного сигналу нейрона:

http://www.victoria.lviv.ua/html/oio/images/theme7/image1379.gif

1. Налаштування значень ваг:

*wi*(*t*+1)=*wi* (*t*)+*r*[*d*(*t*)-*y*(*t*)]*xi* (*t*), *i*=1, ..., *N*

http://www.victoria.lviv.ua/html/oio/images/theme7/image1380.gif

де *wі*(*t*) - вага зв'язку від *і*-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.

1. 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 дуже прості. Простий і швидкий алгоритм навчання.

***Модифікації.*** Багатошарові перцептрони дають можливість будувати більш складні поділяючі поверхні і тому більш поширені.

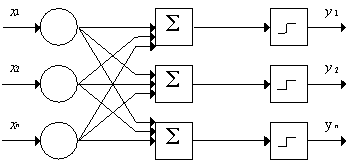


Рис. 2. Перцептрон із багатьма виходами

**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 *ВackРropagation* найбільш популярна, ефективна та легка модель навчання для складних, багатошарових мереж. Вона використовується у різних типах застосувань і породила великий клас нейромереж з різними структурами та методами навчання.

Типова мережа *ВackРropagation* має вхідний прошарок, вихідний прошарок та принаймні один прихований прошарок. Теоретично, обмежень відносно числа прихованих прошарків не існує, але практично застосовують один або два (рис. 3).

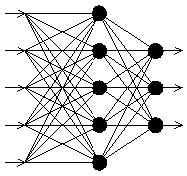


Рис. 3. Багатошаровий перцептрон

Нейрони організовані в пошарову структуру з прямою передачею 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. Визначення числа проміжних прошарків і числа нейронів в них є важливим при моделюванні мережі. Більшість дослідників та інженерів, застосовуючи архітектуру до визначених проблем використовують загальні правила, зокрема:

1. 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 достатня кількість нейронів становить *n*прих  *n*вх + *n*вих, де *n*вх, *n*вих - кількість нейронів у вхідному і, відповідно, у вихідному прошарках.
2. Якщо складність у відношенні між отриманими та бажаними даними на виході збільшується, кількість нейронів прихованого прошарку повинна також збільшитись.
3. 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 тут 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 виявлений, 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 ітеративне, 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 останову).

**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... *xN*), повинні розрізнятися для всіх прикладів навчальної множини.
3. Обчислення вихідного сигналу:

http://www.victoria.lviv.ua/html/oio/images/theme7/image1383.gif

*yim*= *f*(*Sjm*)

*im*=1, 2, ..., *Nm*, *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.

1. Налаштування синаптичних ваг:

*wij*(*t*+1)=*wij*(*t*)+*rgjx'і*

де *wij* - вага від нейрона *i* або від елемента вхідного сигналу *i* до нейрона *j* у момент часу *t*, *xi*' - вихід нейрона *i*, *r* - швидкість навчання, *gj* - значення похибки для нейрона *j*.

Якщо нейрон з номером *j* належить останньому прошарку, тоді

*gj*=*yj*(1-*yj*)(*dj*-*yj*)

де *dj* - бажаний вихід нейрона *j*, *yj* - поточний вихід нейрона *j*.

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

http://www.victoria.lviv.ua/html/oio/images/theme7/image1384.gif

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. В нейронних мережах застосовуються кілька варіантів сигмоїдальних передатних функцій.

*Функція Ферми* (експонентна сигмоїда):

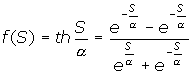
http://www.victoria.lviv.ua/html/oio/images/theme7/image1385.gif

де s - вихід суматора нейрона,  - деякий параметр.

*Раціональна сигмоїда:*

http://www.victoria.lviv.ua/html/oio/images/theme7/image1386.gif

*Гіперболічний тангенс:*



Згадані функції відносяться до однопараметричних. Значення функції залежить від аргументу й одного параметра. Також використовуються багатопараметричні передатні функції, наприклад:

http://www.victoria.lviv.ua/html/oio/images/theme7/image1388.gif

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.*** Багатокритеріальна задача оптимізації в методі зворотного поширення розглядається як набір однокритеріальних задач - на кожній ітерації відбуваються зміни значень параметрів мережі, що покращують роботу лише з одним прикладом навчальної вибірки. Такий підхід істотно зменшує швидкість навчання.

***Переваги.*** Зворотне поширення - ефективний та популярний алгоритм навчання багатошарових нейронних мереж, з його допомогою вирішуються численні практичні задачі.

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

**Delta Bar Delta**

*Delta bar Delta* була розроблена Робертом Джекобсом (*Robert Jacobs*), для покращення оцінки навчання стандартних мереж *FeedForward* і є модифікацією мережі *BackPropagation*.

Процедура *BackPropagation* базується на підході найкрутішого спуску, який мінімізує похибку передбачення мережі під час процесу змінювання синаптичних ваг. Стандартні оцінки навчання застосовуються на базисі "прошарок за прошарком" і значення моменту призначаються глобально. Моментом вважається фактор, що використовується для згладжування оцінки навчання. Момент додається до стандартної зміни ваги і є пропорційним до попередньої зміни ваги. Хоча цей метод є успішним у вирішенні багатьох задач, збіжність процедури занадто повільна для використання.

Парадигма *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 змінюванні 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 до одної ваги з'єднання не застосовується для всіх ваг у прошарку.

Оцінка навчання ваги з'єднання змінюється на основі інформації про біжучу похибку, знайденої із стандартної *ВackРropagation*. 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 змінюватись в часі. Призначення оцінки навчання до кожного з'єднання, дозвіл цій оцінці навчання неперервне змінюватись з часом обумовлюють зменшення часу збіжності.

З дозволом різних оцінок навчання для кожної ваги з'єднання у мережі, пошук найкрутішого спуску може не виконуватись. Замість цього, ваги з'єднань змінюються на основі часткових похідних похибки відносно самої ваги і оцінці "кривини поверхні похибки" поблизу біжучої точки значення ваги. 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 виявляється не досить швидким.

**Extended Delta Bar Delta**

Елі Мінаї (*Ali Minai*) та Рон Вільямс (*Ron Williams*) розробили алгоритм *Extended Delta bar Delta*, як природній наслідок роботи Джекобса. В алгоритм вбудовується пам'ять з особливістю відновлення. Після кожного представлення навчальних даних епохи, оцінюється накопичена похибка. Якщо похибка є меншою за попередню мінімальну похибку, ваги зберігаються у пам'яті, як найкращі на цей час. Параметр допуску керує фазою відновлення. У випадку, коли біжуча похибка перевищує мінімальну попередню похибку, модифіковану параметром допуску, всі значення ваг з'єднань стохастичне повертаються до збереженої у пам'яті найкращої множини ваг.

**Скерований випадковий пошук**

Мережа скерованого випадкового пошуку (*Directed Random Search*), використовує стандартну архітектуру *FeedForward*, яка не базується на алгоритмі *BackProragation* і коректує ваги випадковим чином. Для забезпечення порядку в такому процесі, до випадкового кроку додається компонента напрямку, яка гарантує скерування ваг до попередньо успішного напрямку пошуку. Вплив на всі нейрони здійснюється окремо.

Для збереження ваг всередині компактної області, де алгоритм працює добре, встановлюють верхню межу величини ваги. Встановлюючи межі ваг великими, мережа може продовжувати працювати, оскільки справжній глобальний оптимум лишається невідомим. Іншою особливістю правила навчання є початкова відмінність у випадковому розподілі ваг. У більшості комерційних пакетів існує рекомендоване розробником число для параметра початкової відмінності.

Парадигма випадкового пошуку має декілька важливих рис. Вона є швидкою та легкою у використанні, найкращі результати отримують, 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 надає ефект лише в малій мережі, оскільки при збільшенні числа з'єднань, процес навчання стає довгим та важким.

Існує чотири ключові компоненти мережі з випадковим пошуком. Це є випадковий крок, крок реверсування, скерована компонента та самокорегуюча відмінність.

***Випадковий крок.*** До кожної ваги додається випадкова величина. Вся навчальна множина пропускається через мережу, створюючи "похибку передбачення". Якщо нова загальна похибка навчальної множини є меншою за попередню найкращу похибку передбачення, біжучі значення ваг, які включають випадковий крок стають новою множиною "найкращих" ваг. Біжуча похибка передбачення зберігається як нова, найкраща похибка передбачення.

***Крок реверсування.*** Якщо результати випадкового кроку є гіршими за попередні найкращі, випадкова величина віднімається від початкового значення ваги. Це створює множину ваг, які знаходяться в протилежному напрямку до попереднього випадкового кроку. Якщо загальна "похибка передбачення" є меншою за попередню найкращу похибку, біжучі значення ваг та біжуча похибка передбачення зберігаються як найкращі. Якщо і прямий і зворотній кроки не покращують результат, до найкращих ваг додається повністю нова множина випадкових значень і процес починається спочатку.

***Скерована компонента.*** Для збіжності мережі створюється множина скерованих компонент, отриманих по результатах прямого та зворотного кроків. Скеровані компоненти, що відображають ланку успіхів або невдач попередніх випадкових кроків, додаються до випадкових компонент на кожному кроці процедури і забезпечують елемент "здорового глузду" до пошуку. Доведено, що додавання скерованих компонент забезпечує різке підвищення ефективності алгоритму.

***Самокорегуюча відмінність.*** Визначається параметр початкової відмінності для керування початковим розміром випадкових кроків, що додається до ваг. Адаптивний механізм змінює параметр відмінності, який базується на біжучій оцінці успіху або невдачі. Правило навчання припускає, що біжучий розмір кроків для ваг у правильному напрямку збільшується для випадку декількох послідовних успіхів. Навпаки, якщо воно виявляє декілька послідовних невдач, відмінність зменшується для зменшення розміру кроку.

***Переваги.***Для малих та середніх нейромереж, скерований випадковий пошук надає добрі результати за короткий час. Навчання є автоматичним, вимагає невеликої взаємодії з користувачем.

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

**Нейронна мережа вищого порядку або функціонально-пов'язана нейронна мережа**

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

Існує два основних способи додавання вхідних вузлів. В першому, у модель можуть додаватись перехресні добутки вхідних елементів, це називається вхідним добутком або тензорною моделлю, де кожна компонента вхідного образу перемножується зі всіма компонентами вхідного вектора. Наприклад, для мережі *ВackРropagation* з трьома входами (*A*, *B* і *C*), перехресними добутками будуть *AA, BB, CC, AB, AC* та *BC* (елементи другого порядку). Також можуть додаватись 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 у модель нейронної мережі вищого порядку зі входами: *A, B, C, SIN*(*A*), *COS*(*B*), *LOG*(*C*), *MAX*(*A,B,C*) та ін. Повний ефект повинна забезпечити мережа з об'єднанням моделі тензорного та 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 впливає на швидкодію мережі, тому при розширенні входів має бути враховане поєднання точного рішення та порівняно невеликого часу навчання.

**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 наступному 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 відомі - перевага буде у спроможності мережі виявляти подібність між різноманітними класами.

Мережа Кохонена має всього два прошарки: вхідний і вихідний, що називають самоорганізованою картою. Елементи карти розташовуються в деякому просторі - як правило двовимірному.

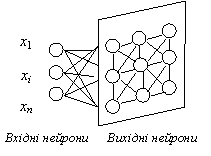


Рис. 4. Мережа Кохонена

Мережа Кохонена навчається методом послідовних наближень. Починаючи з випадковим чином обраного вихідного розташування центрів, алгоритм поступово покращується для кластеризації навчальних даних.

Проте, алгоритм може працювати і на іншому рівні. В результаті ітеративної процедури навчання мережа організовується таким чином, що елементи, які відповідають центрам, розташованим близько один від одного в просторі входів, будуть розташовані близько один від одного і на топологічній карті. Топологічний прошарок мережі можна уявити як двовимірну штахету, яку потрібно так відобразити в *N*-вимірний простір входів, щоб по можливості зберегти вихідну структуру даних. Звісно ж, при будь-якій спробі відтворити *N*-вимірний простір на площині буде загублено багато деталей, але такий прийом дозволяє користувачу візуалізувати дані, що неможливо зрозуміти іншим засобом.

Основний ітераційний алгоритм Кохонена послідовно проходить ряд епох, на кожній епосі опрацьовується один навчальний приклад. Вхідні сигнали - вектори дійсних чисел - послідовно пред'являються мережі. Бажані вихідні сигнали не визначаються. Після пред'явлення достатнього числа вхідних векторів, синаптичні ваги мережі визначають кластери. Крім того, ваги організуються так, що топологічне близькі вузли чуттєві до схожих вхідних сигналів.

Для реалізації алгоритму необхідно визначити міру сусідства нейронів (окіл нейрона-переможця). На мал. 6 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.

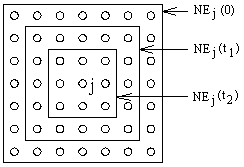


Рис. 5. Зони топологічного сусідства на карті ознак у різні моменти часу

**Алгоритм функціонування мережі Кохонена:**

1. Ініціалізація мережі. Ваговим коефіцієнтам мережі надаються малі випадкові значення. Початкова зона сусідства показана на рис. 5.
2. Пред'явлення мережі нового вхідного сигналу.
3. Обчислення відстані до всіх нейронів мережі:

Відстані *dj* від вхідного сигналу до кожного нейрона *j* визначаються за формулою:

http://www.victoria.lviv.ua/html/oio/images/theme7/image1391.gif

де *xi* - *i-ий* елемент вхідного сигналу в момент часу *t*, *wij*(*t*) - вага зв'язку від *i-го* елемента вхідного сигналу до нейрона *j* у момент часу *t*.

1. Вибір нейрона з найменшою відстанню:

Вибирається нейрон-переможець *j\**, для якого відстань *dj* найменше.

1. Налаштування ваг нейрона *j\** і його сусідів:

Робиться налаштування ваг для нейрона *j\** і всіх нейронів з його околу NE. Нові значення ваг:

*wij*(*t*+1)=*wij*(*t*)+*r*(*t*)(*xi*(*t*)-*wij*(*t*))

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

1. Повернення до кроку 2.

В алгоритмі використовується коефіцієнт швидкості навчання, який поступово зменшується, для тонкішої корекції на новій епосі. В результаті позиція центру встановлюється в певній позиції, яка 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 розпізнаванню 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 тільки в тому випадку, якщо заздалегідь відоме число кластерів.

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

***Модифікації.*** Одна з модифікацій полягає в тому, що до мережі Кохонена додається мережа MAXNET, що визначає нейрон з найменшою відстанню до вхідного сигналу.

**Квантування навчального вектора (*Learning VectorQuantization*)**

Мережа запропонована Тойво Кохоненом у середині 80-х рр., набагато пізніше за його початкову роботу по самоорганізованим картам. Мережа базується на прошарку Кохонена, який здатний до сортування прикладів у відповідні кластери і використовується як для проблем класифікації, так і для кластеризації зображень.

Мережа містить вхідний прошарок, самоорганізовану карту Кохонена та вихідний прошарок. Приклад мережі зображений на рис. 6. Вихідний прошарок має стільки нейронів, скільки є відмінних категорій або класів. Карта Кохонена має ряд нейронів, згрупованих для кожного з цих класів. Кількість елементів обробки на один клас залежить від складності відношення "вхід-вихід". Звичайно, кожен клас буде мати однакову кількість елементів по всьому прошарку. Прошарок Кохонена навчається класифікації за допомогою навчальної множини. Мережа використовує правила контрольованого навчання. Вхідний прошарок повинен містити лише один нейрон для кожного окремого вхідного параметра.

Квантування навчального вектора класифікує вхідні дані у визначені групування, тобто відображає *n*-вимірний простір у *m*-вимірний простір (бере *n* входів і створює *m* виходів). Карти зберігають відношення між близькими сусідами у навчальній множині так, що вхідні образи, які не були попередньо вивчені, будуть розподілені за категоріями їх найближчих сусідів у навчальних даних.

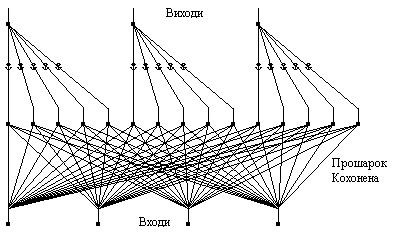


Рис. 6. Приклад мережі з квантуванням навчального вектора.

У режимі навчання, контрольована мережа використовує прошарок Кохонена, де обчислюється відстань від навчального вектора до кожного нейрону і найближчий нейрон оголошується переможцем. Існує лише один переможець на весь прошарок. Переможцю дозволено збуджувати лише один вихідний нейрон, оголошуючи клас або кластер до якого належить вхідний вектор. Якщо нейрон-переможець знаходиться у очікуваному класі навчального вектора, його ваги підсилюються у напрямку навчального вектора. Якщо нейрон-переможець не є у 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, пов'язаної з їх специфічним класом.

Під час режиму функціонування, обчислюється відстань від вхідного вектора до кожного нейрону і знову найближчий нейрон оголошується переможцем. Це в свою чергу генерує один вихід, визначаючи частковий клас, знайдений мережею.

***Недоліки.*** Для складних проблем класифікації з подібними вхідними прикладами, мережа вимагає великої карти Кохонена з багатьма нейронами на клас. Це вибірково може бути подолано вибором доцільних навчальних прикладів або розширенням вхідного прошарку.

Мережа квантування навчального вектора страждає від дефекту, що деякі нейрони мають тенденцію до перемоги занадто часто, тобто налаштовують свої ваги дуже швидко, а інші постійно залишаються незадіяними. Це часто трапляється, коли їх ваги мають значення далекі від навчальних прикладів. Щоб пом'якшити цю проблему, нейрон, який перемагає занадто часто штрафується, тобто зменшуються ваги його зв'язків з кожним вхідним нейроном. Це зменшення ваг є пропорційним до різниці між частотою перемог нейрону та частотою перемог середнього нейрону.

***Переваги.*** Алгоритм граничної корекції використовується для вдосконалення рішення навіть коли було знайдено відносно добре рішення. Алгоритм спроможний діяти, коли нейрон-переможець знаходиться у неправильному класі, а другий найкращий нейрон у правильному класі. Навчальний вектор повинен бути близько від середньої точки простору, що з'єднує ці два нейрони. Неправильний нейрон-переможець зміщується з навчального вектора, а нейрон з іншого місця посувається до навчального вектора. Ця процедура робить чіткішою межу між областями, де можлива невірна класифікація.

На початку навчання бажано відключити відштовхування. Нейрон-переможець пересувається до навчального вектора лише тоді, коли навчальний вектор та нейрон-переможець знаходяться в одному класі. Це право вибору доцільне, коли нейрон повинен оминати область, яка має відмінний клас для досягнення необхідної області.

**Тема 7. Класифікація відомих нейромереж по основних категоріях застосування (продовження)**

* [Перцептрон Розенбалата](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#Perceptron)
* [Нейромережа зворотного поширення похибки](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#BackPropagation)
* [Delta Bar Delta](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#Delta_Bar_Delta)
* [Extended Delta Bar Delta](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#Extended_Delta_Bar_Delta)
* [Скерований випадковий пошук](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#Skerovanyj_Poschuk)
* [Нейронна мережа вищого порядку або функціонально-пов'язана нейронна мережа](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#Vyshij_Porjadok)
* [Мережа Кохонена](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#Cohonen)
* [Квантування навчального вектора](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#LearningVectorQuantization)
* [Мережа зустрічного поширення CounterРropagation](http://www.victoria.lviv.ua/html/oio/html/theme7_1.htm#Counter%D0%A0ropagation)
* [Імовірнісна нейронна мережа](http://www.victoria.lviv.ua/html/oio/html/theme7_1.htm#ProbabilisticNeuralNetworks)
* [Мережа Хопфілда](http://www.victoria.lviv.ua/html/oio/html/theme7_1.htm#Hopfild)
* [Машина Больцмана](http://www.victoria.lviv.ua/html/oio/html/theme7_1.htm#BoltzmannMashine)
* [Мережа Хемінга](http://www.victoria.lviv.ua/html/oio/html/theme7_1.htm#Hamming)
* [Двоскерована асоціативна пам'ять](http://www.victoria.lviv.ua/html/oio/html/theme7_1.htm#AsociatyveMemory)
* [Мережа адаптивної резонансної теорії](http://www.victoria.lviv.ua/html/oio/html/theme7.htm#AdaptiveResonanceTheory)

**Мережа зустрічного поширення (*CounterРropagation*)**

Роберт Хехт-Нільсен (*Robert Hecht-Nielsen*) розробив мережу *СounterРropagation* як засіб для поєднання неконтрольованого прошарку Кохонена із контрольованим вихідним прошарком. Мережа призначена для вирішення складних класифікацій, при мінімізації числа нейронів та часу навчання. Навчання для мережі *СounterРropagation* подібне до мережі з квантуванням навчального вектора.

Приклад мережі зображений на рис. 7. Односкерована мережа *CounterPropagation* має три прошарки: вхідний прошарок, самоорганізовану карту Кохонена та вихідний прошарок, що використовує правило "дельта" для зміни вхідних ваг з'єднань. Цей прошарок називають прошарком Гросберга.

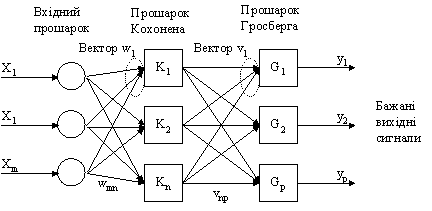


Рис. 7. Мережа зустрічного поширення без зворотних зв'язків

Перша мережа *СounterРropagation* складалась із двоскерованого відображення між вхідним та вихідним прошарками. Дані надходять на вхідний прошарок для генерації класифікації на вихідному прошарку, вихідний прошарок по черзі приймає додатковий вхідний вектор та генерує вихідну класифікацію на вхідному прошарку мережі. Через такий зустрічно-поширений потік інформації випливає назва мережі. Багато розробників використовують односкерований варіант *СounterРropagation*, коли існує лише один шлях прямого поширення від вхідного до вихідного прошарку.

У мережі зустрічного поширення об'єднані два алгоритми: самоорганізована карта Кохонена і зірка Гросберга (*Grossberg Outstar*). Кожен елемент вхідного сигналу подається на всі нейрони прошарку Кохонена. Ваги зв'язків (*wmn*) утворюють матрицю *W*. Кожен нейрон прошарку Кохонена з'єднаний зі всіма нейронами прошарку Гросберга. Ваги зв'язків (*vnp*) утворюють матрицю ваг *V*. Нейронні мережі, що поєднують різні нейропарадигми як будівельні блоки, більш близькі до мозку по архітектурі, ніж однорідні структури. Вважається, що в мозку саме каскадні з'єднання модулів різної спеціалізації дозволяють виконувати необхідні обчислення.

У процесі навчання мережі зустрічного поширення вхідні вектори асоціюються з відповідними вихідними векторами (двійковими або аналоговими). Після навчання мережа формує вихідні сигнали, що відповідають вхідним сигналам. Узагальнююча здатність мережі дає можливість одержувати правильний вихід, якщо вхідний вектор неповний чи спотворений.

**Навчання мережі**

Карта Кохонена класифікує вхідні вектори в групи схожих. В результаті самонавчання прошарок здобуває здатність розділяти несхожі вхідні вектори. Який саме нейрон буде активуватися при пред'явленні конкретного вхідного сигналу, заздалегідь важко передбачити.

При навчанні прошарку Кохонена на вхід подається вхідний вектор і обчислюються його скалярні добутки з векторами ваг всіх нейронів.

Скалярний добуток є мірою подібності між вхідним вектором і вектором ваг. Нейрон з максимальним значенням скалярного добутку з'являється "переможцем" і його ваги підсилюються (ваговий вектор наближається до вхідного).

*wн*=*wc*+*r*(*x*-*wc*)

де *w*н - нове значення ваги, що з'єднує вхідний компонент *x* з нейроном-переможцем, *w*с - попереднє значення цієї ваги, *r* - коефіцієнт швидкості навчання, що спочатку звичайно дорівнює 0.7 і може поступово зменшуватися в процесі навчання. Це дозволяє робити великі початкові кроки для швидкого грубого навчання і менші кроки при підході до остаточної величини.

Кожна вага, зв'язана з нейроном-переможцем Кохонена, змінюється пропорційно різниці між його величиною і величиною входу, до якого він приєднаний. Напрямок зміни мінімізує різницю між вагою і відповідним елементом вхідного прошарку.

Навчальна множина може містити багато подібних між собою вхідних векторів, і мережа повинна бути навченою активувати один нейрон Кохонена для кожного з них. Ваги цього нейрона уявляють собою усереднення вхідних векторів, що його активують.

Виходи прошарку Кохонена подаються на входи нейронів прошарку Гросберга. Входи нейронів обчислюються як зважена сума виходів прошарку Кохонена. Кожна вага коректується лише в тому випадку, якщо вона з'єднана з нейроном Кохонена, який має ненульовий вихід. Величина корекції ваги пропорційна різниці між вагою і необхідним виходом нейрона Гросберга. Навчання прошарку Гросберга - це навчання "з вчителем", алгоритм використовує задані бажані виходи.

**Функціонування мережі**

У своїй найпростішій формі прошарок Кохонена функціонує за правилом "переможець отримує все". Для даного вхідного вектора один і тільки один нейрон Кохонена видає логічну одиницю, всі інші видають нуль.

Прошарок Гросберга функціонує в схожій манері. Його вихід є зваженою сумою виходів прошарку Кохонена.

Якщо прошарок Кохонена функціонує таким чином, що лише один вихід дорівнює одиниці, а інші дорівнюють нулю, то кожен нейрон прошарку Гросберга видає величину ваги, що зв'язує цей нейрон з єдиним нейроном Кохонена, чий вихід відмінний від нуля.

У повній моделі мережі зустрічного поширення є можливість одержувати вихідні сигнали по вхідним і навпаки. Цим двом діям відповідають пряме і зворотне поширення сигналів.

***Області застосування.*** Розпізнавання образів, відновлення образів (асоціативна пам'ять), стиснення даних (із втратами).

***Недоліки.*** Мережа не дає можливості будувати точні апроксимації (точні відображення). У цьому мережа значно уступає мережам зі зворотним поширенням похибки. До недоліків моделі також варто віднести слабкий теоретичний базис модифікацій мережі зустрічного поширення.

***Переваги:***

* Мережа зустрічного поширення проста. Вона дає можливість витягати статистичні властивості з множини вхідних сигналів. Кохонен довів, що для навченої мережі імовірність того, що випадково обраний вхідний вектор буде найближчим до будь-якого заданого вагового вектора, дорівнює *1/k, k* - число нейронів Кохонена.
* Мережа швидко навчається. Час навчання в порівнянні зі зворотним поширенням може бути в 100 разів менше.
* По своїх можливостях будувати відображення мережа зустрічного поширення значно перевершує одношарові перцептрони.
* Мережа корисна для застосувань, у яких потрібно швидка початкова апроксимація.
* Мережа дає можливість будувати функцію і зворотну до неї, що знаходить застосування при рішенні практичних задач.

***Модифікації.*** Мережі зустрічного поширення можуть розрізнятися способами визначення початкових значень синаптичних ваг.

Для підвищення ефективності навчання застосовується додавання шуму до вхідних векторів.

Ще один метод підвищення ефективності навчання - надання кожному нейрону "почуття справедливості". Якщо нейрон стає переможцем частіше, ніж *1/k* (*k* - число нейронів Кохонена), то йому тимчасово збільшують поріг, даючи тим самим навчатися й іншим нейронам.

Крім "методу акредитації", при якому для кожного вхідного вектора активується лише один нейрон Кохонена, може бути використаний "метод інтерполяції", при використанні якого ціла група нейронів Кохонена, що мають найбільші виходи, може передавати свої вихідні сигнали в прошарок Гросберга. Цей метод підвищує точність відображень, реалізованих мережею.

**Імовірнісна нейронна мережа**

Імовірнісна нейронна мережа була розроблена Дональдом Спехтом (*Donald Specht*). Ця мережна архітектура була вперше представлена у двох статтях : "Імовірнісні нейронні мережі для класифікації" (*Probabilistic Neural Networks for Classification*) 1988, "Відображення або асоціативна пам'ять та імовірнісні нейронні мережі" (*Mapping or Associative Memory and Probabilistic Neural Networks*) 1990 р.

Виходи мережі можна інтерпретувати, як оцінки ймовірності належності елементу певному класу. Імовірнісна мережа вчиться оцінювати функцію густини ймовірності, її вихід розглядається як очікуване значення моделі в даній точці простору входів. Це значення пов'язане з густиною ймовірності спільного розподілу вхідних і вихідних даних.

Задача оцінки густини ймовірності відноситься до області байєсівської статистики. Звичайна статистика по заданій моделі показує, яка ймовірність того або іншого виходу (наприклад, на гральній кістці 6 очок буде випадати в середньому в одному випадку з шістьох). Байєсова статистика інтерпретує по іншому: правильність моделі оцінюється по наявних достовірних даних, тобто надає можливість оцінювати густину ймовірності розподілу параметрів моделі по наявних даних.

При рішенні задач класифікації можна оцінити густину ймовірності для кожного класу, порівняти між собою ймовірності приналежності до різних класів і обрати модель з параметрами, при яких густина ймовірності буде найбільшою.

Оцінка густини ймовірності в мережі заснована на ядерних оцінках. Якщо приклад розташований в даній точці простору, тоді в цій точці є певна густина ймовірності. Кластери з близько розташованих точок, свідчать, що в цьому місці густина імовірності велика. Поблизу спостереження є більша довіра до рівня густини, а по мірі віддалення від нього довіра зменшується і плине до нуля. В методі ядерних оцінок в точку, що відповідає кожному прикладу, поміщається деяка проста функція, потім вони всі додаються і в результаті утворюється оцінка для загальної густини імовірності. Найчастіше в якості ядерних функцій беруть дзвоноподібні функції (гаусові). Якщо є достатня кількість навчальних прикладів, такий метод дає добрі наближення до істинної густини імовірності.

Імовірнісна мережа має три прошарки: вхідний, радіальний та вихідний. Радіальні елементи беруться по одному на кожний приклад. Кожний з них містить гаусову функцію з центром в цьому прикладі. Кожному класу відповідає один вихідний елемент. Вихідний елемент з'єднаний лише з радіальними елементами, що відносяться до його класу і підсумовує виходи всіх елементів, що належать до його класу. Значення вихідних сигналів утворюються пропорційно ядерних оцінок ймовірності приналежності відповідним класам.

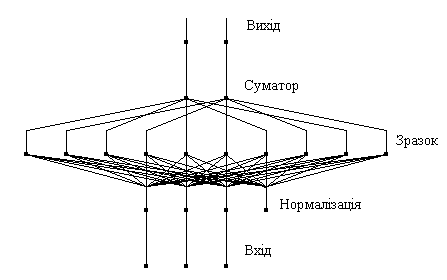


Рис. 8. Приклад імовірнісної нейронної мережі

Базова модель імовірнісної мережі має модифікації. Припустимо, що пропорції класів у навчальній множині відповідають їх пропорціям у всій досліджуваній вибірці (апріорна імовірність). Наприклад, якщо серед всіх людей хворими є 2%, то в навчальній множині для мережі, яка діагностує захворювання, хворих також повинно бути 2%. Якщо ж апріорні імовірності відрізняються від пропорції в навчальній виборці, мережа буде видавати невірний результат. Це можна врахувати, вводячи корегуючі коефіцієнти для різноманітних класів.

Навчання імовірнісної нейронної мережі є набагато простішим, ніж ВackРropagation. Істотним недоліком мережі є її розмір, оскільки вона фактично вміщує в собі всі навчальні дані, потребує багато пам'яті і може повільно працювати.

**Мережа Хопфілда**

Джон Хопфілд вперше представив свою асоціативну мережу у 1982 р. у Національній Академії Наук. На честь Хопфілда та нового підходу до моделювання, ця мережна парадигма згадується як мережа Хопфілда. Мережа базується на аналогії фізики динамічних систем. Початкові застосування для цього виду мережі включали асоціативну, або адресовану за змістом пам'ять та вирішували задачі оптимізації.

Мережа Хопфілда використовує три прошарки: вхідний, прошарок Хопфілда та вихідний прошарок. Кожен прошарок має однакову кількість нейронів. Входи прошарку Хопфілда під'єднані до виходів відповідних нейронів вхідного прошарку через змінні ваги з'єднань. Виходи прошарку Хопфілда під'єднуються до входів всіх нейронів прошарку Хопфілда, за винятком самого себе, а також до відповідних елементів у вихідному прошарку. В режимі функціонування, мережа скеровує дані з вхідного прошарку через фіксовані ваги з'єднань до прошарку Хопфілда. Прошарок Хопфілда коливається, поки не буде завершена певна кількість циклів, і біжучий стан прошарку передається на вихідний прошарок. Цей стан відповідає образу, вже запрограмованому у мережу.

Навчання мережі Хопфілда вимагає, щоб навчальний образ був представлений на вхідному та вихідному прошарках одночасно. Рекурсивний характер прошарку Хопфілда забезпечує засоби корекції всіх ваг з'єднань. Недвійкова реалізація мережі повинна мати пороговий механізм у передатній функції. Для правильного навчання мережі відповідні пари "вхід-вихід" мають відрізнятися між собою.

Якщо мережа Хопфілда використовується як пам'ять, що адресується за змістом вона має два головних обмеження. По-перше, число образів, що можуть бути збережені та точно відтворені є строго обмеженим. Якщо зберігається занадто багато параметрів, мережа може збігатись до нового неіснуючого образу, відмінному від всіх запрограмованих образів, або не збігатись взагалі. Межа ємності пам'яті для мережі приблизно 15% від числа нейронів у прошарку Хопфілда. Другим обмеженням парадигми є те, що прошарок Хопфілда може стати нестабільним, якщо навчальні приклади є занадто подібними. Зразок образу вважається нестабільним, якщо він застосовується за нульовий час і мережа збігається до деякого іншого образу з навчальної множини. Ця проблема може бути вирішена вибором навчальних прикладів більш ортогональних між собою.

Структурна схема мережі Хопфилда приведена на рис. 9.

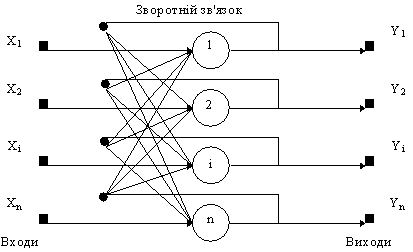


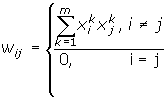
Рис. 9. Структурна схема мережі Хопфілда.

Задача, розв'язувана даною мережею в якості асоціативної пам'яті, як правило, формулюється таким чином. Відомий деякий набір двійкових сигналів (зображень, звукових оцифровок, інших даних, що описують якийсь об'єкти або характеристики процесів), вважають зразковим. Мережа повинна вміти з зашумленого сигналу, поданого на її вхід, виділити ("пригадати" по частковій інформації) відповідний зразок або "дати висновок" про те, що вхідні дані не відповідають жодному із зразків. У загальному випадку, будь-який сигнал може бути описаний вектором *x*1, *хі*, *хn*..., *n* - число нейронів у мережі і величина вхідних і вихідних векторів. Кожний елемент *xi* дорівнює або +1, або -1. Позначимо вектор, що описує *k*-ий зразок, через *Xk*, а його компоненти, відповідно, - *xik*, *k*=0, ..., *m*-1, *m* - число зразків. Якщо мережа розпізнає (або "пригадує") якийсь зразок на основі пред'явлених їй даних, її виходи будуть містити саме його, тобто *Y = Xk*, де *Y* - вектор вихідних значень мережі: *y*1*, yi, yn*. У противному випадку, вихідний вектор не співпаде з жодний зразковим.

Якщо, наприклад, сигнали являють собою якесь зображення, то, відобразивши у графічному виді дані з виходу мережі, можна буде побачити картинку, що цілком збігається з однієї зі зразкових (у випадку успіху) або ж "вільну імпровізацію" мережі (у випадку невдачі).

**Алгоритм функціонування мережі**

1. На стадії ініціалізації мережі вагові коефіцієнти синапсів встановлюються таким чином:



Тут *i* і *j* - індекси, відповідно, предсинаптичного і постсинаптичного нейронів; *xik*, *xjk* - *i*-ий і *j*-ий елементи вектора *k*-ого зразка.

1. На входи мережі подається невідомий сигнал (t - номер ітерації). Його поширення безпосередньо встановлює значення виходів:

*yi(0) = xi , i = 0...n-1,*

тому позначення на схемі мережі вхідних сигналів у явному виді носить чисто умовний характер. Нуль у скобці справа від *yi* означає нульову ітерацію в циклі роботи мережі.

1. Розраховується новий стан нейронів

http://www.victoria.lviv.ua/html/oio/images/theme7/image1397.gif, j=0...n-1

і нові значення виходів

http://www.victoria.lviv.ua/html/oio/images/theme7/image1398.gif

де *f* - передатна функція у виді порогової, приведена на рис. 10.

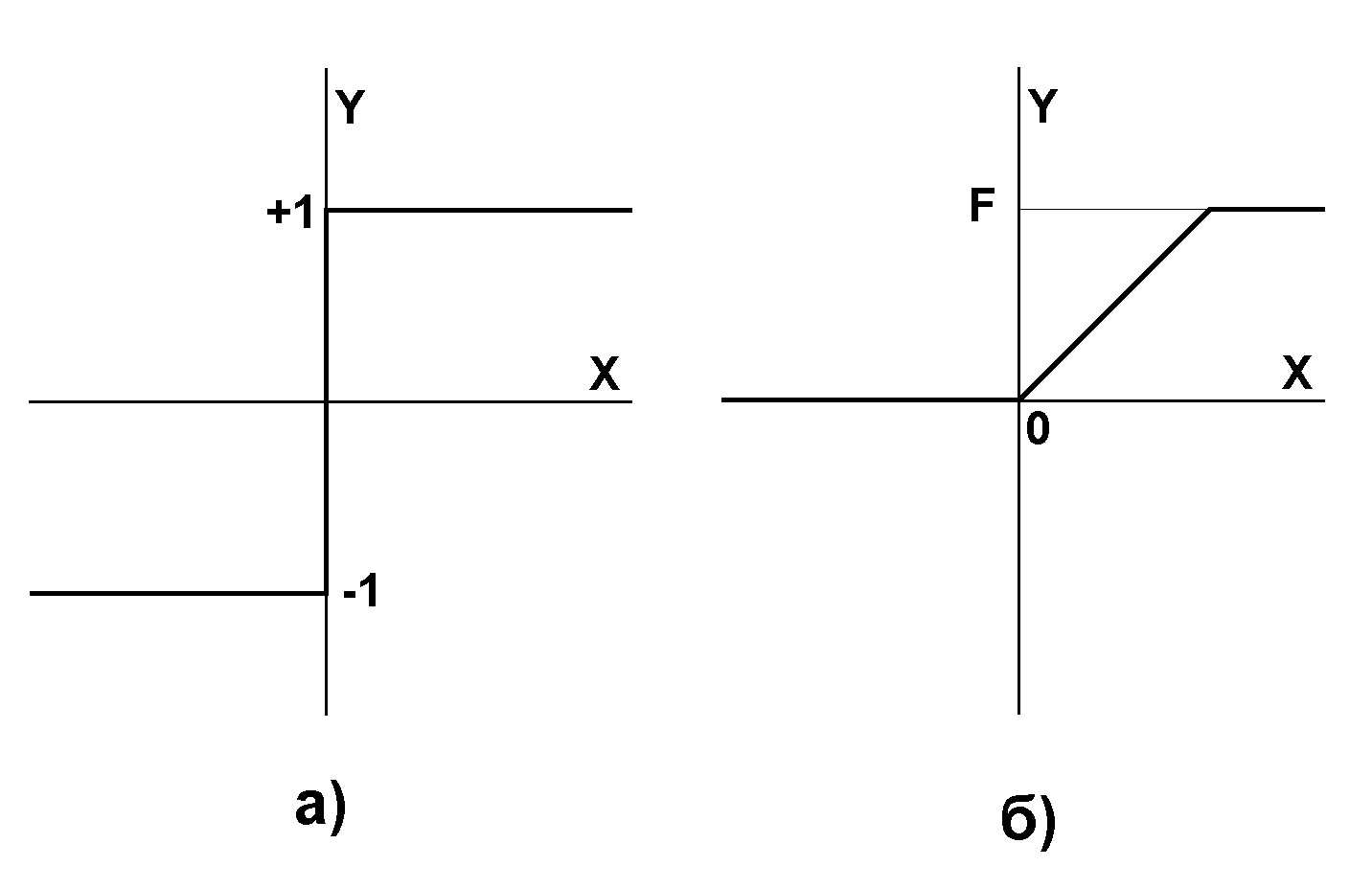


Рис. 10. Передатні функції

1. Перевіряємо чи змінилися вихідні значення виходів за останню ітерацію. Якщо так - перехід до пункту 2, інакше (якщо виходи стабілізувались) - кінець. При цьому вихідний вектор являє собою зразок, що найкраще відповідає вхідним даним.

Іноді мережа не може провести розпізнавання і видає на виході неіснуючий образ. Це пов'язано з проблемою обмеженості можливостей мережі. Для мережі Хопфилда число запам'ятованих образів *m* не повинно перевищувати величини, приблизно рівної 0.15•*n*. Крім того, якщо два образи А и Б сильно схожі, вони, можливо, будуть викликати в мережі перехресні асоціації, тобто пред'явлення на входи мережі вектора А призведе до появи на її виходах вектори Б и навпаки.

**Машина Больцмана**

Машина Больцмана (*Boltzmann mashine*) є подібною за функцією та дією до мережі Хопфілда і включає поняття "модельованого відпалу" для пошуку в просторі станів прошарку образів глобальний мінімум.

Еклі (*Ackley*), Хінтон (*Hinton*) та Сейновскі (*Sejnowski*) розробили правило больцманівського навчання у 1985 р. Подібно до мережі Хопфілда, машина Больцмана має простір станів, який базується на вагах з'єднань у прошарку образів. Процеси навчання мережі, наповненої образами, включає вивчення рельєфу простору станів. Під час ітеративного навчання знаходяться краща множина вихідних значень.

В процесі навчання машина Больцмана моделює відпал металу. Як і при фізичному відпалі, температура починається з вищих значень і зменшується з часом. Збільшена температура додає збільшений шумовий коефіцієнт до кожного нейрону у прошарку образів. Звичайно, кінцева температура є нулем. Для досягнення кращого рішення доцільно на нижчих температурах додавати більше ітерацій.

Машина Больцмана, навчаючись на високій температурі, веде себе більш подібно до випадкової моделі, а на низьких температурах вона веде себе як детермінована модель. Через випадкову компоненту у відпаловому навчанні, нейрон може прийняти нове значення стану, що збільшується швидше, ніж зменшується загальний простір станів. Імітація фізичного відпалу дозволяє уникаючи локальний мінімум, просуватись до глобального.

Як і в мережі Хопфілда, мережі може бути представлений частковий образ для доповнення відсутньої інформації. Обмеження на число класів, що є менше ніж 15 % від загальної кількості елементів у прошарку образів, все ще застосовується.

**Алгоритм функціонування мережі**

1. Визначити змінну *T*, що представляє штучну температуру.
2. Пред'явити мережі множину входів і обчислити виходи та цільову функцію.
3. Дати випадкову зміну вагам і перерахувати вихід мережі та зміну цільової функції у відповідності зі зробленою зміною ваг.
4. Якщо цільова функція зменшилась, то зберегти зміну ваг.

Якщо зміна ваг приводить до збільшення цільової функції, то імовірність збереження цієї зміни обчислюється за допомогою розподілу Больцмана:

http://www.victoria.lviv.ua/html/oio/images/theme7/image1400.gif

де *P*(*c*) - імовірність зміни *c* у цільовій функції; *k* - константа, аналогічна константі Больцмана, що вибирається в залежності від задачі; *T* - штучна температура.

Вибирається випадкове число *r* з рівномірного розподілу від нуля до одиниці. Якщо *P*(*c*) більше, ніж *r*, то зміна зберігається, у противному випадку величина ваги повертається до попереднього значення.

Ця процедура дає можливість системі робити випадковий крок у напрямку, що псує цільову функцію, дозволяючи їй тим самим вириватися з локальних мінімумів.

Кроки 3 і 4 повторюються для кожної з ваг мережі, поступово зменшуючи температуру *T*, поки не буде досягнуте припустиме низьке значення цільової функції. У цей момент пред'являється інший вхідний вектор і процес навчання повторюється. Мережа навчається на всіх векторах навчальної множини, поки цільова функція не стане припустимої для всіх з них.

Швидкість зменшення температури повинна бути зворотно пропорційна логарифму часу. При цьому мережа збігається до глобального мінімуму.

***Області застосування.*** Розпізнавання образів, класифікація.

***Недоліки.*** Повільний алгоритм навчання.

***Переваги.***Алгоритм дає можливість мережі вибиратися з локальних мінімумів адаптивного рельєфу простору станів.

***Модифікації.*** Випадкові зміни можуть проводитися не тільки для окремих ваг, але і для всіх нейронів прошарків в багатошарових мережах або для всіх нейронів мережі одночасно. Ці модифікації алгоритму дають можливість скоротити загальне число ітерацій навчання.

**Мережа Хемінга**

Мережа Хемінга (*Hamming*) є розширенням мережі Хопфілда. Ця мережа була розроблена Річардом Ліппманом (*Richard Lippman*) у середині 80-х рр. Мережа Хемінга реалізує класифікатор, що базується на найменшій похибці для векторів двійкових входів, де похибка визначається відстанню Хемінга. Відстань Хемінга визначається як число бітів, які відрізняються між двома відповідними вхідними векторами фіксованої довжини. Один вхідний вектор є незашумленим прикладом образу, інший є спотвореним образом. Вектор виходів навчальної множини є вектором класів, до яких належать образи. У режимі навчання вхідні вектори розподіляються до категорій для яких відстань між зразковими вхідними векторами та біжучим вхідним вектором є мінімальною.

Мережа Хемінга має три прошарки: вхідний прошарок з кількістю вузлів, скільки є окремих двійкових ознак; прошарок категорій (прошарок Хопфілда), з кількістю вузлів, скільки є категорій або класів; вихідний прошарок, який відповідає числу вузлів у прошарку категорій.

Мережа є простою архітектурою прямого поширення з вхідним рівнем, повністю під'єднаним до прошарку категорій. Кожен елемент обробки у прошарку категорій є зворотно під'єднаним до кожного нейрона у тому ж самому прошарку і прямо під'єднаним до вихідного нейрону. Вихід з прошарку категорій до вихідного прошарку формується через конкуренцію.

Навчання мережі Хемінга є подібним до методології Хопфілда. На вхідний прошарок надходить бажаний навчальний образ, а на виході вихідного прошарку надходить значення бажаного класу, до якого належить вектор. Вихід містить лише значення класу до якої належить вхідний вектор. Рекурсивний характер прошарку Хопфілда забезпечує засоби корекції всіх ваг з'єднань.

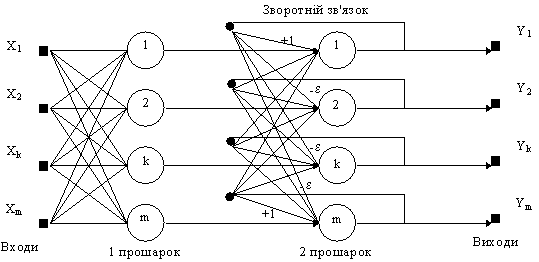


Рис. 11. Структурна схема мережі Хемінга

**Алгоритм функціонування мережі Хемінга**

1. На стадії ініціалізації ваговим коефіцієнтам першого прошарку і порогу передатної функції присвоюються такі значення:

*Wik*=*xIk*/2, i=0...n-1, k=0...m-1

*bk* = *n / 2, k = 0...m-1*

Тут *xik* - *i*-ий елемент *k*-ого зразка.

Вагові коефіцієнти гальмуючих синапсів у другому прошарку беруть рівними деякій величині 0 < *v* < 1/m. Синапс нейрона, пов'язаний із його ж виходом має вагу +1.

1. На входи мережі подається невідомий вектор *x*1*, xi*, *xn* *...*Розраховуються стани нейронів першого прошарку (верхній індекс у скобках указує номер прошарку):

http://www.victoria.lviv.ua/html/oio/images/theme7/image1403.gif*, j=0...m-1*

Після цього отримання значення ініціалізують значення виходів другого прошарку:

*yj(2) = yj(1), j = 0...m-1*

1. Обчислюються нові стани нейронів другого прошарку:

http://www.victoria.lviv.ua/html/oio/images/theme7/image1404.gif

і значення їх виходів:

http://www.victoria.lviv.ua/html/oio/images/theme7/image1405.gif

Передатна функція *f* має вид порога, причому величина *b* повинна бути достатньо великою, щоб будь-які можливі значення аргументу не призводили до насичення.

1. Перевіряється, чи змінилися виходи нейронів другого прошарку за останню ітерацію. Якщо так - перейти до кроку 3. Інакше - кінець.

Роль першого прошарку є умовною: скориставшись один раз на першому кроці 1 значеннями його вагових коефіцієнтів, мережа більше не вертається до нього, тому перший прошарок може бути взагалі виключений із мережі.

Мережа Хемінга має ряд переваг над мережею Хопфілда. Вона реалізує оптимальний класифікатор мінімуму похибки, якщо похибки вхідних бітів є випадковими та незалежними. Для функціонування мережі Хемінга потрібна менша кількість нейронів, оскільки середній прошарок вимагає лише один нейрон на клас, замість нейрону на кожен вхідний вузол. І, нарешті, мережа Хемінга не страждає від неправильних класифікацій, які можуть трапитись у мережі Хопфілда. В цілому, мережа Хемінга є як швидшою, так і точнішою за мережу Хопфілда.

**Двоскерована асоціативна пам'ять**

Ця мережна модель була розроблена Бартом Козко (*Bart Kosko*) і розширює модель Хопфілда. Множина парних образів навчається за образами, що представлені як біполярні вектори. Подібно до мережі Хопфілда, коли представляється зашумлена версія одного образу, визначається найближчий образ, асоційований з ним.

На рис. 12 показаний приклад двоскерованої асоціативної пам'яті. Вона має стільки входів, скільки є вихідних нейронів. Два приховані прошарки містяться на двох окремих асоціативних елементах пам'яті і представляють подвоєний розмір вхідних векторів. Середні прошарки повністю з'єднуються один з одним. Вхідний та вихідний прошарки потрібні для реалізації засобів введення та відновлення інформації з мережі.

Середні прошарки розроблені для збереження асоційованих пар векторів. Коли зашумлений вектор образу вважається вхідним, середні прошарки коливаються до досягнення стабільного стану рівноваги, який відповідає найближчій навченій асоціації і буде генерувати початковий навчальний образ на виході. Подібно до мережі Хопфілда, двоскерована асоціативна пам'ять є схильною до неправильного відшукування навченого образу, якщо надходить невідомий вхідний вектор, який не був у складі навчальної множини.

Двоскерована асоціативна пам'ять відноситься до гетероасоціативної пам'яті. Вхідний вектор надходить на один набір нейронів, а відповідний вихідний вектор продукується на іншому наборі нейронів. Вхідні образи асоціюються з вихідними.

Для порівняння: мережа Хопфилда є автоасоціативною. Вхідний образ може бути відновлений чи виправлений мережею, але не може бути асоційований з іншим образом. У мережі Хопфілда використовується одношарова структура асоціативної пам'яті, у якій вихідний вектор з'являється на виході тих же нейронів, на які надходить вхідний вектор.

Двоскерована асоціативна пам'ять, як і мережа Хопфілда, здатна до узагальнення, виробляючи правильні вихідні сигнали, незважаючи на спотворені входи.

Розглянемо схему двоскерованої асоціативної пам'яті. Вхідний вектор *A* обробляється матрицею ваг *W* мережі, у результаті чого продукується вектор вихідних сигналів мережі *B*. Вектор *B* обробляється транспонованою матрицею *W*T ваг мережі, яка продукує сигнали, що представляють новий вхідний вектор *A*. Цей процес повторюється доти, поки мережа не досягне стабільного стану, у якому ні вектор *A*, ні вектор *B* не змінюються.

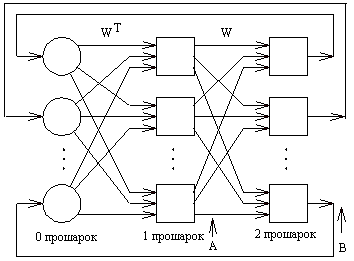


Рис. 12. Двоскерована асоціативна пам'ять

Нейрони в прошарках 1 і 2 функціонують, як і в інших парадигмах, обчислюючи суму зважених входів і значення передатної функції *F*:

http://www.victoria.lviv.ua/html/oio/images/theme7/image1407.gif

або у векторній формі:

*B*=*F*(*AW*)

де *B* - вектор вихідних сигналів нейронів прошарку 2, *A* - вектор вихідних сигналів нейронів прошарку 1, *W* - матриця ваг зв'язків між прошарками 1 і 2, *F* - передатна функція.

Аналогічно *A*=*F*(*BWT*), де *WT* є транспозицією матриці *W*.

В якості передатної функції використовується експонентна сигмоїда.

Прошарок 0 не робить обчислень і не має пам'яті. Він є лише засобом розподілу вихідних сигналів прошарку 2 до елементів матриці *WT*.

Формула для обчислення значень синаптичних ваг:

http://www.victoria.lviv.ua/html/oio/images/theme7/image1408.gif

де *Aj* і *Bj* - вхідні і вихідні сигнали навчальної вибірки.

Вагова матриця обчислюється як сума добутків всіх векторних пар навчальної вибірки.

Системи зі зворотним зв'язком мають тенденцію до коливань. Вони можуть переходити від стану до стану, ніколи не досягаючи стабільності. Доведено, що двоскерована асоціативна пам'ять безумовно стабільна при будь-яких значеннях ваг мережі.

***Області застосування.*** Асоціативна пам'ять, розпізнавання образів.

***Недоліки.*** Ємність двоскерованої асоціативної пам'яті жорстко обмежена. Якщо *n* - кількість нейронів у вхідному прошарку, то число векторів, що можуть бути запам'ятовані в мережі не перевищує *L*=*n*/2*log*2*n*. Так, якщо *n*=1024, то мережа здатна запам'ятати не більш 25 образів.

Двоскерована асоціативна пам'ять має деяку непередбачуваність у процесі функціонування, можливі помилкові відповіді.

***Переваги:***

* У порівнянні з автоасоціативною пам'яттю (наприклад, мережею Хопфилда), двоскерована асоціативна пам'ять дає можливість будувати асоціації між векторами *A* і *B*, що у загальному випадку мають різні розмірності. За рахунок таких можливостей гетероасоціативна пам'ять має більш широкий клас застосувань, ніж автоасоціативна пам'ять.
* Двоскерована асоціативна пам'ять - проста мережа, що може бути реалізована у виді окремої СБИС чи оптоелектронним способом.
* Процес формування синаптичних ваг простий і швидкий. Мережа швидко збігається в процесі функціонування.

***Модифікації:***

* Сигнали в мережі можуть бути як дискретними, так і аналоговими. Для обох випадків доведена стабільність мережі.
* Були запропоновані моделі двоскерованої асоціативної пам'яті з навчанням без вчителя (адаптивна двоскерована асоціативна пам'ять).
* Введення латеральних зв'язків всередині прошарку дає можливість реалізувати конкуруючу двоскеровану асоціативну пам'ять.

**Мережа адаптивної резонансної теорії**

Розроблена Стівеном Гросбергом та Карпентером у середині 80-х рр. Парадигма використовує функцію неконтрольованого навчання і аналізує значні вхідні дані, виявляє можливі ознаки та класифікує образи у вхідному векторі.

Мережа адаптивної резонансної теорії складається з двох взаємопов'язаних прошарків нейронів, розташованих між вхідним та вихідним прошарками. Кожен вхідний образ нижчого прошарку резонансу стимулювати очікуваний образ на вищому прошарку, який пересилається знов до нижчого прошарку, щоб впливати на наступний вхід. Це створює "резонанс" між нижчим та вищим прошарками для полегшення мережної адаптації образів.

Мережа переважно використовується у біологічному моделюванні, проте існують деякі технічні застосування. Головним обмеженням мережної архітектури є її шумова чутливість. Навіть мала кількість шуму на вхідному векторі плутає узагальнюючі можливості навченої мережі.

Мережа ART-1 навчається без учителя і реалізує алгоритм кластеризації, дуже схожий на алгоритм "послідовного лідера". Відповідно до цього алгоритму перший вхідний сигнал вважається зразком першого кластера. Наступний вхідний сигнал порівнюється зі зразком першого кластера. Говорять, що вхідний сигнал "прямує за лідером" і належить першому кластеру, якщо відстань до зразка першого кластера менше порога. У противному випадку другий вхідний сигнал - зразок другого кластера. Цей процес повторюється для всіх наступних вхідних сигналів. Таким чином, число кластерів росте з часом і залежить як від значення порога, так і від метрики відстані, що використовується для порівняння вхідних сигналів і зразків класів.

Основна частина мережі ART-1 схожа з мережею Хемінга. За допомогою послідовних зв'язків обчислюється відповідність вхідних сигналів і зразків кластерів. Максимальне значення відповідності підсилюється за допомогою латеральних зв'язків вихідних нейронів.

Мережа ART-1 відрізняється від мережі Хемінга зворотними зв'язками від вихідних нейронів до вхідних, крім того є можливість виключати вихідний нейрон з максимальним значенням відповідності і проводити тестування відповідності вхідного сигналу і зразків кластерів, як того вимагає алгоритм "послідовного лідера".

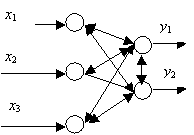


Рис. 13. Основні компоненти класифікатора Карпентер/Гросберга

**Алгоритм функціонування мережі**

1. Ініціалізація мережі:

*vij*(0)=1;*Wij*(0)=1/(1+*N*)

0 *i*  *N*-1; 0 *j*  *M*-1; 0 *b*  1;

де *wij*(*t*) - синаптична вага зв'язку від *i-го* нейрона першого прошарку до *j-го* нейрона другого прошарку в момент часу *t*, *vij*(*t*) - синаптична вага зв'язку від *i-го* нейрона другого прошарку до *j-го* нейрона першого прошарку в момент часу *t*, *b* - значення порога.

Ваги *vij*(*t*) і *wij*(*t*) визначають зразок, що відповідає нейрону *j*.

Поріг *b* показує, наскільки повинен вхідний сигнал збігатися з одним із запам'ятованих зразків, щоб вони вважалися схожими. Близьке до одиниці значення порога вимагає майже повного збігу.

При малих значеннях порога вхідний сигнал і зразок, навіть якщо вони сильно розрізняються вважаються приналежними одному кластеру.

1. Пред'явлення мережі нового бінарного вхідного сигналу:

Вхідні сигнали пред'являються вихідному прошарку нейронів аналогічно тому, як це робиться в мережі Хемінга.

1. Обчислення значень відповідності:

http://www.victoria.lviv.ua/html/oio/images/theme7/image1412.gif

Значення відповідності обчислюються паралельно для всіх зразків, запам'ятованих у мережі, аналогічно мережі Хемінга.

1. Вибір зразка з найбільшою відповідністю:

*yj*=*max*(*yj*)

Ця операція виконується за допомогою латерального гальмування.

1. Порівняння з порогом:

http://www.victoria.lviv.ua/html/oio/images/theme7/image1413.gif; http://www.victoria.lviv.ua/html/oio/images/theme7/image1414.gif

якщо http://www.victoria.lviv.ua/html/oio/images/theme7/image1415.gifперехід до кроку 7, інакше до кроку 6.

На цьому кроці обчислюється відношення скалярного добутку вхідного сигналу і зразка з найбільшим значенням відповідності до числа одиничних біт вхідного сигналу. Значення відношення порівнюється з порогом, введеному на першому кроці.

Якщо значення відношення більше порога, то вхідний сигнал вважається схожим на зразок з найбільшим значенням відповідності. У цьому випадку зразок модифікується шляхом виконання операції AND (логічне "I"). Новий зразок є зразок на попередньому кроці + вхідний сигнал.

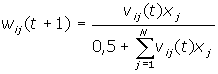
Якщо значення відношення менше порога, то вважається, що вхідний сигнал відрізняється від всіх зразків і він розглядається як новий зразок. У мережу вводиться нейрон, що відповідає новому зразку, і обчислюються значення синаптичних ваг.

1. Виключення приклада з найбільшим значенням відповідності:

Вихід нейрона з найбільшим значенням відповідності тимчасово встановлюється рівним нулю і більш не приймає участь у кроці 4.

1. Адаптація приклада з найбільшим значенням відповідності:

*vij*(*t*+1)=*vij*(*t*)*xj*



1. Включення всіх виключених на кроці 6 зразків. Повернення до кроку 2.

***Вхідні сигнали*** в цій моделі бінарні.

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

***Ємність мережі*** збігається з числом нейронів другого прошарку і може збільшуватися в процесі функціонування мережі.

***Області застосування.*** Розпізнавання образів, кластерний аналіз.

***Недоліки****.* Необмежене збільшення числа нейронів у процесі функціонування мережі. У присутності шуму виникають значні проблеми, зв'язані з неконтрольованим ростом числа зразків.

***Переваги****.* Навчання без учителя.

***Модифікації.*** Існує модель ART-2 з аналоговими значеннями вхідними сигналами.

**Тема 8. Нейромережі в задачах відображення**

* [Типи задач відображення і підходи до їх вирішення](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#8_1)
  + [Факторний аналіз](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#8_1_1)
  + [Кореляційний аналіз](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#8_1_2)
  + [Ранжування входів](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#8_1_3)
* [Вибір даних для обробки](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#8_2)
  + [Згладжування даних](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#8_2_1)
* [Задачі прогнозування](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#prognoz)
* [Адаптація нейромереж в режимах прогнозування](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#adaptacija) 
  + [Однопараметрична задача прогнозування](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#odnoparametre)
  + [Багатопараметрична задача прогнозування](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#bagatoparametre)
  + [Однокрокове прогнозування (передбачення)](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#odnokrok)
  + [Багатокрокове прогнозування](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#bagatokrok)
  + [Багатокрокове прогнозування з перенавчанням нейромережі на кожному кроці прогнозу](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#bagatokrok_perenavchannja)
* [Критерії оцінки якості функціонування мережі](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#kryterii)
* [Оцінювання точності прогнозів](http://www.victoria.lviv.ua/html/oio/html/theme8.htm#ocinuvannja)

**Типи задач відображення і підходи до їх вирішення**

Відображення розглядають як функцію f, визначену на множині Х, яка приймає свої значення серед елементів множини Y і кожному елементу з Х має відповідати один і лише один елемент з Y. Пару векторів X и Y називають прикладом або реалізацією.

В задачах відображення нейромережі здійснюють оцінювання та передбачення поведінки об'єктів, в тому числі систем та процесів, що підлягають певним законам і можуть бути задані сукупністю своїх реалізацій.

Кожна реалізація повинна містити набір ознак, які визначають основний зміст об'єкта. Якщо однією з ознак об'єкта дослідження є час, тоді реалізації можуть бути представлені у вигляді часових рядів.

В більшості реальних об'єктів дослідження можливо виділити їх основні складові:

* детермінована складова, яка в принципі підлягає точному передбаченню;
* ймовірнісна складова, яку можна передбачити з заданим ступенем ймовірності;
* чисто випадкова складова, яку неможливо ні врахувати, ні передбачити.

В залежності від ступеня впливу тієї чи іншої складової, можна говорити про певний тип множин даних, що використовується для навчання нейромереж:

* множина даних детермінована, з усіма врахованими основними параметрами, викликаними дією відомих причин, яка характеризується малим рівнем шумів;
* множина даних з наявністю ймовірнісної складової, що випливає з експериментальної постановки задачі, з різним ступенем врахування діючих факторів та з впливом похибок оцінювання;
* множина даних з наявністю чисто випадкової складової внаслідок не врахування ряду визначальних ознак явища.

Такий поділ слід вважати приблизним, але при оцінюванні об'єкта дослідження потрібно обирати такі ознаки, для яких можливе зменшення чисто випадкової складової, оцінювання ймовірнісної складової і максимальне збільшення детермінованої частини.

Задачі відображення можна розбити на два основних класи:

* класифікація
* регресія

У **задачах класифікації** потрібно визначити, до якого з декількох заданих класів належить даний вхідний набір. Прикладами можуть служити надання кредиту, діагностика захворювань, розпізнавання образів.

У **задачах регресії** потрібно передбачати значення змінної, що приймає неперервні числові значення: ціна акцій, витрата палива в автомобілі, прибуток кампанії і т.п.

Передбачення явищ можна поділити на:

* передбачення відгуків для множини дискретних вхідних даних, не пов'язаних із часом (економічні, соціологічні оцінки та ін.); дані представлені таблично;
* прогнозування явищ, які безперервно змінюються у часі (фізичні процеси, природні явища, тощо); дані представлені у вигляді часових рядів.

Для вирішення задачі за допомогою нейронної мережі, необхідно зібрати дані для навчання. Навчальна множина даних являє собою набір прикладів, для яких відомо значення вхідних і вихідних параметрів.

Перше, що потрібно вирішити, - які параметри використовувати і скільки прикладів вибрати.

Початково, вибір параметрів здійснюється інтуїтивно. Досвід роботи в обраній предметній області допоможе визначити, які змінні є важливими. Для початку має сенс включити всі змінні, які, на Вашу думку, можуть впливати на результат - на наступних етапах цю множину можна скоротити.

Для забезпечення обґрунтованого вибору, вилучення несуттєвих ознак, що вносять додаткові спотворення при навчанні, можливе застосування відомих методів математичної статистики.

**Факторний аналіз**

Внесок кожної вхідної ознаки можна оцінити також за її впливом на середнє значення вихідної величини. Нехай зовнішній вихід моделі НМ залежить від декількох факторів

*y*=*f*(*a*1*x*1, *a*2*x*2, ..., *aixi*...)

Оберемо деякий фактор *аіхі*. Для всіх реалізацій навчальної множини визначимо значення вихідної величини при наявності та відсутності цього фактору. Обчислимо дисперсію, викликану відсутністю фактора *аіхі*.

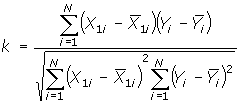
http://www.victoria.lviv.ua/html/oio/images/theme8/image1437.gif

де *Yxі\**, *Yxі* - відповідно значення середньої величини при відсутності та наявності фактора *аіхі*.

Визначаємо інтервал *a*I=± 2*Sai*, в який не повинна попадати оцінка коефіцієнтів *аі*. При малих коефіцієнтах даний фактор вилучається.

**Кореляційний аналіз**

Деякі з параметрів, що приймаються до уваги, справляють незначний вплив на формування виходів і можуть бути відкинуті. В якості показника взаємозалежності між системою вхідних величин *X*=(*X*1, *X*2, ..., *X*n) і вихідних величин *Y* , можна вибрати коефіцієнт парної кореляції (наприклад вхідної змінної *X*1, та вихідного значення *Y*)

,

де http://www.victoria.lviv.ua/html/oio/images/theme8/image1440.gif; http://www.victoria.lviv.ua/html/oio/images/theme8/image1441.gif; *N* - число реалізацій.

Значення *k* < 0,6 вважають за порогове. Наприклад, при знаходженні коефіцієнтів кореляції між виходом та входами можна визначити ступінь впливу кожного вхідного параметру на вихід, і використати даний показник для ранжування входів.

**Ранжування входів**

При проведенні прогнозування, суттєвим для якості прогнозу є врахування реального впливу кожного параметра входу *x*(*х*1, ..., *хn*) на вихідний вектор *y*. За допомогою кореляційного аналізу обчислюються заздалегідь коефіцієнти парної кореляції між виходом *y* та кожним з параметрів входу *х*1, ..., *хj*, ..., *хn*, що дозволяє сформувати вхідну матрицю згідно ступеня впливу кожного параметра і дозволяє застосувати принцип ранжування входів, який узгоджується з будовою біологічного нейрона. В нейромережу вводиться єдиний параметр для всіх входів мережі - коефіцієнт зважування *Kf*, який може приймати значення в діапазоні від 0 до 1.

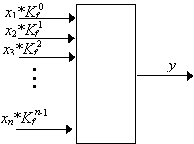


Рис. 1. Вплив коефіцієнта зважування входів.

Для 1 входу всі значення параметра *х*1 не змінюються, для 2 входу зменшуються в *Kf*1 разів, а для останнього *n*-го входу вага параметра *xn* зменшується в *Kfn*-1 разів (рис. 1). При *Kf*=1 всі входи рівнозначні, при *Kf*=0 враховується лише перший вхід, решта входів ігнорується, при 0<*Kf*<1 зменшується вплив несуттєвих параметрів на вихідну величину *y*.

Такий підхід вимагає проведення попереднього аналізу інформації, але значне покращення точності прогнозу підтверджує його ефективність.

**Вибір даних для обробки**

Всяка нейронна мережа приймає на вході числові значення і видає на виході також числові значення. Передатна функція для кожного елемента мережі звичайно вибирається таким чином, щоб її вхідний аргумент міг приймати довільні значення, а вихідні значення лежали б у строго обмеженому діапазоні. При цьому, хоча вхідні значення можуть бути будь-якими, виникає ефект насичення, коли елемент виявляється чуттєвим лише до вхідних значень, що лежать у деякій обмеженій області (наприклад, сигмоїдні або S - функції). У цьому випадку вихідне значення завжди буде лежати в інтервалі (0,1), а область чутливості для входів ледь ширше інтервалу (-1,+1). Дана функція є гладкою, а її похідна легко обчислюється - ця обставина дуже істотна для роботи алгоритму навчання мережі (у цьому також криється причина того, що порогова функція для цієї мети практично не використовується).

При використанні нейронних мереж можуть виникати деякі проблеми, зокрема:

* дані мають нестандартний масштаб,
* дані є нечисловими,
* в даних є пропущені або недостовірні значення.

Числові дані масштабуються в придатний для мережі діапазон. Звичайно дані масштабуються по лінійній шкалі. У пакетах програмних нейромереж реалізовані алгоритми, що автоматично знаходять масштабуючі параметри для перетворення числових значень у потрібний діапазон.

Більш важкою задачею є робота з даними нечислового характеру. Нехай, потрібно навчити нейромережу оцінювати вартість об'єктів нерухомості. Ціна будинку залежить від того, у якому районі міста він розташований. Місто може бути поділено на кілька десятків районів, що мають власні назви, і здається природним увести для позначення району змінну з номінальними значеннями. На жаль, у цьому випадку навчити нейронну мережу буде дуже важко, і замість цього краще привласнити кожному району визначений ранг (ґрунтуючись на експертних оцінках).

Найчастіше нечислові дані бувають представлені у виді номінальних змінних. Номінальні змінні можуть бути двозначними (наприклад, *Стать* ={*Чоловік*, *Жінка*}) або багатозначними (тобто приймати більше двох значень станів). Двозначну номінальну змінну легко перетворити в числову (наприклад, *Чоловік* = 0, *Жінка* = 1). З багатозначними номінальними змінними справа обстоїть складніше. Їх теж можна представити одним числовим значенням (наприклад, *Собака* = 0, *Миша* = 1, *Кішка* = 2), однак при цьому виникне (можливо) помилкове впорядкування значень номінальної змінної: у розглянутому прикладі *Миша* виявиться чимось середнім між *Собакою* і *Кішкою*. Існує більш точний спосіб, відомий як кодування 1-из-N, в якому одна номінальна змінна представляється декількома числовими змінними. Кількість числових змінних дорівнює числу можливих значень номінальної змінної; при цьому всякий раз рівно одна з *N* змінних приймає ненульове значення (наприклад, *Собака* = {1,0,0}, *Миша* = {0,1,0}, *Кішка* = {0,0,1}). На жаль, номінальна змінна з великим числом можливих станів потребує при кодуванні методом 1-из-N дуже великої кількості числових змінних, а це приводить до росту розмірів мережі і створює труднощі при її навчанні. В таких ситуаціях краще спробувати знайти інший спосіб представлення даних.

Нечислові дані інших типів можна або перетворити в числову форму, або оголосити незначними. Значення дат і часу, якщо вони потрібні, можна перетворити в числові, віднімаючи з них початкову дату (час). Позначення грошових сум перетворити зовсім нескладно. З довільними текстовими полями (наприклад, прізвищами людей) працювати не можна і їх потрібно зробити незначними.

У багатьох реальних задачах приходиться мати справу з не зовсім достовірними даними. Значення деяких змінних можуть бути спотворені шумом чи частково бути відсутніми. Існують спеціальні засоби роботи з пропущеними значеннями (вони можуть бути замінені на середнє значення цієї змінної чи на інші її статистики), так що якщо даних не багато, можна включити в розгляд випадки з пропущеними значеннями. Нейронні мережі у цілому стійкі до шумів. Однак у цієї стійкості є межа. Наприклад, викиди, тобто значення, що лежать дуже далеко від області нормальних значень деякої змінної, можуть спотворити результат навчання. У таких випадках найкраще постаратися знайти і виявити ці викиди (вилучити відповідні приклади або перетворити викиди в пропущені значення). Якщо викиди виявити важко, то можна скористатися можливостями зробити процес навчання стійким до викидів, однак таке стійке до викидів навчання, як правило, менш ефективно, ніж стандартне.

**Згладжування даних**

Позитивний ефект надається при використанні додаткової нейромережі, що функціонує в режимі згладжування вхідних даних навчальної множини. В режимі навчання додаткової мережі кожна реалізація навчальної множини набуває вигляду: вектор вхідних значень вектор вхідних значень (рис. 3).

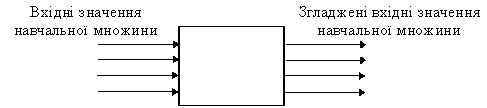


Рис. 3. Приклад застосування нейромережі для згладжування даних

В режимі функціонування на входи подаються вхідні значення навчальної множини, на виході отримуємо згладжені значення, без наявних викидів, які в подальшому можна використовувати для опрацювання. Можна дати наступне пояснення ефекту згладжування даних. Залежність вихідних значень нейромережі від вхідних може бути представлена сумарним степеневим поліномом, так як передатні функції нейронів прихованого шару - поліноміальні. При незначному числі нейронів прихованого шару і невисоких степенях поліномів сумарний поліном буде невисокого степеня, що не дає можливості відтворювати викиди, тобто приводить до згладжуваного відтворення.

Питання про те, скільки прикладів потрібно мати для навчання мережі, часто виявляється непростим. Відомо ряд правил, що погоджують число необхідних прикладів з розмірами мережі (найпростіше з них говорить, що число прикладів повинне бути в десять разів більше числа зв'язків у мережі). Насправді це число залежить також від складності того відображення, що нейронна мережа прагне відтворити. З ростом кількості параметрів кількість необхідних прикладів росте нелінійно, так що вже при досить невеликому числі параметрів може знадобитися величезне число прикладів.

Для більшості реальних задач буває досить декількох сотень чи тисяч прикладів. Для особливо складних задач може знадобитися ще більша кількість, однак дуже рідко може зустрітися задача, де вистачило б менш сотні прикладів. Якщо даних менше, то інформації для навчання мережі недостатньо.

**Задачі прогнозування**

Особливе значення мають задачі передбачення та прогнозування часових рядів, серед яких виділяються завдання з набором певних специфічних ознак, тому варто провести їх класифікацію. Задачі дослідження явищ, розвиток яких пов'язаний із часом, можна поділити на декілька класів:

**За характером основних ознак об'єкту:**

* прогнозування явищ, реалізації яких представлені у вигляді детермінованих часових рядів. Такі задачі, зокрема, можна вирішити шляхом застосування методів математичного аналізу;
* прогнозування явищ, реалізації яких представлені у вигляді індетермінованих часових рядів. Вирішення цих задач традиційно здійснюється шляхом застосування методів теорії ймовірностей та математичної статистики. Зокрема, реалізації таких явищ, можуть мати вигляд:

а) стаціонарного часового ряду, який характеризується однорідністю в часі, без суттєвих змін характеру коливань та їх середньої амплітуди; вибір проміжку для формування навчальної множини довільний; як приклад такого ряду на рис. 4 наведений графік сумарного річного стоку Дніпра за період з 1810 до 1964 року;

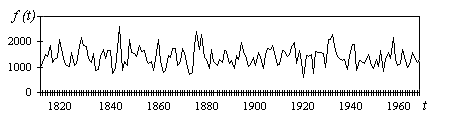


Рис. 4. Розподіл річного стоку Дніпра в часі

б) нестаціонарного часового ряду, який характеризується певною тенденцією розвитку в часі (рис. 2); при дослідженні нестаціонарних процесів можна виділити ділянки, на яких процес можна вважати стаціонарним; вибір проміжку для формування навчальної множини в такому випадку обирається згідно задачі прогнозування;

На рис. 5 наведені щоденні нормовані дані мікросейсмічних коливань Землі за певний період часу.

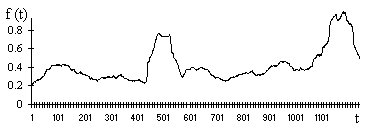


Рис. 5. Розподіл мікросейсмічних коливань Землі в часі

**За числом ознак об'єкту досліджень:**

* одновимірна задача; явище представлене лише однією ознакою, зміни якої відбуваються в часі; на рис. 3 зображені дані спостережень відносних чисел Вольфа, усереднені за місяць, за період з 1900 по 1924 рік;

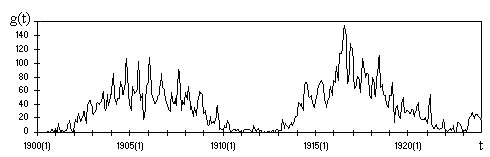


Рис. 6. Розподіл чисел Вольфа в часі

* багатовимірна задача; об'єкт або явище представлені кількома ознаками (рис. 7); задача прогнозування може бути розширена завдяки представленню даних в просторі (рис. 8).

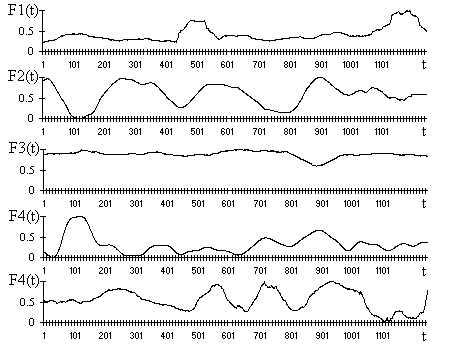


Рис. 7. Основні передвісники сейсмічної активності

На рис. 7 представлені щоденні дані деяких з основних ознак, що визначають землетрус. Показані дані, зокрема: *F*1 - акоїстика, *F*2 - деформації земної поверхні, *F*3 - мікросейсмічні коливання Землі, *F*4 - сумарна енергія землетрусу, *F*5 - температура Землі, що складають ознаки, були виміряні в однакові відліки часу за певний період і надалі використовувались для експериментів.

Як приклад багатовимірної задачі покажемо оцінювання числа груп сонячних плям, усереднених за рік, що визначаються двовимірним розподілом в області двох аргументів - широта-час (рис. 8).

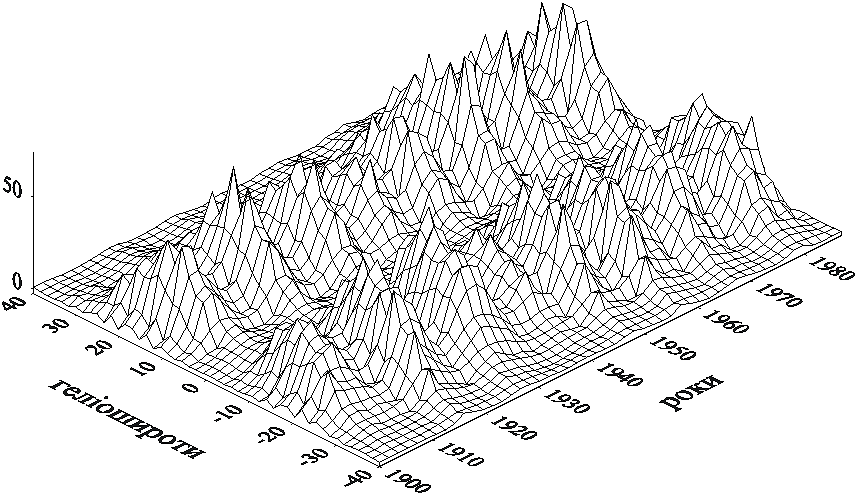


Рис. 8. Просторове відображення кількості сонячних плям в координатах геліоширота-час.

Враховуючи специфічний характер прогнозування часових рядів та певний різнобій в термінології, дотримуватимемося ряду визначень.

*Передісторією* ряду назвемо набір елементів часового ряду, який враховується для одного кроку прогнозування наступних елементів часового ряду. Однокрокове прогнозування зводиться до задач відображення у випадку, коли значення елементів передісторії можуть визначати лише один дискретний відлік вихідних величин. Багатокрокове прогнозування характеризується збільшенням дискретних відліків вихідної величини і, відповідно, збільшенням часу, на який здійснюється прогноз (час випередження *Твип*). При багатокроковому прогнозуванні *Твип*=*а*\**R*, де *R* - кількість кроків обчислення прогнозування; *а* - крок дискретизації вихідного параметра (наприклад, рік, місяць, день, тощо).

**За часом випередження розрізняють види прогнозів:**

* згладжування, *R*= 0;
* короткотерміновий прогноз, *R*= 1  2;
* середньотерміновий прогноз, *R*= 3  7;
* довготерміновий прогноз, *R*= 10  15.

Очевидно, що вид прогнозу суттєво впливає на вибір засобів і методику його реалізації.

**Адаптація нейромереж в режимах прогнозування**

Дані про поведінку об'єкта, ознаки якого пов'язані з часом, представлені як результати спостережень в рівномірні відліки часу. Для моментів часу *t*=1, 2, ..., *n* дані спостережень набувають вигляду часового ряду *х*(*t*1), *х*(*t*2), ..., *х*(*tn*). Інформація про значення часового ряду до моменту *n* дозволяє давати оцінки параметрів *x*(*tn*+1), *x*(*n*+2), ..., *x*(*n*+*m*). Для здійснення прогнозування елементів часових рядів широко використовують так званий метод "часових вікон".

В залежності від кількості ознак, що представляють значення рядів при формуванні множин даних, виділимо задачі двох типів.

**Однопараметрична задача прогнозування**

Нехай часовий ряд *x*(*t*) задано відліками процесу *x*(*t*1), *x*(*t*2),..., *x*(*tі*) в дискретні моменти часу *t*. Задамо ширину (кількість дискретних відліків) вхідного часового вікна *m*, ширину вихідного вікна *р*. Вхідне та вихідне вікна накладаються на дані ряду, починаючи з першого елемента (рис. 9).

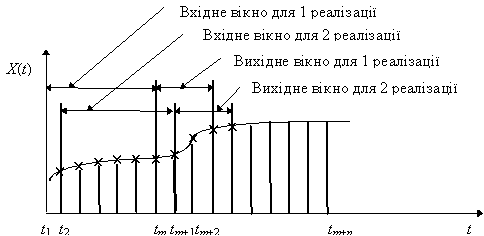


Рис. 9. Формування множин даних для однопараметричної задачі за методом "часових вікон"

Вхідне вікно формує дані для входів нейронної мережі, а вихідне, відповідно, для виходів. Подібна пара вхідного та вихідного векторів приймається за одну реалізацію часового ряду. При зсуві часових вікон за часовим рядом з кроком *s*, отримуємо другу і наступні реалізації.

Значення ширини вікон та кроку зміщення повинні узгоджуватись з особливостями часового ряду, що забезпечується шляхом проведення експериментів. Нехай вхідне вікно має ширину *m*, вихідне вікно *р*=1, крок зміщення *s*=1. Тоді сформована множина значень для однопараметричної задачі матиме вигляд, наведений нижче:

Таблиця 1.

**Множина даних для однопараметричної задачі**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Входи | | | | Виходи |
| *x*(*t*1) | *x*(*t*2) | ... | *x*(*tm*) | *x*(*tm*+1) |
| *x*(*t*2) | *x*(*t*3) | ... | *x*(*tm*+1) | *x*(*tm*+2) |
| *x*(*t*3) | *x*(*t*4) | ... | *x*(*tm*+2) | *x*(*tm*+3) |
| ... | ... | ... | ... | ... |
| *x*(*tі*) | *x*(*tі*+1) | ... | *x*(*ti*+*m*-1) | *x*(*ti*+*m*) |

**Багатопараметрична задача прогнозування**

В багатопараметричних задачах прогнозування підходи до розв'язання проблеми залишаються подібними (рис.10).

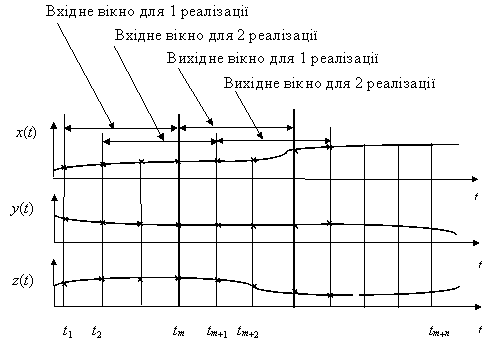


Рис. 10. Формування множин даних для багатопараметричної задачі

Нехай потрібно спрогнозувати взаємозалежні величини *x*(*t*), *y*(*t*), ..., *z*(*t*). Якщо прийняти ширину вхідного вікна *m*, вихідного *р*=1, кроку зміщення *s*=1, можна сформувати множину даних наступним чином:

Таблиця 2

**Множина даних для багатопараметричної задачі**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Входи | | | | | | | | |
| *x*(t1) | *x*(*t*2) | *x*(*t*m) | *y*(*t*1) | *y*(*t*2) | *y*(*t*m) | *z*(*t*1) | *z*(*t*2) | *z*(*t*m) |
| *x*(*t*2) | *x*(*t*3) | *x*(*t*m+1) | *y*(*t*2) | *y*(*t*3) | *y*(*t*m+1) | *z*(*t*2) | *z*(*t*3) | *z*(*t*m+1) |
| *x*(*t*3) | *x*(*t*4) | *x*(*t*m+2) | *y*(*t*3) | *y*(*t*4) | *y*(*t*m+2) | *z*(*t*3) | *z*(*t*4) | *z*(*t*m+2) |
| *x*(*ti*) | *x*(*ti*+1) | *x*(*ti*+m-1) | *y*(*ti*) | *y*(*ti*+1) | *y*(*ti*+m-1) | *z*(*ti*) | *z*(*ti*+1) | *z*(*ti*+m-1) |

|  |  |  |  |
| --- | --- | --- | --- |
| Виходи | | | |
| *x*(*t*m+1) | *y*(*t*m+1) | *z*(*t*m+1) |  |
| *x*(*t*m+2) | *y*(*t*m+2) | *z*(*t*m+2) |  |
| *x*(*t*m+3) | *y*(*t*m+3) | *z*(*t*m+3) |  |
| *x*(*ti*+m-1) | *y*(*ti*+m-1) | *z*(*ti*+m-1) |  |

Функціонування нейромережі здійснюється у відповідності з показаним методом часових вікон, зберігаючи значення ширини вікон та кроку зсуву.

Конкретизація підходів до реалізації прогнозування в значній мірі залежить також від особливостей явища, що досліджується.

**Однокрокове прогнозування (передбачення)**

Задача однокрокового прогнозування зводиться до задачі відображення, коли один вхідний вектор відображається у вихідний (рис. 11).

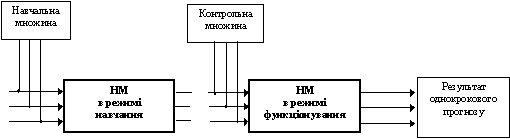


Рис. 11. Послідовність використання нейромереж для задач передбачення

У випадку однопараметричної задачі передбачення навчальна множина до моменту *n*, за умови *m*=3, *p*=1, *s*=1, матиме вигляд наведений в таблиці 3.

Таблиця 3

|  |  |  |  |
| --- | --- | --- | --- |
| Входи | | | Вихід |
| *x*(*t*1) | *x*(*t*2) | *x*(*t*3) | *x*(*t*4) |
| *x*(*t*2) | *x*(*t*3) | *x*(*t*4) | *x*(*t*5) |
| ... | ... | ... | ... |
| *x*(*tn*-3) | *x*(*tn*-2) | *x*(*tn*-1) | *x*(*tn*) |

В режимі навчання встановлюються коефіцієнти ваг зв'язків, після чого стає можливим перехід до режиму функціонування. Для передбачення на входи нейромережі надходять значення останньої реалізації навчальної множини *x*(*tn*-2), *x*(*tn*-1), *x*(*tn*). На виході формується прогнозована величина *x*\*(*tn*+1).

Для багатопараметричної задачі передбачення на входи навченої нейромережі подаються вектори *x*(*tn*-2), *y*(*tn*-2), *z*(*tn*-2), *x*(*tn*-1), *y*(*tn*-1), *z*(*tn*-1), *x*(*tn*), *y*(*tn*), *z*(*tn*). На виходи нейромережі надходять передбачені величини *x*\*(*tn*+1), *y*\*(*tn*+1), *z*\*(*tn*+1), які відкладаються у вихідний вектор передбачених даних.

Показаний режим є однокроковим, який працює в режимі відображення (реальний вхідпрогнозований вихід). Передбачення застосовують також для моделювання дискретних послідовностей, що не пов'язані з часом. Враховуючи специфіку часових рядів, такий тип прогнозу не завжди є доцільним, але для певних випадків короткотермінових прогнозів ним можливо скористатись.

**Багатокрокове прогнозування**

Багатокрокове прогнозування застосовують лише для явищ, ознаки яких представлені у вигляді часових рядів.

Для однопараметричної задачі прогнозування навчальна множина матиме вигляд наведений в табл. 3. Під час навчання мережа налаштовує коефіцієнти ваг зв'язків і поліномів передатних функцій, які в подальшому і визначають режим функціонування. Багатокрокове прогнозування часового ряду здійснюється наступним чином (рис. 12). На входи нейромережі подається вектор відомих значень *x*(*tn*-2), *x*(*tn*-1), *x*(*tn*). На виході формується прогнозована величина *x*\*(*tn*+*1*), яка визначає вектор прогнозованих виходів і одночасно долучається до значень навчальної множини, тобто, приймається як достовірна. Далі на входи подається вектор *x*(*tn*-1), *x*(*tn*), *x*\*(*tn*+1), а на виході отримується *x*\*(*tn*+2) і наступні прогнозовані значення.

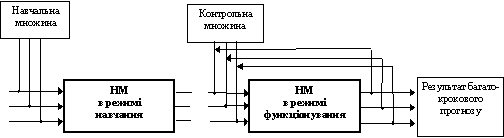


Рис. 12. Послідовність використання НМ для задач багатокрокового прогнозування

Для багатопараметричної задачі прогнозування на входи навченої нейромережі подаються вектори *x*(*tn*-2), *y*(*tn*-2), *z*(*tn*-2), *x*(*tn*-1), *y*(*tn*-1), *z*(*tn*-1), *x*(*tn*), *y*(*tn*), *z*(*tn*). На виході продукуються величини *x*\*(*tn*+1), *y*\*(*tn*+1), *z*\*(*tn*+1), які формують вектор вихідних значень і послідовно долучаються до значень навчальної множини. При зсуві вікна на крок прогнозу вихідні дані, що були спродуковані мережею, сприймаються як реальні і приймають участь у прогнозуванні наступного значення виходу, тобто на входи подаємо вектор *x*(*tn*-1), *y*(*tn*-1), *z*(*tn*-1), *x*(*tn*), *y*(*tn*), *z*(*tn*), *x*\*(*tn*+1), *y*\*(*tn*+1), *z*\*(*tn*+1), а на виході отримуємо *x*\*(*tn*+2), *y*\*(*tn*+2), *z*\*(*tn*+2) і наступні прогнозовані значення.

Багатокрокове прогнозування дозволяє робити коротко- та середньотермінові прогнози, оскільки суттєвий вплив на точність має накопичення похибки на кожному кроці прогнозування. При застосуванні довготермінового багатокрокового прогнозування спостерігається характерне для багатьох прогнозуючих систем поступове затухання процесу, фазові зсуви і інші спотворення картини прогнозу. Такий тип прогнозування підходить для часових рядів, які підпадають під означення стаціонарного процесу з невеликою випадковою складовою.

**Багатокрокове прогнозування з перенавчанням нейромережі на кожному кроці прогнозу**

Швидкі неітераційні алгоритми навчання дозволяють запропонувати новий тип багатокрокового прогнозу, який може бути застосований при довготермінових прогнозах із збереженням задовільної точності прогнозування.

Аналогічно з попереднім алгоритмом прогнозування на входи мережі у режимі функціонування надходить остання реалізація навчальної множини *x*(*tn*-2), *x*(*tn*-1), *x*(*tn*). Прогнозоване значення виходу *x*\*(*tn*+1) відкладається у векторі прогнозованих вихідних значень і в якості достовірного додається до реальних значень навчальної множини. Навчальна множина збільшується на одне часове вікно. Відбувається процес перенавчання мережі на збільшеній навчальній множині, під час якого визначаються нові вагові коефіцієнти *k* синаптичних зв'язків і поліномів передатних функцій нейронів (рис. 13).

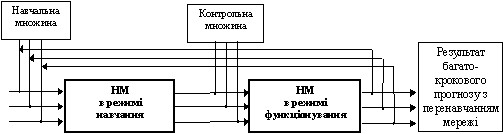


Рис. 13. Послідовність використання нейромережі для задач багатокрокового прогнозування з перенавчанням

Реалізація *x*(*tn*-1), *x*(*tn*), *x*\*(*tn*+1), як значення наступного вхідного вікна подається на входи мережі в режимі функціонування. Мережа продукує нове вихідне значення *x*\*(*tn*+2), яке відповідно також відкладається у вектор продукованих виходів і долучається до реальних значень навчальної множини, з метою подальшого перенавчання мережі та встановлення поновлених коефіцієнтів поліномів передатних функцій і синаптичних зв'язків. Ітераційна процедура перенавчання поширюється до прогнозованого значення *x*\*(*tN*).

Такий підхід дозволяє при великих інтервалах випередження усунути затухання прогностичних властивостей мережі за рахунок постійного коректування вагових коефіцієнтів синаптичних зв'язків.

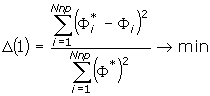
Відзначимо, що алгоритм багатокрокового прогнозування з перенавчанням мережі для традиційних мереж прямого поширення з ітераційним навчанням є практично нездійсненним через великі часові затримки, необхідні на переналаштовування коефіцієнтів мережі.

**Критерії оцінки якості функціонування мережі**

Критерії оцінки слід поділити на внутрішні та зовнішні. Внутрішні критерії формуються на основі інформації множини даних, які були використані для навчання, тоді як зовнішні критерії використовують нову інформацію тестової множин, елементи яких не використовувалась при навчанні. Оптимальна складність моделі мережі встановлюється за сукупністю зовнішніх та внутрішніх критеріїв.

До зовнішніх критеріїв вибору моделі можна віднести:

**Критерій регулярності** - зовнішній критерій, для обчислення величини якого вимагається тестова множина даних.



де  (1) - величина критерію; *і* - номер відгуку; *і* - значення відгуків, що продукуються мережею; *і*\* - точні значення відгуків.

Фізичний сенс застосування критерію регулярності полягає у виборі моделі, яка буде максимально точною на елементах тестової множини, які не входили до складу навчальної множини.

**Критерій мінімального зміщення** вимагає максимального співпадіння значень вихідної величини для двох моделей, де в якості навчальних елементів були використані дані різних підмножин навчальної множини.

Критерій мінімального зміщення дозволяє обирати модель, яка "слабо реагує" на зміну навчальної множини і дозволяє вирішити задачу відновлення закону, що діє для зашумлених тестових даних.

**Критерій зміщенності показників в часі**- допомагає оцінити рівень взаємозв'язку змінних. Окремі показники можуть мати різну післядію, тому розділене прогнозування кожного з них може забезпечити кращий результат. Відповідно, при наявності тісного взаємозв'язку між показниками, прогнозування їх сукупності покращує результати прогнозу кожного з них. Можливе також включення нового параметра, який може бути або додатковою ознакою або лінійною комбінацією вже включених ознак. Застосування цього критерію допомагає в оптимальному підборі таких ознак явища, які можуть забезпечити вищу точність прогнозування.

**Критерій фізичної достовірності**- вимагає виключення моделей, які під час проведення експерименту, можуть продукувати нереальні результати (великі викиди для множини, що прогнозується).

**Оцінювання точності прогнозів**

Як правило, після навчання нейромережі здійснюють контрольне відтворення даних, які складали навчальну множину. Якщо точність відтворення задовільна і відхилення знаходяться в допустимих межах, вважають, що побудовано задовільну модель і слід очікувати достатню якість відображення. Якщо при відтворенні мережею даних навчальної множини спостерігаються великі розбіжності, можна припустити що це викликано:

* наявністю неточних даних з великою випадковою складовою. Для усунення цього явища підвищують вимоги до точності вимірювань; у випадку часового ряду, можливе зменшення кроку дискретизації, наприклад використання щомісячних значень замість річних;
* неврахуванням суттєвих ознак, які в значній мірі визначають закономірність; ця проблема може бути вирішена розширенням набору ознак, які приймаються до уваги;

Після отримання передбачених значень при наявності правильних можливо отримати абсолютні та відносні відхилення на всій контрольній множині, для кожного кроку прогнозування. При наявності задовільних результатів прогнозування на контрольній множині, можна вважати, що налаштована мережа для даної задачі має оптимальну складність і готова до відтворення даних, для яких немає відповідних відомих відгуків.

|  |  |
| --- | --- |
| |  | | --- | | **Тема 9. Сучасні напрямки розвитку нейрокомп'ютерних технологій**   * [Загальні задачі](http://www.victoria.lviv.ua/html/oio/html/theme9.htm#9_0) * [Прикладні задачі](http://www.victoria.lviv.ua/html/oio/html/theme9.htm#9_1)   + [Обробка зображень](http://www.victoria.lviv.ua/html/oio/html/theme9.htm#9_1_1)   + [Обробка сигналів](http://www.victoria.lviv.ua/html/oio/html/theme9.htm#9_1_2)   + [Системи керування динамічними об'єктами](http://www.victoria.lviv.ua/html/oio/html/theme9.htm#9_1_3) * [Нейромережеві експертні системи](http://www.victoria.lviv.ua/html/oio/html/theme9.htm#9_1_4) * [Нейрочіпи і нейрокомп'ютери](http://www.victoria.lviv.ua/html/oio/html/theme9.htm#9_2) * [Підсумок](http://www.victoria.lviv.ua/html/oio/html/theme9.htm#9_4)   Детальний аналіз розробок нейрокомп'ютерів дозволив виділити основні перспективні напрямки сучасного розвитку нейрокомп'ютерних технологій: нейропакети, нейромережеві експертні системи, СУБД із включенням нейромережевих алгоритмів, обробка зображень, керування динамічними системами й обробка сигналів, керування фінансовою діяльністю, оптичні нейрокомп'ютери, віртуальна реальність. Розробками в цій області займається більш 300 закордонних компаній, причому число їх постійне збільшується. Серед них такі гіганти як Intel, IBM і Motorolla. Сьогодні спостерігається тенденція переходу від програмних реалізацій до програмно-апаратної реалізації нейромережевих алгоритмів з різким збільшенням числа розробок нейрочипів з нейромережевою архітектурою. Різко зросла кількість військових розробок, в основному скерованих на створення надшвидкісних, "розумних" суперобчислювачів.  Якщо говорити про головний перспективний напрямок - інтелектуалізації обчислювальних систем, додавання їм властивостей людського мислення і сприйняття, то тут нейрокомп'ютери - практично єдиний шлях розвитку обчислювальної техніки. Багато невдач на шляху вдосконалення штучного інтелекту протягом останніх 30 років пов'язано з тим, що для рішення важливих і складних по постановці задач вибирались обчислювальні засоби, не адекватні по можливостях розв'язуваній задачі, в основному з числа традиційних комп'ютерів. При цьому, як правило, не вирішувалась задача, а показувалась принципова можливість її рішення. Сьогодні активний розвиток комп'ютерних технологій створив об'єктивні умови для побудови обчислювальних систем, адекватних по можливостях і архітектурі практично будь-яким задачам штучного інтелекту.  У Японії з 1993 року прийнята програма "Real world computing program". Її основна мета - створення еволюціонуючої адаптивної ЕОМ. Проект розрахований на 10 років. Основою розробки є нейротехнологія, яка використовується для розпізнавання образів, обробки семантичної інформації, керування інформаційними потоками і роботами, що здатні адаптуватися до навколишнього оточення. Тільки в 1996 році було проведено біля сотні міжнародних конференцій по нейрокомп'ютерах і суміжних проблемах. Розробки нейрокомп'ютерів ведуться в багатьох країнах світу, зокрема, в Австралії створений свій зразок комерційного супернейрокомп'ютера.  Завжди звучить питання: для якого класу задач найбільш ефективне застосування того чи іншого обчислювального пристрою, побудованого по нових ознаках. Стосовно нейрокомп'ютерів відповідь на нього постійно міняється протягом уже майже 50 років.  В історії обчислювальної техніки завжди були задачі, не розв'язувані традиційними комп'ютерами з архітектурою фон Неймана і для них перехід до нейромережевих технологій характерний у випадку різкого збільшення розмірності простору або рішення необхідності різкого скорочення часу. Можна виділити три ділянки застосування нейромережевих технологій: загальна, прикладна і спеціальна.  **Загальні задачі**  Це задачі досить просто зводяться до обробки нейронною мережею багатовимірних векторів дійсних змінних, наприклад:   * контроль кредитних карток. Сьогодні 60% кредитних карток у США обробляються за допомогою нейромережевих технологій; * система виявлення схованих речовин за допомогою системи на базі теплових нейронів і за допомогою нейрокомп'ютера на замовлених цифрових нейрочипах. Подібна система фірми SAIC експлуатується вже в багатьох аеропортах США при огляді багажу для виявлення наркотиків, вибухових речовин, ядерних і інших матеріалів; * система автоматизованого контролю безпечного збереження ядерних виробів.   **Прикладні задачі**  **Обробка зображень**  Найбільш перспективними задачами обробки зображень нейрокомп'ютерами є обробка аерокосмічних зображень (стиснення із відновленням, сегментація, обробка зображень), пошук, виділення і розпізнавання на зображенні рухомих об'єктів заданої форми, обробка потоків зображень, обробка інформації у високопродуктивних сканерах.  **Обробка сигналів**  У першу чергу це клас задач, зв'язаних із прогнозуванням часових залежностей:   * прогнозування фінансових показників; * прогнозування надійності електродвигунів; * передбачення потужності АЕС і прогнозування надійності систем електроживлення на літаках;   При рішенні цих задач спостережується перехід від найпростіших регресійних і інших статистичних моделей прогнозу до істотно нелінійних адаптивних екстраполюючих фільтрів, реалізованих у вигляді складних нейронних мереж.  **Системи керування динамічними об'єктами**  Це одна із самих перспективних, областей застосування нейрокомп'ютерів. В США і Фінляндія ведуть роботи з використання нейрокомп'ютерів для керування хімічними реакторами. В країнах СНД цим не займалися, зокрема, через моральне застаріння існуючих реакторів і недоцільності вдосконалення їхніх систем керування.  Перспективної вважається розробка нейрокомп'ютера для керування рухливою установкою гіперзвукового літака. Доцільним для вирішення за допомогою нейрокомп'ютера є задача навчання нейронної мережі виробленню точного маневру винищувача, задача керування роботами: пряма, зворотна кінематичні і динамічні задачі, планування маршруту руху робота. Перехід до нейрокомп'ютерів тут зв'язаний у першу чергу з обмеженістю обсягів розміщення обчислювальних систем, а також з необхідністю реалізації ефективного керування в реальному масштабі часу.  **Нейромережеві експертні системи**  Необхідність реалізації експертних систем за алгоритмами нейромереж виникає при значному збільшенні числа правил і висновків. Прикладами реалізації конкретних нейромережевих експертних систем можуть служити система вибору повітряних маневрів у ході повітряного бою і медична діагностична експертна система для оцінки стану льотчика.  **Нейрочіпи і нейрокомп'ютери**  Головний результат розробки нейромережевих алгоритму рішення задачі - можливість створення архітектури нейрочіпу, адекватного розв'язуваній задачі. Для реалізації нейромережевих алгоритмів з використанням універсальних мікропроцесорних засобів ефективніше створити архітектури, орієнтовані на виконання нейромережевих операцій, ніж використовувати стандартні, орієнтовані на модифікацію однопроцесорних алгоритмів рішення задач.  На відміну від інших напрямків розвитку надпродуктивної обчислювальної техніки нейрокомп'ютери дають можливість вести розробки з використанням наявного потенціалу електронної промисловості. Необхідно відзначити ряд важливих особливостей даних робіт:   * цей напрямок дозволяє створити унікальні суперкомп'ютери на наявній елементній базі; * розробки нейрочіпів і нейрокомп'ютерів характеризуються переходом від цифрової обробки до аналого-цифрової і аналогової; * нейромережеві архітектури у порівнянні з іншими приводять до активізації використання нових технологічних напрямків реалізації: нейросистеми на пластмасі, оптоелектронні й оптичні нейрокомп'ютери, молекулярні нейрокомп'ютери і нанонейроелементи; виникає потреба в універсалізації САПР нейрочипів. * народження технології систем на пластмасі і нанотехнології може привести до появи нових надпаралельних архитектур. Починаючи з нанонейроелементів, ми впритул підходимо до інших принципово нових архітектурних елементів, що утворюють надпаралельні високопродуктивні обчислювальні системи.   **Підсумок**  Нейрокомп'ютери є перспективним напрямком розвитку сучасної високопродуктивної обчислювальної техніки, а теорія нейронних мереж і нейроматематика являють собою пріоритетні напрямки обчислювальної науки, і при відповідній підтримці інтенсивно розвиваються.  Основою активного розвитку нейрокомп'ютерів є принципова відмінність нейромережевих алгоритмів рішення задач від однопроцесорних та малопроцессорних.  Нейрокомп'ютери є предметом досліджень відразу декількох дисциплін, тому єдине визначення нейрокомп'ютера можна дати тільки з врахуванням різних точок зору, адекватних різним напрямкам науки.  **Математична статистика.** Нейрокомп'ютери - це системи, що дозволяють сформувати опис характеристик випадкових процесів і сукупності випадкових процесів, що мають складні, багатомодальні чи взагалі невідомі функції розподілу.  **Математична логіка і теорія автоматів.** Нейрокомп'ютери - це системи, в яких алгоритм рішення задачі представлений логічною мережею елементів окремого виду - нейронів з повним відмовленням від булевских елементів типу І, АБО, НІ. Як наслідок цього введені специфічні зв'язки між елементами, що є предметом окремого розгляду.  **Теорія керування.** Як об'єкт керування вибирається окремий випадок, добре формалізований об'єкт - багатошарова нейронна мережа, а динамічний процес її налаштування уявляє собою процес рішення задачі. При цьому практично весь апарат синтезу адаптивних систем керування переноситься на нейронну мережу як окремий вид об'єкта керування.  **Обчислювальна математика.** Нейрокомп'ютери реалізують алгоритми рішення задач, представлені у виді нейронних мереж. Це обмеження дозволяє розробляти алгоритми, потенційно більш паралельні, ніж будь-яка інша їхня фізична реалізація. Множина нейромережевих алгоритмів рішення задач складає новий перспективний розділ обчислювальної математики, умовно названою нейроматематикой.  **Обчислювальна техніка.** Нейрокомп'ютер - це обчислювальна система, в якій реалізовані два принципових технічних рішення:  · спрощений до рівня нейрона процесорний елемент однорідної структури і різко ускладнені зв'язки між елементами; · програмування обчислювальної структури перенесено на змінювання вагових зв'язків між процесорними елементами.  Загальне визначення нейрокомп'ютера може бути представлене в наступному виді.  **Нейрокомп'ютер** - це обчислювальна система з архітектурою апаратного і програмного забезпечення, адекватної виконанню алгоритмів, представлених у нейромережевому логічному базисі.   * Нейрокомп'ютери дають стандартний спосіб рішення багатьох нестандартних задач. І неважливо, що спеціалізована машина краще вирішує один клас задач. Важливіше, що один нейрокомп'ютер вирішить і цю задачу, і другу, і третю і не треба щораз проектувати спеціалізовану ЕОМ, нейрокомп'ютер зробить все сам і майже не гірше. * Замість програмування навчання. Нейрокомп'ютер вчиться, потрібно лише формувати навчальні множини. Праця програміста заміняється новою працею вчителя. Краще це чи гірше? Ні те, ні інше. Програміст наказує машині всі деталі роботи, вчитель створює "навчальне середовище", до якого пристосовується нейрокомп'ютер. З'являються нові можливості для роботи. * Нейрокомп'ютери ефективні там, де потрібний аналог людської інтуїції, зокрема, для розпізнавання образів, читання рукописних текстів, підготовки аналітичних прогнозів, перекладу з однієї природної мови на іншу і т.п. Саме для таких задач звичайно важко скласти явний алгоритм. * Нейронні мережі дозволяють створити ефективне програмне та математичне забезпечення для комп'ютерів з високим ступенем розпаралелювання обробки. * Нейрокомп'ютери "демократичні" та дружні, як текстові процесори, тому з ними може працювати будь-який, навіть зовсім недосвідчений користувач. | |

**Тема 10. Популярно про генетичні алгоритми**

* [Історія появи еволюційних алгоритмів](http://www.victoria.lviv.ua/html/oio/html/theme10.htm#10_1)
  + [Еволюційна теорія](http://www.victoria.lviv.ua/html/oio/html/theme10.htm#10_1_1)
  + [Природний відбір і генетичне спадкування](http://www.victoria.lviv.ua/html/oio/html/theme10.htm#10_1_2)
* [Задачі оптимізації](http://www.victoria.lviv.ua/html/oio/html/theme10.htm#10_2)
* [Робота генетичного алгоритму](http://www.victoria.lviv.ua/html/oio/html/theme10.htm#10_3)
* [Застосовування генетичних алгоритмів](http://www.victoria.lviv.ua/html/oio/html/theme10.htm#10_4)

**Історія появи еволюційних алгоритмів**

Природа вражає своєю складністю і багатством всіх своїх проявів. Серед прикладів можна назвати складні соціальні системи, імунні і нейронні системи, складні взаємозв'язки між видами. Вони - всього лише деякі з чудес, що стали більш очевидні, коли ми стали глибше досліджувати себе самих і світ навколо нас. Наука - це одна із систем віри, що змінює інші, якими ми намагаємось пояснити те, що спостерігаємо, цим самим, змінюючи себе, щоб пристосуватися до нової інформації, одержуваної з зовнішнього середовища. Багато чого з того, що ми бачимо і спостерігаємо, можна пояснити єдиною теорією: теорією еволюції через спадковість, змінність і відбір.

Теорія еволюції вплинула на зміну світогляду людей із самої своєї появи. Теорія, що Чарльз Дарвін представив у роботі, відомої як "Походження Видів", у 1859 році, стала початком цієї зміни. Багато областей наукового знання в даний час насолоджуються волею думки в атмосфері, що багатьом зобов'язана революції, викликаною теорією еволюції і розвитку. Але Дарвін, подібно багатьом своїм сучасникам, хто припускав, що в основі розвитку лежить природний відбір, не міг не помилятися. Наприклад, він не зміг показати механізм спадкування, при якому підтримується змінність. Його гіпотеза про пангенезис виявилася неправильною. Це було за п'ятдесят років до того, як теорія спадковості почала поширюватися по світу, і за тридцять років до того, як "еволюційний синтез" зміцнив зв'язок між теорією еволюції і відносно молодою наукою генетикою. Однак Дарвін виявив головний механізм розвитку: відбір у сполученні зі змінністю або, як він його називав, "спуск із модифікацією". У багатьох випадках, специфічні особливості розвитку через змінність і відбір все ще не безперечні, однак, основні механізми пояснюють неймовірно широкий спектр явищ, що спостерігаються в Природі.

Тому не дивно, що вчені, які займаються комп'ютерними дослідженнями, звернулися до теорії еволюції в пошуках натхнення. Можливість того, що обчислювальна система, наділена простими механізмами змінності і відбору, могла б функціонувати за аналогією з законами еволюції в природних системах, була дуже приваблива. Ця надія стала причиною появи ряду обчислювальних систем, побудованих на принципах природного відбору.   
Історія еволюційних обчислень почалася з розробки ряду різних незалежних моделей. Основними з них були генетичні алгоритми і класифікаційні системи Голанда (Holland), опубліковані на початку 60-х років і які отримали загальне визнання після виходу у світ книги, що стала класикою в цій області, - "Адаптація в природних і штучних системах" ("Adaptation in Natural and Artifical Systems", 1975).

Головні труднощі з можливістю побудови обчислювальних систем, заснованих на принципах природного відбору і застосуванням цих систем у прикладних задачах, полягає в тому, що природні системи досить хаотичні, а всі наші дії, фактично, носять чітку спрямованість. Ми використовуємо комп'ютер як інструмент для рішення визначених задач, що ми самі і формулюємо та акцентуємо увагу на максимально швидкому виконанні при мінімальних витратах. Природні системи не мають ніяких таких цілей чи обмежень, у всякому разі, нам вони не відомі. Виживання в природі не спрямовано до деякої фіксованої мети, замість цього еволюція робить крок вперед у будь-якому доступному напрямку. Можливо це велике узагальнення, але я думаю, що зусилля, спрямовані на моделювання еволюції за аналогією з природними системами, до дійсного часу можна розбити на дві великі категорії:

1. системи, що змодельовані на біологічних принципах. Вони успішно використовуються для задач функціональної оптимізації і можуть легко бути описані небіологічною мовою;
2. системи, що є біологічно більш реалістичними, але які не виявилися особливо корисними в прикладному змісті. Вони більше схожі на біологічні системи і менш спрямовані (чи не спрямовані зовсім). Вони мають складну і цікаву поведінку, і, скоріше, незабаром набудуть практичного застосування.

Звичайно, на практиці ми не можемо розділяти ці речі так строго. Ці категорії - просто два полюси, між якими лежать різні обчислювальні системи. Ближче до першого полюса - еволюційні алгоритми, такі як Еволюційне Програмування (Evolutionary Programming), Генетичні Алгоритми (Genetic Algorithms) і Еволюційні Стратегії (Evolution Strategies). Ближче до другого полюса - системи, що можуть бути класифіковані як Штучне Життя (Artificial Life).

Звичайно, еволюція біологічних систем не єдине "джерело натхнення" творців нових методів, що моделюють природні процеси.

**Еволюційна теорія**

Як відомо, еволюційна теорія затверджує, що життя на нашій планеті виникло спочатку лише в найпростіших формах - у вигляді одноклітинних організмів. Ці форми поступово ускладнювалися, пристосовуючись до навколишнього середовища, породжуючи нові види, і тільки через багато мільйонів років з'явилися перші тварини і люди. Можна сказати, що кожен біологічний вид з часом вдосконалює свої якості так, щоб найбільш ефективно справлятися з найважливішими задачами виживання, самозахисту, розмноження і т.д. Таким шляхом виникло захисне забарвлення в багатьох риб і комах, панцир у черепахи, отрута в скорпіона і багато інших корисних застосувань.

За допомогою еволюції природа постійно оптимізує все живе, знаходячи часом неординарніші рішення. Неясно, за рахунок чого відбувається цей прогрес, однак йому можна дати наукове пояснення, ґрунтуючись всього на двох біологічних механізмах - природного відбору і генетичного спадкування.

**Природний відбір і генетичне спадкування**

Ключову роль в еволюційній теорії грає природний відбір. Його суть полягає в тому, що найбільш пристосовані особини краще виживають і приносять більше нащадків, ніж менш пристосовані. Помітимо, що сам по собі природний відбір ще не забезпечує розвиток біологічного виду. Тому дуже важливо вивчити, яким чином відбувається спадкування, тобто як властивості нащадку залежать від властивостей батьків.

Основний закон спадкування інтуїтивно зрозумілий кожному - він полягає в тому, що нащадки схожі на батьків. Зокрема, нащадки більш пристосованих батьків будуть, швидше за все, одними з найбільш пристосованих у своєму поколінні. Щоб зрозуміти, на чому заснована ця подібність, нам буде потрібно небагато поглибитися в побудову природної клітини - у світ генів і хромосом.

Майже в кожній клітині будь-якої особини є набір хромосом, що несуть інформацію про цю особину. Основна частина хромосоми - нитка ДНК, що визначає, які хімічні реакції будуть відбуватися в даній клітині, як вона буде розвиватися і які функції виконувати.

Ген - це відрізок ланцюга ДНК, відповідальний за визначену властивість особини, наприклад за колір очей, тип волосся, колір шкіри і т.д. Вся сукупність генетичних ознак людини кодується за допомогою приблизно 60 тис. генів, довжина яких складає більш 90 млн. нуклеотидів.

Розрізняють два види клітин: статеві (такі, як сперматозоїд і яйцеклітина) і соматичні. В кожній соматичній клітині людини міститься 46 хромосом. Ці 46 хромосом - насправді 23 пари, причому в кожній парі одна з хромосом отримана від батька, а друга - від матері. Парні хромосоми відповідають за однакові ознаки - наприклад, батьківська хромосома може містити ген чорного кольору око, а парна їй материнська - ген голубого кольору. Існують визначені закони, що керують участю тих чи інших генів у розвитку особини. Зокрема, у нашому прикладі нащадок буде чорнооким, оскільки ген блакитних очей є "слабким" (рецесивним) і подавляється геном будь-якого іншого кольору.

В статевих клітинах хромосом тільки 23, і вони непарні. При заплідненні відбувається злиття чоловічої і жіночої статевих клітин і утворюється клітина зародка, що містить саме 46 хромосом. Які властивості нащадок одержить від батька, а які - від матері? Це залежить від того, які саме статеві клітини брали участь у заплідненні. Справа в тім, що процес вироблення статевих клітин (так званий мейоз) в організмі піддається випадкам, завдяки яким нащадки все-таки багато в чому відрізняються від своїх батьків. При мейозі, зокрема, відбуваються наступне: парні хромосоми соматичної клітини зближаються впритул, потім їхні нитки ДНК розриваються в декількох випадкових місцях і хромосоми обмінюються своїми частинами (рис. 1).

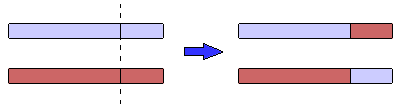


Рис.1. Умовна схема кросінговеру

Цей процес забезпечує появу нових варіантів хромосом і зветься "кросинговер". Кожна з нових хромосом з'явиться потім всередині однієї з статевих клітин, і її генетична інформація може реалізуватись в нащадках даної особини.

Другий важливий фактор, що впливає на спадковість, - це мутації, що виражаються в зміні деяких ділянок ДНК. Мутації також випадкові і можуть бути викликані різними зовнішніми факторами, такими, як радіоактивне випромінювання. Якщо мутація відбулася в статевій клітині, то змінений ген може передатися нащадку і проявитися у вигляді успадкованої хвороби або в інших нових властивостях нащадка. Вважається, що саме мутації є причиною появи нових біологічних видів, а кросинговер визначає вже змінність всередині виду (наприклад, генетичні розходження між людьми).

**Задачі оптимізації**

Як уже було відзначено вище, еволюція - це процес постійної оптимізації біологічних видів. Тепер ми в стані зрозуміти, як відбувається цей процес. Природний відбір гарантує, що найбільш пристосовані особини дадуть досить велику кількість нащадків, а завдяки генетичному спадкуванню ми можемо бути упевнені, що частина нащадків не тільки збереже високу пристосованість батьків, але буде мати і деякі нові властивості. Якщо ці нові властивості виявляються корисними, то з великою імовірністю вони перейдуть і в наступне покоління. Таким чином, відбувається нагромадження корисних якостей і поступове підвищення пристосованості біологічного виду в цілому. Знаючи, як вирішується задача оптимізації видів у природі, ми тепер застосуємо схожий метод для рішення різних реальних задач.

Задачі оптимізації - найбільш розповсюджений і важливий для практики клас задач. Їх приходиться вирішувати кожному з нас або в побуті, розподіляючи свій час між різними справами, або на роботі, домагаючись максимальної швидкості роботи програми чи максимальної прибутковості компанії - у залежності від посади. Серед цих задач є розв'язувані простим шляхом, але є і такі, точне рішення яких знайти практично неможливо.

Уведемо позначення і приведемо кілька класичних прикладів. Як правило, у задачі оптимізації ми можемо керувати декількома параметрами (позначимо їх значення через x1, x2, ..., xn, а нашою метою є максимізація (чи мінімізація) деякої функції, f(x1, x2, ..., xn), що залежить від цих параметрів. Функція f називається цільовою функцією. Наприклад, якщо потрібно максимізувати цільову функцію "дохід компанії", тоді керованими параметрами будуть число співробітників компанії, обсяг виробництва, витрати на рекламу, ціни на кінцеві продукти і т.д. Важливо відзначити, що ці параметри зв'язані між собою - зокрема, при зменшенні числа співробітників швидше за все впаде й обсяг виробництва.

Генетичний алгоритм - новітній, але не єдино можливий спосіб рішення задач оптимізації.

Відомо два основні шляхи рішення таких задач - переборний та градієнтний. Розглянемо класичну задачу комівояжера. Суть задачі полягає у знаходженні короткого шляху проходження всіх міст.

* Переборний метод є найпростішим. Для пошуку оптимального рішення (максимум цільової функції) потрібно послідовно обчислити значення функції у всіх точках. Недоліком є велика кількість обчислень.
* Іншим способом є градієнтний спуск. Обираємо випадкові значення параметрів, а потім значення поступово змінюють, досягаючи найбільшої швидкості зросту цільової функції. Алгоритм може зупинитись, досягнувши локального максимуму. Градієнтні методи швидкі, але не гарантують оптимального рішення (оскільки цільова функція має декілька максимумів).

Генетичний алгоритм уявляє собою комбінацію переборного та градієнтного методів. Механізми кросоверу (схрещування) та мутації реалізують переборну частину, а відбір кращих рішень - градієнтний спуск.

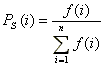
Тобто, якщо на деякій множині задана складна функція від декількох змінних, тоді генетичний алгоритм є програмою, яка за зрозумілий час знаходить точку, де значення функції знаходиться достатньо близько до максимально можливого значення. Обираючи прийнятний час розрахунку, отримуємо одне з кращих рішень, які можна отримати за цей час.

**Робота генетичного алгоритму**

Уявимо собі штучний світ, населений множиною істот (особин), причому кожна особина - це деяке рішення нашої задачі. Будемо вважати особину тим більше пристосованої, чим краще відповідне рішення (чим більше значення цільової функції воно дає). Тоді задача максимізації цільової функції зводиться до пошуку найбільш пристосованої істоти. Звичайно, ми не можемо поселити в наш віртуальний світ всі істоти відразу, тому що їх дуже багато. Замість цього ми будемо розглядати багато поколінь, що змінюють один одного. Тепер, якщо ми зуміємо ввести в дію природний відбір і генетичне спадкування, тоді отримане середовище буде підкорятися законам еволюції. Помітимо, що, відповідно до нашого визначення пристосованості, метою цієї штучної еволюції буде саме створення найкращих рішень. Очевидно, еволюція - нескінченний процес, у ході якого пристосованість особин поступово підвищується. Примусово зупинивши цей процес через досить довгий час після його початку і вибравши найбільш пристосовану особину у поточному поколінні, ми одержимо не абсолютно точну, але близьку до оптимальної відповідь. Така, коротенько, ідея генетичного алгоритму. Перейдемо тепер до точних визначень і опишемо роботу генетичного алгоритму більш детально.

Для того щоб говорити про генетичне спадкування, потрібно наділити наші особини хромосомами. У генетичному алгоритмі хромосома - це деякий числовий вектор, що відповідає параметру, який підбирається, а набір хромосом даної особини визначає рішення задачі. Які саме вектори варто розглядати в конкретній задачі, вирішує сам користувач. Кожна з позицій вектора хромосоми називається ген.

Простий генетичний алгоритм випадковим образом генерує початкову популяцію структур. Робота генетичного алгоритму уявляє собою ітераційний процес, що продовжується доти, поки не виконаються задане число поколінь або будь-який інший критерій зупинки. В кожному поколінні генетичного алгоритму реалізується відбір пропорційно пристосованості, одноточковий кросинговер і мутація. Спочатку, пропорційний відбір призначає кожній структурі імовірність Ps(i) рівну відношенню її пристосованості до сумарної пристосованості популяції:



Потім відбувається відбір (із заміщенням) усіх n особин для подальшої генетичної обробки, відповідно до величини Ps(і).

При такому відборі члени популяції з більш високою пристосованістю з більшою імовірністю будуть частіше вибиратися, ніж особини з низькою пристосованістю. Після відбору, n обраних особин випадковим чином розбиваються на n/2 пари. Для кожної пари з імовірністю Pc може застосовуватися кросинговер. Відповідно з імовірністю 1-Pc кросинговер не відбувається і незмінені особини переходять на стадію мутації. Якщо кросинговер відбувається, отримані нащадки заміняють собою батьків і переходять до мутації.

Визначимо тепер поняття, що відповідають мутації і кросинговеру в генетичному алгоритмі.

**Мутація** - це перетворення хромосоми, що випадково змінює одну чи декілька її позицій (генів). Найбільш розповсюджений вид мутацій - випадкова зміна тільки одного з генів хромосоми.

**Кросинговер** (у літературі по генетичних алгоритмах також вживається назва кросовер або схрещування) - це операція, при якій із двох хромосом породжується одна чи декілька нових хромосом. Одноточковий кросинговер працює в такий спосіб. Спочатку, випадковим образом вибирається одна з l-1 точок розриву. (Точка розриву - ділянка між сусідніми бітами в рядку.) Обидві батьківські структури розриваються на два сегменти по цій точці. Потім, відповідні сегменти різних батьків склеюються і виходять два генотипи нащадків.

Наприклад, припустимо, один батько складається з 10 нулів, а інший - з 10 одиниць. Нехай з 9 можливих точок розриву обрана точка 3. Батьки і їхні нащадки показані нижче.

Кросинговер

Батько 1 0000000000 000~0000000--> 111~0000000 1110000000 Нащадок 1

Батько 2 1111111111 111~1111111 --> 000~1111111 0001111111 Нащадок 2

Після того як закінчується стадія кросинговеру, виконуються оператори мутації. У кожному рядку, що піддається мутації, кожен біт з імовірністю Pm змінюється на протилежний.

Популяція, отримана після мутації записує поверх старої і цим цикл одного покоління завершується. Наступні покоління обробляються подібним чином: відбір, кросинговер і мутація.

В даний час дослідники ген пропонують багато інших операторів відбору, кросинговеру і мутації. От лише найбільш розповсюджені з них.

Елітні методи відбору гарантують, що при відборі обов'язково будуть виживати кращий чи кращі члени популяції сукупності. Найбільш поширена процедура обов'язкового збереження тільки одної кращої особини, якщо вона не пройшла як інші через процес відбору, кросинговеру і мутації. Елітизм може бути впроваджений практично в будь-який стандартний метод відбору.

Двоточковий кросинговер і рівномірний кросинговер - цілком гідні альтернативи одноточковому оператору. В двоточковому кросинговері вибираються дві точки розриву, і батьківські хромосоми обмінюються сегментом, що знаходиться між двома цими точками. У рівномірному кросинговері, кожен біт першого батька успадковується першим нащадком із заданою імовірністю; у противному випадку цей біт передається другому нащадку. І навпаки.

Блок-схема генетичного алгоритму зображена на рис. 2. Спочатку генерується початкова популяція особин (індивідуумів), тобто деякий набір рішень задачі. Як правило, це робиться випадковим образом. Потім ми повинні змоделювати розмноження всередині цієї популяції. Для цього випадково відбирається декілька пар індивідуумів, відбувається схрещування між хромосомами в кожній парі, а отримані нові хромосоми втілюються в популяцію нового покоління. У генетичному алгоритмі зберігається основний принцип природного відбору - чим пристосованіше індивідуум (чим більше відповідне йому значення цільової функції), тим з більшою імовірністю він буде брати участь у схрещуванні. Тепер моделюються мутації - у декількох випадково обраних особинах нового покоління змінюються деякі гени. Потім стара популяція частково або цілком знищується і ми переходимо до розгляду наступного покоління. Популяція наступного покоління в більшості реалізацій генетичних алгоритмів містить стільки ж особин, скільки початкова, але в силу відбору пристосованість у ній у середньому вище. Тепер описані процеси відбору, схрещування й мутації повторюються вже для цієї популяції і т.д.

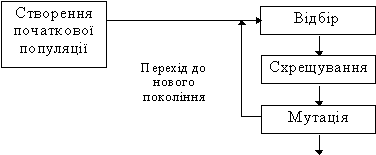


Рис. 2. Блок-схема генетичного алгоритму

В кожному наступному поколінні ми будемо спостерігати виникнення зовсім нових рішень нашої задачі. Серед них будуть як погані, так і хороші, але завдяки відбору число прийнятних рішень буде зростати. Помітимо, що в природі не буває абсолютних гарантій, і найпристосованіший тигр може загинути від рушничного пострілу, не залишивши нащадків. Імітуючи еволюцію на комп'ютері, ми можемо уникати подібних небажаних подій і завжди зберігати життя кращому з індивідуумів поточного покоління - така методика називається "стратегією елітизму".

**Застосовування генетичних алгоритмів**

Генетичні алгоритми в різних формах застосовуються до вирішення багатьох наукових і технічних проблем. Генетичні алгоритми використовуються при створенні інших обчислювальних структур, наприклад, автоматів або мереж сортування. У машинному навчанні вони використовуються при проектуванні нейронних мереж або керуванні роботами. Вони також застосовуються при моделюванні розвитку в різних предметних областях, включаючи біологічні (екологія, імунологія і популяційна генетика) та соціальні (такі як економіка і політичні системи) системи.

Проте, можливо найбільш популярне застосування генетичних алгоритмів - оптимізація багатопараметричних функцій. Багато реальних задач можуть бути сформульовані як пошук оптимального значення, де значення - складна функція, що залежить від певних вхідних параметрів. У деяких випадках, потрібно знайти ті значення параметрів, при яких досягається найкраще точне значення функції. В інших випадках, точний оптимум не потрібний - рішенням може вважатися будь-яке значення, краще за певну задану величину. У цьому випадку, генетичні алгоритми - часто найбільш прийнятний метод для пошуку "прийнятних" значень. Сила генетичного алгоритму полягає в його здатності маніпулювати одночасно багатьма параметрами, що використовується в сотнях прикладних програм, включаючи проектування літаків, налаштування параметрів алгоритмів і пошуку стійких станів систем нелінійних диференціальних рівнянь.

Наприкінці можна підсумувати, що генетичні алгоритми є ефективною процедурою пошуку, що конкурує з іншими процедурами. Ефективність генетичних алгоритмів сильно залежить від таких деталей, як метод кодування рішень, операторів, налаштування параметрів, окремих критеріїв успіху. Теоретична робота, відбита в літературі, присвяченої генетичним алгоритмам, не дає підстав говорити про вироблення певних строгих механізмів для чітких передбачень.

**Тема 11. Нечітка логіка**

* [Нечіткі множини](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_1)
* [Основні характеристики нечітких множин](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_2)
  + [Приклади нечітких множин](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_2_1)
  + [Методи побудови функцій приналежності нечітких множин](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_2_2)
* [Операції над нечіткими множинами](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_2_3)
  + [Приклади](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_3_4)
  + [Наочне представлення операцій над нечіткими множинами](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_3_5)
  + [Властивості операцій И і З](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_3_6)
* [Нечітка і лінгвістична змінні](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_4)
  + [Приклад](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_4_1)
* [Нечіткі висловлення і нечіткі моделі систем](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_5)
  + [Висловлення на множині значень фіксованої лінгвістичної змінної](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_5_1)
* [Нечіткі множини в системах керування](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_6)
  + [Загальна структура нечіткого мікроконтролера](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_6_1)
* [Нечітка логіка в Matlab](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_7)
* [Переваги нечітких систем](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_8)
* [Застосування нечітких систем](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_9)

Мабуть, найбільш вражаючою властивістю людського інтелекту є здатність приймати правильні рішення в умовах неповної і нечіткої інформації. Побудова моделей наближених роздумів людини і використання їх у комп'ютерних системах представляє сьогодні одну з найважливіших проблем науки.

Основи нечіткої логіки були закладені наприкінці 60-х років у працях відомого американського математика Латфі Заде. Соціальне замовлення на дослідження подібного роду було викликано зростаючим незадоволенням експертними системами. Хвалений "штучний інтелект", що легко справлявся із задачами керування складними технічними комплексами, був безпорадним при найпростіших висловленнях повсякденного життя, типу "Якщо машиною перед тобою керує недосвідчений водій - тримайся від неї подалі". Для створення дійсно інтелектуальних систем, здатних адекватно взаємодіяти з людиною, необхідний був новий математичний апарат, що переводить невиразні і неоднозначні життєві твердження в мову чітких і формальних математичних формул.

Першим серйозним кроком у цьому напрямку з'явилася теорія нечітких множин, розроблена Заде. Його робота "Fuzzy Sets", що з'явилася в 1965 році в журналі "Information and Control", заклала основи моделювання інтелектуальної діяльності людини і з'явилася початковим поштовхом до розвитку нової математичної теорії. Він же дав і назву для нової області науки -"fuzzy logic"(fuzzy - нечіткий, розмитий, м'який).

Щоб стати класиком, треба небагато випередити свій час. Існує легенда про те, яким чином була створена теорія "нечітких множин". Один раз Заде мав довгу дискусію зі своїм другом відносно того, чия з дружин є більш привабливою. Термін "приваблива" є дуже невизначеним і в результаті дискусії вони не змогли прийти до задовільного підсумку. Це змусило Заде сформулювати концепцію, що виражає нечіткі поняття типу "приваблива" у числовій формі.

Подальші роботи професора Л.Заде і його послідовників заклали міцний фундамент нової теорії і створили передумови для впровадження методів нечіткого управління в інженерну практику.

Апарат теорії нечітких множин, продемонструвавши ряд багатообіцяючих можливостей застосування - від систем керування літальними апаратами до прогнозування підсумків виборів, виявився разом з тим надмірно складним для втілення, враховуючи наявний на той час рівень технології - і на багато років нечітка логіка зайняла своє місце в ряді інших спеціальних наукових дисциплін - десь посередині між експертними системами і нейронними мережами...

Своє друге народження теорія нечіткої логіки пережила на початку вісімдесятих років, коли відразу кілька груп дослідників (в-основному в США і Японії) всерйоз зайнялися створенням електронних систем різного застосування, що використовують нечіткі керуючі алгоритми. Теоретичні основи для цих спроб були закладені в ранніх працях Коско й інших учених.

Третій період почався з кінця 80-х років і дотепер. Цей період характеризується бумом практичного застосування теорії нечіткої логіки в різних сферах науки і техніки. До 90-го року з'явилося близько 40 патентів, що відносяться до нечіткої логіки (30 - японських). Сорок вісім японських компаній утворили спільну лабораторію LIFE (Laboratory for International Fuzzy Engineering), японський уряд фінансував 5-річну програму по нечіткій логіці, що включає 19 різних проектів - від систем оцінки глобального забруднення атмосфери і передбачення землетрусів до АСУ заводських цехів і складів. Результатом виконання цієї програми з'явилася поява цілого ряду нових масових мікрочіпів, заснованих на нечіткій логіці. Сьогодні їх можна знайти в пральних машинах і відеокамерах, цехах заводів і моторних відсіків автомобілів, у системах керування складськими роботами і бойовими вертольотами.

У США розвиток нечіткої логіки йде по шляху створення систем, що потрібні великому бізнесу і військовим. Нечітка логіка застосовується при аналізі нових ринків, біржовій грі, оцінці політичних рейтингів, виборі оптимальної цінової стратегії і т.п. З'явилися і комерційні системи масового застосування.

Зсув центра досліджень нечітких систем вбік практичних застосувань привело до постановки цілого ряду проблем, зокрема:

* нові архітектури комп'ютерів для нечітких обчислень;
* елементна база нечітких комп'ютерів і контролерів;
* інструментальні засоби розробки;
* інженерні методи розрахунку і розробки нечітких систем керування, тощо.

**Нечіткі множини**

Нехай E - універсальна множина, x - елемент E, а R - певна властивість. Звичайна (чітка) підмножина A універсальної множини E, елементи якої задовольняють властивості R, визначається як множина впорядкованої пари A = {mA (х)/х}, де mA(х) - характеристична функція, що приймає значення 1, якщо x задовольняє властивості R, і 0 - в іншому випадку.

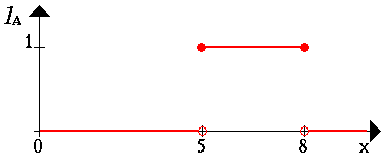
Нечітка підмножина відрізняється від звичайної тім, що для елементів x з E немає однозначної відповіді "ні" відносно властивості R. У зв'язку з цим, нечітка підмножина A універсальної множини E визначається як множина впорядкованої пари A = {mA(х)/х}, де mA(х) - характеристична функція приналежності (або просто функція приналежності), що приймає значення в деякій впорядкованій множині M (наприклад, M = [0,1]).

Функція приналежності вказує ступінь (або рівень) приналежності елемента x до підмножини A. Множина M називають множиною приналежностей. Якщо M = {0,1}, тоді нечітка підмножина A може розглядатися як звичайна або чітка множина.

Розглянемо множину X всіх чисел від 0 до 10. Визначимо підмножину A множини X всіх дійсних чисел від 5 до 8.

A = [5,8]

Покажемо функцію приналежності множини A, ця функція ставить у відповідність число 1 чи 0 кожному елементу в X, у залежності від того, належить даний елемент підмножині A чи ні. Результат представлений на наступному малюнку:



Можна інтерпретувати елементи, яким поставлена у відповідність 1, як елементи, що знаходяться в множині A, а елементи, яким поставлений у відповідність 0, як елементи, що не знаходяться в множині A.

Ця концепція використовується в багатьох областях застосувань. Але можна легко знайти ситуації, в яких даній концепції буде бракувати гнучкості.

У даному прикладі опишемо множину молодих людей. Більш формально можна записати так

B = {множина молодих людей}

Оскільки, взагалі, вік починається з 0, то нижня межа цієї множини повинна бути нулем. Верхню межу визначити небагато складніше. Спочатку встановимо верхню межу, скажемо, рівну 20 рокам. Таким чином, маємо B як чітко обмежений інтервал, буквально: B=[0,20] . Виникає питання: чому хтось у свій двадцятирічний ювілей - молодий, а відразу наступного дня вже не молодий? Очевидно, це структурна проблема, і якщо пересунути верхню межу в довільну точку, то можна задати таке ж питання.

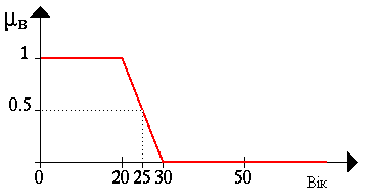
Більш природний шлях отримання множини B складається в ослабленні строгого поділу на молодих і не молодих. Зробимо це, виносячи не тільки чіткі судження Так, він належить множини молодих людей або Ні, вона не належить множини молодих людей, але і більш гнучкі формулювання ТАК, він належить до досить молодих людей або Ні, він не дуже молодий.

Розглянемо як за допомогою нечіткої множини визначити такий вираз, як він ще молодий.

В першому прикладі ми кодували всі елементи множини за допомогою 0 чи 1. Простим способом узагальнити дану концепцію є введення значення між 0 і 1. Реально можна навіть допустити нескінченне число значень між 0 і 1, в одиничному інтервалі I = [0, 1].

Інтерпретація чисел при співвідношенні всіх елементів множини стає тепер більш складною. Звичайно, знову число 1 ставиться у відповідність тому елементу, що належить множині B, а 0 означає, що елемент точно не належить множині B. Всі інші значення визначають ступінь приналежності до множини B.

Для наочності приведемо характеристичну функцію множини молодих людей, як і в першому прикладі.



Нехай E = {x1, x2, x3, x4, x5 }, M = [0,1]; A - нечітка множина, для якої

mA(x1)=0,3; mA(x2)=0; mA(x3)=1; mA(x4)=0,5; mA(x5)=0,9

Тоді A можна представити у виді:

A = {0,3/x1; 0/x2; 1/x3; 0,5/x4; 0,9/x5 } або

A = 0,3/x1 + 0/x2 + 1/x3 + 0,5/x4 + 0,9/x5,

(знак "+" є операцією не додавання, а об'єднання) або

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | x1 | x2 | x3 | x4 | x5 |
| A = | 0,3 | 0 | 1 | 0,5 | 0,9 |

**Основні характеристики нечітких множин**

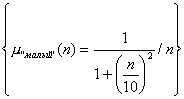
Нехай M = [0,1] і A - нечітка множина з елементами з універсальної множини E і множиною приналежностей M.

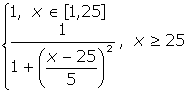
* Величина http://www.victoria.lviv.ua/html/oio/images/theme11/ris3.gifmA(x) називається **висотою** нечіткої множини A. Нечітка множина A є нормальною, якщо її висота дорівнює 1, тобто верхня границя її функції приналежності дорівнює 1 ( http://www.victoria.lviv.ua/html/oio/images/theme11/ris3.gifmA(x)=1). При mA(x)<1 нечітка множина називається **субнормальною**.
* Нечітка множина є **порожньою**, якщо "xОE m A(x)=0. Непорожню субнормальну множину можна нормалізувати по формулі mA(x) := http://www.victoria.lviv.ua/html/oio/images/theme11/ris4.gif
* Нечітка множина є **унімодальною**, якщо mA(x)=1 лише для одного x з E.
* **Носієм** нечіткої множини A є звичайна підмножина з властивістю mA(x)>0, тобто носій A = {x/mA(x)>0} " xОE.
* Елементи xОE, для яких mA(x)=0,5 називаються **точками переходу** множини A.

**Приклади нечітких множин**

1. Нехай E = {0,1,2,..,10}, M =[0,1]. Нечітку множину "кілька" можна визначити таким чином: "кілька" = 0,5/3+0,8/4+1/5+1/6+0,8/7+0,5/8;

її характеристики: висота = 1, носій={3,4,5,6,7,8}, точки переходу - {3,8}.

2. Нехай E = {0,1,2,3,...,n,...}. Нечітку множину "малий" можна визначити: "малий" = 

3. Нехай E = {1,2,3,...,100} і відповідає поняттю "вік", тоді нечітку множину "молодий", можна визначити за допомогою m"молодий"(x) = 

Нечітка множина "молодий" на універсальній множині E' ={Іванов, Петров, Сидоров,...} задається за допомогою функції приналежності m"молодий"(x) на E = {1,2,3,..100} (вік), що називається відносно E' функцією сумісності, при цьому:

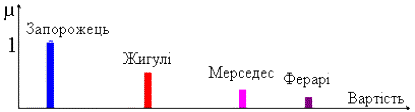
m"молодий"(Сидоров) = m"молодий"(x), де x - вік Сидорова.

4. Нехай E = {Запорожець, Жигулі, Мерседес,....} - множина марок автомобілів, а E' = [0,µ] - універсальна множина "вартість", тоді на E' ми можемо визначити нечіткі множини типу: "для небагатих ", "для середнього класу", "престижні", з функціями приналежності типу:



Маючи ці функції і знаючи ціни автомобілів з E у даний момент часу, мі тим самим визначимо на E' нечіткі множини з цими ж назвами.

Так, наприклад, нечітка множина "для небагатих", що задана на універсальній множині E={Запорожець, Жигулі, Мерседес,....} виглядає таким чином:



Аналогічно можна визначити нечітку множину "швидкісні", "середні", "тихохідні" і т.д.

**Методи побудови функцій приналежності нечітких множин**

У приведених вище прикладах використані прямі методи, коли експерт або просто задає для кожного xОE значення mA(x), або визначає функцію приналежності. Як правило, прямі методи завдання функції приналежності використовуються для вимірних понять, таких як швидкість, година, відстань, тиск, температура і т.д., тобто коли виділяються полярні значення.

У багатьох задачах при характеристиці об'єкта можна виділити набір ознак і для кожного з них визначити полярні значення, що відповідають значенням функції приналежності, 0 чи 1.

Наприклад, у задачі розпізнавання обличчя можна виділити наступні пункти:

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | 0 | 1 |
| x1 | висота чола | низький | широкий |
| x2 | профіль носа | кирпатий | горбатий |
| x3 | довжина носа | короткий | довгий |
| x4 | розріз очей | вузькі | широкі |
| x5 | колір очей | світлі | темні |
| x6 | форма підборіддя | гострий | квадратний |
| x7 | товщина губ | тонкі | товсті |
| x8 | колір обличчя | темний | світлий |
| x9 | обрис обличчя | овальне | квадратне |

Для конкретного обличчя А експерт, виходячи з приведеної шкали, задає mA(x)О [0,1], формуючи векторну функцію приналежності { mA(x1), mA(x2),... mA(x9)}.

Непрямі методи визначення значень функції приналежності використовуються у випадках, коли немає елементарних вимірних властивостей, через які визначається потрібна нечітка множина. Як правило, це методи попарних порівнянь. Якби значення функцій приналежності були нам відомі, наприклад, mA(xi) = wi, i=1,2,...,n, тоді попарні порівняння можна представити матрицею відношень A = {aij}, де aij=wi/wj (операція ділення).

**Операції над нечіткими множинами**

**Вміщення**

Нехай A і B - нечіткі множини на універсальній множині E.

Говорять, що A міститься в B, якщо "x ОE mA(x) <mB(x).

Позначення: A М B.

Іноді використовують термін "домінування", тобто у випадку коли A М B, говорять, що B домінує A.

**Рівність**

A і B рівні, якщо "xОE mA(x) = mB (x).

Позначення: A = B.

**Доповнення**

Нехай M = [0,1], A і B - нечіткі множини, задані на E. A і B доповнюють один одного, якщо

"xОE mA(x) = 1 - m B(x).

Позначення: B = http://www.victoria.lviv.ua/html/oio/images/theme11/image010.gifчи A =http://www.victoria.lviv.ua/html/oio/images/theme11/image012.gif

Очевидно, що http://www.victoria.lviv.ua/html/oio/images/theme11/image012.gif= A. (Доповнення визначене для M = [0,1], але очевидно, що його можна визначити для будь-якого впорядкованого M).

**Перетинання**

AЗB - найбільша нечітка підмножина, що міститься одночасно в A і B.

mAЗB(x) = min( mA(x), mB(x)).

**Об'єднання**

А И В - найменша нечітка підмножина, що включає як А, так і В, з функцією приналежності:

mAИ B(x) = max(mA(x), m B(x)).

**Різниця**

А - B = АЗ http://www.victoria.lviv.ua/html/oio/images/theme11/image010.gifз функцією приналежності:

mA-B(x) = mA З http://www.victoria.lviv.ua/html/oio/images/theme11/image010.gif(x) = min( mA(x), 1 - m B(x)).

**Диз'юнктивна сума**

АЕB = (А - B)И(B - А) = (А Зhttp://www.victoria.lviv.ua/html/oio/images/theme11/image010.gif ) И( http://www.victoria.lviv.ua/html/oio/images/theme11/image012.gifЗ B) з функцією приналежності:

mA-B(x) = max{[min{m A(x), 1 - mB(x)}];[min{1 - mA(x), mB(x)}] }

**Приклади**

Нехай:

A = 0,4/ x1 + 0,2/ x2+0/ x3+1/ x4;

B = 0,7/ x1+0,9/ x2+0,1/ x3+1/ x4;

C = 0,1/ x1+1/ x2+0,2/ x3+0,9/ x4.

Тут:

1. AМB, тобто A міститься в B чи B домінує A, С незрівнянно ні з A, ні з B, тобто парі {A, С} і {A, С} - парі недомінуємих нечітких множин.

2. A № B №C.

3. http://www.victoria.lviv.ua/html/oio/images/theme11/image012.gif= 0,6/ x1 + 0,8/x2 + 1/x3 + 0/x4;

http://www.victoria.lviv.ua/html/oio/images/theme11/image010.gif= 0,3/x1 + 0,1/x2 + 0,9/x3 + 0/x4.

4. AЗB = 0,4/x1 + 0,2/x2 + 0/x3 + 1/x4.

5. АИС = 0,7/x1 + 0,9/x2 + 0,1/x3 + 1/x4.

6. А - С = АЗ http://www.victoria.lviv.ua/html/oio/images/theme11/image010.gif= 0,3/x1 + 0,1/x2 + 0/x3 + 0/x4;

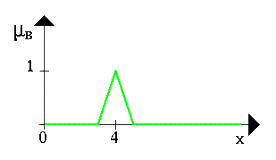
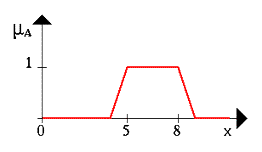
В - А =http://www.victoria.lviv.ua/html/oio/images/theme11/image012.gif З С = 0,6/x1 + 0,8/x2 + 0,1/x3 + 0/x4.

7. А Е В = 0,6/x1 + 0,8/x2 + 0,1/x3 + 0/x4.

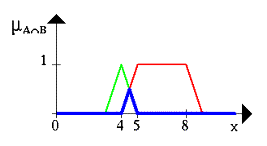
**Наочне представлення операцій над нечіткими множинами**

Для нечітких множин можна будувати візуальне представлення. Розглянемо прямокутну систему координат, на осі ординат якої відкладаються значення mA(x), на осі абсцис у довільному порядку розташовані елементи E (мі вже використовували таке представлення в прикладах нечітких множин). Якщо E по своїй природі впорядковано, те цей порядок бажано зберегти в розташуванні елементів на осі абсцис. Таке представлення робить наочними прості операції над нечіткими множинами.

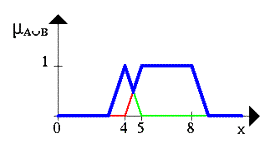
Нехай A нечіткий інтервал між 5 до 8 і B нечітке число близько 4, як показано на рисунку.



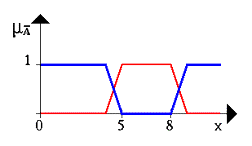
Проілюструємо нечітку множину між 5 і 8 I (AND) близько 4 (синя лінія).



Нечітка множина між 5 і 8 АБО (OR) близько 4 показана на наступному рисунку (знову синя лінія).



Наступний малюнок ілюструє операцію заперечення. Синя лінія - це ЗАПЕРЕЧЕННЯ нечіткої множини A.



В наступному рисунку заштрихована частина відповідає нечіткій множині A і зображує область значень А і всіх нечітких множин, що містяться в A. Решта рисунків зображують відповідно http://www.victoria.lviv.ua/html/oio/images/theme11/image012.gif, AЗhttp://www.victoria.lviv.ua/html/oio/images/theme11/image012.gif, AИhttp://www.victoria.lviv.ua/html/oio/images/theme11/image012.gif.

|  |  |
| --- | --- |
| http://www.victoria.lviv.ua/html/oio/images/theme11/image041.gif | http://www.victoria.lviv.ua/html/oio/images/theme11/image042.gif |
| http://www.victoria.lviv.ua/html/oio/images/theme11/image043.gif | http://www.victoria.lviv.ua/html/oio/images/theme11/image044.gif |

**Властивості операцій И і З**

Нехай А, В, С - нечіткі множини, тоді виконуються наступні властивості:

* http://www.victoria.lviv.ua/html/oio/images/theme11/image046.gif- комутативність;
* http://www.victoria.lviv.ua/html/oio/images/theme11/image048.gif- асоціативність;
* http://www.victoria.lviv.ua/html/oio/images/theme11/image050.gif- ідемпотентність;
* http://www.victoria.lviv.ua/html/oio/images/theme11/image052.gif- дистрибутивність;
* AИЖ = A, де Ж - порожня множина, тобто mЖ(x) = 0 "xОE;
* AЗЖ = Ж;
* AЗE = A, де E - універсальна множина;
* AИE = E;
* http://www.victoria.lviv.ua/html/oio/images/theme11/image054.gif- теореми де Моргана.

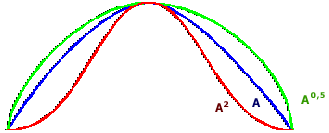
На відміну від чітких множин, для нечітких множин у загальному випадку:

* AЗhttp://www.victoria.lviv.ua/html/oio/images/theme11/image012.gif№ Ж,
* AИhttp://www.victoria.lviv.ua/html/oio/images/theme11/image012.gif№ E.

(Що, зокрема, проілюстровано вище в прикладі наочного представлення нечітких множин).

* CON(A) = A2 - операція концентрування,
* DIL(A) = A0,5 - операція розмиття,

які використовуються при роботі з лінгвістичними змінними.



**Множення на число**

Якщо a - позитивне число, таке, що a http://www.victoria.lviv.ua/html/oio/images/theme11/image061.gifm A(x)Ј1, тоді нечітка множина aA має функцію приналежності:

maA(x) = amA(x).

[Далі...](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm)

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | |  | **Тема 11. Нечітка логіка (продовження)**   * [Нечіткі множини](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_1) * [Основні характеристики нечітких множин](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_2)   + [Приклади нечітких множин](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_2_1)   + [Методи побудови функцій приналежності нечітких множин](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_2_2) * [Операції над нечіткими множинами](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_2_3)   + [Приклади](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_3_4)   + [Наочне представлення операцій над нечіткими множинами](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_3_5)   + [Властивості операцій И і З](http://www.victoria.lviv.ua/html/oio/html/theme11.htm#11_3_6) * [Нечітка і лінгвістична змінні](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_4)   + [Приклад](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_4_1) * [Нечіткі висловлення і нечіткі моделі систем](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_5)   + [Висловлення на множині значень фіксованої лінгвістичної змінної](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_5_1) * [Нечіткі множини в системах керування](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_6)   + [Загальна структура нечіткого мікроконтролера](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_6_1) * [Нечітка логіка в Matlab](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_7) * [Переваги нечітких систем](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_8) * [Застосування нечітких систем](http://www.victoria.lviv.ua/html/oio/html/theme11_1.htm#11_9)   **Нечітка і лінгвістична змінні**  При описі об'єктів і явищ за допомогою нечітких множин використовується поняття нечіткої і лінгвістичної змінних .  Нечітка змінна характеризується трійкою <a, X, A>, де   * a - найменування змінної, * X - універсальна множина (область визначення a), * A - нечітка множина на X, що описує обмеження (тобто m A(x)) на значення нечіткої змінної a.   Лінгвістичною змінною називається набір <b ,T,X,G,M>, де   * b - найменування лінгвістичної змінної; * Т - множина її значень (терм-множина), що представляють собою імена нечітких змінних, областю визначення, кожної з яких є множина X. Множина T називається базовою терм-множиною лінгвістичної змінної; * G - синтаксична процедура, що дозволяє оперувати елементами терм-множини T, зокрема, генерувати нові терми (значення). Множина TИG(T), де G(T) - множина згенерованих термів, називається розширеною терм-множиною лінгвістичної змінної; * М - семантична процедура, що дозволяє перетворити кожне нове значення лінгвістичної змінної, утвореною процедурою G, у нечітку змінну, тобто сформувати відповідну нечітку множину.   Щоб уникнути великої кількості символів:   * символ b використовують як для назви самої змінної, так і для всіх її значень; * для позначення нечіткої множини і його назви користуються одним символом , наприклад, терм "молодий", що є значенням лінгвістичної змінної b = "вік", одночасно є і нечіткою множиною М ("молодий").   Присвоєння декількох значень символам припускає, що контекст дозволяє робити можливі невизначеності.  **Приклад**  Нехай експерт визначає товщину виробу, за допомогою поняття "мала товщина", "середня товщина" і "велика товщина", при цьому мінімальна товщина дорівнює 10 мм, а максимальна - 80 мм.  Формалізація такого опису може бути проведена за допомогою наступної лінгвістичної змінної <b, T, X, G, M>, де   * b - товщина виробу; * T - {"мала товщина", "середня товщина", "велика товщина"}; * X - [10, 80]; * G - процедура утворення нових термів за допомогою зв'язувань "і", "або" і модифікаторів типу "дуже", "не", "злегка" і ін. Наприклад, "мала або середня товщина", "дуже мала товщина" і ін.; * М - процедура завдання на X = [10, 80] нечітких підмножин А1="мала товщина", А2 = "середня товщина", А3="велика товщина", а також нечітких множин для термів з G(T) відповідно до правил трансляції нечітких зв'язувань і модифікаторів "і", "або", "не", "дуже", "злегка" і ін. операції над нечіткими множинами виду: А З C, АИ C, http://www.victoria.lviv.ua/html/oio/images/theme11/image012.gif, CON А = А2 , DIL А = А0,5 і ін.   Разом із розглянутими вище базовими значеннями лінгвістичної змінної "товщина" (Т={"мала товщина", "середня товщина", "велика товщина"}) існують можливі значення, що залежать від області визначення Х. У даному випадку значення лінгвістичної змінної "товщина виробу" можуть бути визначені як "близько 20 мм", "близько 50 мм", "близько 70 мм", тобто у вигляді нечітких чисел.  **Функції приналежності нечітких множин:**  "мала товщина" = А1 , "середня товщина"= А2, " велика товщина"= А3 .  http://www.victoria.lviv.ua/html/oio/images/theme11/image065.gif  Функція приналежності:  нечітка множина "мала чи середня товщина" = А1ИА1.  http://www.victoria.lviv.ua/html/oio/images/theme11/image067.gif  **Нечіткі висловлення і нечіткі моделі систем**  Нечіткими висловленнями будемо називати висловлення наступного виду:   1. Висловлення <b є b'>, де b - найменування лінгвістичної змінної, b' - її значення, якому відповідає нечітка множина на універсальній множині Х. Наприклад, висловлення <тиск великий> припускає, що лінгвістичній змінній "тиск" надається значення "великий", для якого на універсальній множині Х змінної "тиск" визначена, відповідно даному значенню "великий", нечітка множина. 2. Висловлення <b є mb'>, де m - модифікатор, якому відповідають слова "ДУЖЕ", "БІЛЬШ-МЕНШ", "НАБАГАТО БІЛЬШЕ" і ін. Наприклад: <тиск дуже великий>, <швидкість набагато більше середньої> і ін. 3. Складні висловлення, утворені з висловлень видів 1. і 2. і союзів "І", "АБО", "ЯКЩО.., ТОДІ...", "ЯКЩО.., ТОДІ.., ІНАКШЕ".   **Висловлення на множині значень фіксованої лінгвістичної змінної**  Якщо значення фіксованої лінгвістичної змінної відповідають нечітким множинам однієї універсальної множини Х, можна ототожнювати модифікатори "дуже" чи "не" з операціями "CON" і "доповнення", а союзи "І", "АБО" з операціями "перетинання" і "об'єднання" над нечіткими множинами .  Для ілюстрації поняття лінгвістичної змінної мі як приклад розглядали лінгвістичну змінну "товщина виробу" з базовою терм-множиною Т = {"мала", "середня", "велика"}. При цьому на Х = [10, 80] ми визначили нечіткі множини А1, А2, А3, що відповідають базовим значенням: "мала", "середня", "велика".  У цьому випадку висловленню <товщина виробу дуже мала> відповідає нечітка множина CONA = A2; висловленню <товщина виробу не велика або середня> - нечітка множина А2И http://www.victoria.lviv.ua/html/oio/images/theme11/image069.gif; висловленню <товщина виробу не мала і не велика> А1Зhttp://www.victoria.lviv.ua/html/oio/images/theme11/image069.gif .  **Нечіткі множини в системах керування**  Найбільш важливим застосуванням теорії нечітких множин є контролери нечіткої логіки. Їх функціонування дещо відрізняється від роботи звичайних контролерів; для опису системи замість диференційних рівнянь використовуються знання експертів. Ці знання можуть бути виражені за допомогою лінгвістичних змінних, які описані нечіткими множинами.  **Загальна структура нечіткого мікроконтролера**  Загальна структура мікроконтролера, що використовує нечітку логіку, показана на рис.1. Вона містить у своєму складі наступні складові:   * блок фазіфікації; * базу знань; * блок рішень; блок дефазіфікації.   Блок фазіфікації перетворює чіткі величини, виміряні на виході об'єкта керування, у нечіткі величини, що описані лінгвістичними змінними в базі знань.  Блок рішень використовує нечіткі умовні ( if - then ) правила, закладені в базі знань, для перетворення нечітких вхідних даних у необхідні керуючі впливи, що носять також нечіткий характер.  Блок дефазіфікації перетворює нечіткі дані з виходу блоку рішень у чітку величину, що використовується для керування об'єктом.  http://www.victoria.lviv.ua/html/oio/images/theme11/image072.gif  Рис. 1. Загальна структура нечіткого мікроконтролера  Як приклад відомих мікроконтролерів, що підтримують нечітку логіку можна назвати 68HC11, 68HC12 фірми Motorola, MCS-96 фірми Intel, а також деякі інші.  Всі системи з нечіткою логікою функціонують за одним принципом: показання вимірювальних приладів: фазіфікуються (перетворюються в нечіткий формат), обробляються, дефазіфікуються й у вигляді звичайних сигналів подаються на виконавчі пристрої.  Розглянемо випадок керування мобільним роботом, задачею якого є об'їзд перешкод. Введемо дві лінгвістичні змінні: ДИСТАНЦІЯ (відстань від робота до перешкоди) і НАПРЯМОК (кут між подовжньою віссю робота та напрямком на перешкоду).  Розглянемо лінгвістичну змінну ДИСТАНЦІЯ. Значеннями її можна визначити терми ДАЛЕКО, СЕРЕДНЕ, БЛИЗЬКО і ДУЖЕ БЛИЗЬКО. Для фізичної реалізації лінгвістичної змінної необхідно визначити точні фізичні значення термів цієї змінної. Нехай змінна ДИСТАНЦІЯ може приймати будь-які значення з діапазону від нуля до нескінченності. Відповідно до теорії нечітких множин, у такому випадку кожному значенню відстані з зазначеного діапазону може бути поставлене у відповідність деяке число від нуля до одиниці, що визначає ступінь приналежності даної фізичної відстані (припустимо 40 см) до того чи іншого терму лінгвістичної змінної ДИСТАНЦІЯ. Ступінь приналежності визначаємо функцією приналежності М(d), де d-відстань до перешкоди. У нашому випадку відстані 40 см. Можна задати ступінь приналежності до терму ДУЖЕ БЛИЗЬКО рівним 0,7 , а до терму БЛИЗЬКО - 0,3 (рис. 2.). Конкретне визначення ступеня приналежності може проходити тільки при роботі з експертами.  http://www.victoria.lviv.ua/html/oio/images/theme11/image074.gif  Рис. 2. Лінгвістична змінна і функція приналежності  Змінній НАПРЯМОК, яка може приймати значення в діапазоні від 0 до 360 градусів, задамо терми ЛІВИЙ, ПРЯМИЙ і ПРАВИЙ.  Тепер необхідно задати вихідні змінні. У даному прикладі достатньо однієї, яка назвемо РУЛЬОВИЙ КУТ. Вона може містити терми: РІЗКО ВЛІВО, ВЛІВО, ПРЯМО, ВПРАВО, РІЗКО ВПРАВО. Зв'язок між входом та виходом запам'ятовується в таблиці нечітких правил.  Таблиця нечітких правил  http://www.victoria.lviv.ua/html/oio/images/theme11/image075.gif  Кожний запис у даній таблиці відповідає своєму нечіткому правилу, наприклад: Якщо дистанція близько і напрямок правий , тоді рульовий кут різко вліво  Таким чином, мобільний робот з нечіткою логікою буде працювати за наступним принципом: дані з сенсорів про відстань до перешкоди та напрямок до неї будуть фазіфіковані, оброблені згідно табличних правил, дефазіфіковані і дані, що отримані, у вигляді керуючих сигналів надходять на приводи робота.  **Нечітка логіка в Matlab**  Fuzzy logic toolbox - вбудована в Matlab сукупність функцій, що містить набір засобів, які дозволяють:   * створювати і редагувати нечіткі системи всередині середовища Matlab; * вбудовувати нечітку підсистему в SimuLink (поставляється з Matlab) при моделюванні загальної системи; * побудувати нечітку систему в Matlab у вигляді процедури, що викликається з програми, яка написана на мові Сі.   Даний набір інструментів забезпечує три категорії інструментальних засобів програмування нечітких систем:   * функції командного рядка (command line functions); * графічний інтерактивний інтерфейс; * використання вбудованих блоків SimuLink.   Перша категорія - готові функції, які можна викликати відразу з командного рядка Matlab. Практично усі вони являють собою м-файли, що містять послідовність виразів, що виконують спеціалізований нечіткий алгоритм. Для перегляду вихідного коду функцій необхідно набрати в командному рядку:  type ім'я\_функції  Крім того, Matlab дозволяє їх модифікувати шляхом копіювання і перейменування відповідного файлу та наступного його редагування. Таким чином, нечіткий набір інструментів є розширеним власними функціями.  Друга категорія дозволяє отримати доступ до тих самих функцій через графічний користувальницький інтерфейс, за допомогою якого набагато зручніше конструювати й аналізувати нечіткі системи.  Третя категорія - моделювання в середовищі SimuLink. Тут підсистеми представляються у виді блоків - можна з'єднати будь-яким чином і відразу отримати результати.  У Matlab є багато вбудованих функцій приналежності, зокрема:   * сигмоїдальна; * двостороння сигмоїдальна; * гаусова; * дзвоноподібної форми * S-функція приналежності; * Z-функція приналежності; * трапецієподібна; * трикутна й ін.   Усі дії над нечіткими числами задаються мінімальним набором функцій і відбуваються всередині програми. Таким чином, користувачу необов'язково вивчати усі тонкощі теорії нечітких множин, достатньо лише визначити усі вхідні і вихідні змінні і задати таблицю правил, а решту роботи робить Matlab. Дефазіфікація виконується в один з п'ятьох методів, зазначених програмістом. Крім того, можна вивести на екран відповідно до введених правил результуючі поверхні керування в залежності від комбінації входів, схему отриманої нечіткої програми, і це лише мала частина всіх можливостей даного набору інструментів.  **Переваги нечітких систем**  Коротко перелічимо відмітні переваги fuzzy-систем у порівнянні з іншими:   * можливість оперувати вхідними даними, заданими нечітко: наприклад, що безупинно змінюються в часі значення (динамічні задачі), значення, що неможливо задати однозначно (результати статистичних опитувань, рекламні компанії і т.д.); * можливість нечіткої формалізації критеріїв оцінки і порівняння: оперування критеріями "більшість", "можливе", переважно" і т.д.; * можливість проведення якісних оцінок як вхідних даних, так і виведених результатів: ви оперуєте не тільки власне значеннями даних, але їхнім ступенем вірогідності (не плутати з імовірністю!) і її розподілом; * можливість проведення швидкого моделювання складних динамічних систем і їхній порівняльний аналіз із заданим ступенем точності: оперуючи принципами поведінки системи, описаними fuzzy-методами, ви по-перше, не витрачаєте багато часу на з'ясування точних значень змінних і складання рівнянь, що їх описують, по-друге, можете оцінити різні варіанти вихідних значень.   **Застосування нечітких систем**  Що стосується вітчизняного ринку комерційних систем на основі нечіткої логіки, то його формування почалося в середині 1995 року. Найбільш популярні в замовників наступні пакети:   * CubiCalc 2.0 RTC - одна з найбільш могутніх комерційних експертних систем на основі нечіткої логіки, що дозволяє створювати власні прикладні експертні системи ; * CubiQuick - дешева <університетська> версія пакета CubiCalc ; * RuleMaker - програма автоматичного витягу нечітких правил із вхідних даних ; * FuziCalc - електронна таблиця з нечіткими полями, що дозволяє робити швидкі оцінки при неточно відомих даних без нагромадження похибки; * OWL - пакет, що містить вихідні тексти усіх відомих видів нейронних мереж, нечіткої асоціативної пам'яті і т.д.   Основними споживачами нечіткої логіки на ринку СНД є банкіри і фінансисти, а також фахівці в області політичного й економічного аналізу. Вони використовують CubiCalc для створення моделей різних економічних, політичних, біржових ситуацій. Що ж стосується легкого в освоєнні пакета FuziCalc, то він зайняв своє місце на комп'ютерах великих банкірів і фахівців з надзвичайних ситуацій - тобто тих, для кого найбільше важлива швидкість проведення розрахунків в умовах неповноти і неточності вхідної інформації. Однак можна з упевненістю сказати, що епоха розквіту прикладного використання нечіткої логіки на вітчизняному ринку ще попереду.  Сьогодні елементи нечіткої логіки можна знайти в десятках промислових виробів - від систем керування електропоїздами і бойовими вертольотами до пилососів і пральних машин. Рекламні кампанії багатьох фірм (переважно японських) підносять успіхи у використанні нечіткої логіки як особливу конкурентну перевагу. Без застосування нечіткої логіки немислимі сучасні ситуаційні центри керівників західних країн, у яких приймаються ключові політичні рішення і моделюються всілякі кризові ситуації. Одним із вражаючих прикладів масштабного застосування нечіткої логіки стало комплексне моделювання системи охорони здоров'я і соціального забезпечення Великобританії (National Health Service - NHS), що вперше дозволило точно оцінити й оптимізувати витрати на соціальні нестатки .  Не обійшли засоби нечіткої логіки і програмні системи, що обслуговують великий бізнес. Першими, зрозуміло, були фінансисти, задачі яких вимагають щоденного прийняття правильних рішень у складних умовах непередбаченого ринку. Перший рік використання системи Fuji Bank приносив банку в середньому $770000 на місяць (і це тільки офіційно оголошений прибуток !).  Слідом за фінансистами, стурбовані успіхами японців і втратою стратегічної ініціативи, когнітивними нечіткими схемами зацікавилися промислові гіганти США. Motorola, General Electric, Otis Elevator, Pacific Gas & Electric, Ford і інші на початку 90-х почали інвестувати в розробку виробів, що використовують нечітку логіку. Маючи солідну фінансову "підтримку", фірми, що спеціалізуються на нечіткій логіці, одержали можливість адаптувати свої розробки для широкого кола застосувань. "Зброя еліти" вийшла на масовий ринок.  Серед лідерів нового ринку виділяється американська компанія Hyper Logic, заснована в 1987 році Фредом Уоткинсом (Fred Watkins). Спочатку компанія спеціалізувалася на нейронних мережах, однак незабаром цілком сконцентрувалася на нечіткій логіці. Недавно вийшла на ринок друга версія пакета CubiCalc фірми HyperLogic, яка є однієї з найбільш могутніх експертних систем на основі нечіткої логіки. Пакет містить інтерактивну оболонку для розробки нечітких експертних систем і систем керування, а також run-time модуль, що дозволяє оформляти створені користувачем системи у виді окремих програм.  Крім Hyper Logic серед "патріархів" нечіткої логіки можна також назвати такі фірми як IntelligenceWare, InfraLogic, Aptronix. Усього ж на світовому ринку представлено більш 100 пакетів, які тим чи іншим видом використовують нечітку логіку. У трьох десятках СУБД реалізована функція нечіткого пошуку. Власні програми на основі нечіткої логіки анонсували такі гіганти як IBM, Oracle і інші.  На принципах нечіткої логіки побудований і один з російських програмних продуктів - відомий пакет "Бізнес-прогноз". Призначення цього пакета - оцінка ризиків і потенційної прибутковості різних бізнес-планів, інвестиційних проектів і просто ідей щодо розвитку бізнесу. "Ведучи" користувача по сценарію його задуму, програма задає ряд питань, що допускають як точні кількісні відповіді, так і наближені якісні оцінки - типу "малоймовірно", "ступінь ризику високий" і т.д. Узагальнивши всю отриману інформацію у виді єдиної схеми бізнесу-проекту, програма не тільки виносить остаточний вердикт про ризикованість проекту й очікуваних прибутків, але і вказує критичні точки і слабкі місця в авторському задумі. Від аналогічних іноземних пакетів "Бізнес-прогноз" відрізняється простотою, дешевиною і, зрозуміло, російськомовним інтерфейсом. Утім, цілком очевидно, що програма "Бізнес-прогноз" - лише перша ластівка, за якої неминуче підуть нові розробки вчених СНД. | |

http://www.victoria.lviv.ua/html/oio/index.html