МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДВНЗ "Ужгородський національний університет"
Факультет інформаційних технологій
Кафедра інформаційних управляючих систем та технологій

В. М. Коцовський

Інтелектуальні інформаційні системи

Конспект лекцій

Ужгород – 2019

ЗМІСТ

1. ІНФОРМАЦІЙНІ СИСТЕМИ

Згідно до стандарту ISO 2382-1 інформаційна система (ІС) — система обробки інформації, що працює спільно з організаційними ресурсами, такими як люди, технічні засоби та фінансові ресурси, які забезпечують і розподіляють інформацію. Інше визначення: ІС — сукупність апаратно-програмних та організаційних засобів для збереження та обробки інформації з метою забезпечення інформаційних потреб користувачів.

Основним завданням ІС є забезпечення конкретних інформаційних потреб у межах певної предметної області. Під *предметною областю* розуміють деяку частину реального світу. Переважна більшість сучасних ІС включають до свого складу бази даних та СУБД, тому на практиці часто синонімом до терміну ІС є термін «система баз даних».

1.1. Класифікація інформаційних систем

Виділяють кілька рівнів класифікації.

По розподіленості:

1. настільні або локальні ІС (усі компоненти знаходяться на одному комп'ютері);
2. розподілені ІС:
   a. файл-серверні (БД знаходиться на сервері, СУБД та клієнтські програми — на робочих станціях);
   b. клієнт-серверні (БД та СУБД знаходяться на сервері, клієнтські програми — на робочих станціях).

По ступеню автоматизації ІС:

1. автоматизовані ІС (автоматизація неповна і необхідним є постійне втручання персоналу);
2. автоматичні ІС (автоматизовані).

По характеру обробки даних:

1. Інформаційно-пошукові (мета — пошук та видача інформації);
2. ІС обробки даних (АСУ та СППР).

По охопленню задач:

1. персональні;
2. групові;
3. корпоративні.

По сфері застосування:

1. навчальні;
2. військові;
3. економічні;
4. медичні;
5. географічні;
6. правові і т.п.

2. ІНТЕЛЕКТУАЛЬНІ ІНФОРМАЦІЙНІ СИСТЕМИ

Інтелектуальна інформаційна система (ІІС) — це один з видів автоматизованих інформаційних систем, інколи ІІС називають системою, засновану на знаннях. ІІС є комплексом програмних, лінгвістичних і логіко-математичних засобів для реалізації

основного завдання: здійснення підтримки діяльності людини і пошуку інформації в режимі розширеного діалогу на природній мові.

2.1. Класифікація ІІС

- Експертні системи:
  Власне Експертні системи (ЕС)
  Інтерактивні банери (web + ЕС)
- Системи спілкування
  Інтелектуальні пошукові системи
  Віртуальні співрозмовники

ІІС можуть розміщуватися на якому-небудь сайті, де користувач ставить системі питання на природній мові або, відповідаючи на питання системи, знаходить необхідну інформацію (якщо це експертна система). Але, як правило, ЕС в інтернеті виконують рекламно-інформаційні функції (інтерактивні банери), а серйозні системи (такі, як, наприклад, ЕС діагностику устаткування) використовуються локально, оскільки виконують конкретні специфічні завдання.

Інтелектуальні пошукові системи відрізняються від віртуальних співрозмовників тим, що вони досить безликі і у відповідь на питання видають деякий витяг з джерел знань (інколи досить великого обсягу), а співрозмовники володіють «характером», особливою манерою спілкування (можуть використовувати сленг, ненормативну лексику), і їхні відповіді мають бути гранично лаконічними (інколи навіть просто у формі смайликів, якщо це відповідає контексту :-)).

Для розробки ІІС раніше використовувалися логічні мови (Пролог, Лісп і т. д.), а зараз використовуються різні процедурні та об’єктно-орієнтовані мови. Логіко-математичне забезпечення розробляється як для самих модулів систем, так і для стикування цих модулів. Проте на сьогоднішній день не існує універсальної логіко-математичної системи, яка могла б задовольнити потреби будь-якого розробника ІІС, тому доводиться або комбінувати накопичений досвід, або розробляти логіку системи самостійно. В області лінгвістики теж існує безліч проблем, наприклад, для забезпечення роботи системи в режимі діалогу з користувачем на природній мові необхідно закласти в систему алгоритми формалізації природної мови, а це завдання виявилося куди складнішим, ніж передбачалося на світанку розвитку інтелектуальних систем. Ще одна проблема — постійна мінливість мови, яка обов'язково має бути відбита в системах штучного інтелекту.

Забезпечення роботи ІІС

- Математичне
- Лінгвістичне
- Програмне
- Технічне
- Технологічне
- Кадрове

2.2. Класифікація задач, які вирішують ІС

- *Інтерпретація данних*. Це одне з традиційних завдань для експертних систем. Під інтерпретацією розуміється процес визначення змісту даних, результати якого мають бути погодженими і коректними. Зазвичай передбачається багатоваріантний аналіз даних.
- *Діагностика*. Під діагностикою розуміється процес співвідношення об'єкта з деяким класом об'єктів і виявлення несправності в деякій системі. Несправність — це відхилення від норми. Таке трактування дозволяє з єдиних теоретичних позицій розглядати і несправність устаткування в технічних системах, і захворювання живих організмів, і всілякі природні аномалії. Важливою специфікою є тут необхідність розуміння функціональної структури («анатомії») діагностуючої системи.
- *Моніторинг.* Основне завдання моніторингу — безперервна інтерпретація даних у реальному масштабі часу і сигналізація про вихід тих або інших параметрів за допустимі межі. Головні проблеми — «пропуск» тривожної ситуації і інверсне завдання «помилкового» спрацьовування. Складність цих проблем в нечіткості симптомів тривожних ситуацій і необхідність обліку тимчасового контексту.
- *Проектування.* Проектування полягає в підготовці специфікацій на створення «об'єктів» із заздалегідь визначеними властивостями. Під специфікацією розуміється весь набір необхідних документів — креслення, записка пояснення і так далі Основні проблеми тут — здобуття чіткого структурного опису знань про об'єкт і проблема «сліду». Для організації ефективного проектування і в ще більшій мірі того, що перепроектувало необхідно формувати не лише самі проектні рішення, але й мотиви їхнього прийняття. Таким чином, в завданнях проектування тісно зв'язуються два основні процеси, які виконуються в рамках відповідної ЕС: процес виведення рішення і процес пояснення.
- *Прогнозування.* Прогнозування дозволяє передбачати наслідки деяких подій або явищ на підставі аналізу наявних даних. Прогнозуючі системи логічно виводять ймовірні наслідки із заданих ситуацій. У прогнозуючій системі зазвичай використовується параметрична динамічна модель, в якій значення параметрів «підганяються» під задану ситуацію. Висновки, що виводяться з цієї моделі, складають основу для прогнозів з ймовірними оцінками.
- *Планування.* Під плануванням розуміється знаходження планів дій, що відносяться до об'єктів, здатних виконувати деякі функції. У таких ЕС використовуються моделі поведінки реальних об'єктів з тим, аби логічно вивести наслідки планованої діяльності.
- *Навчання.* Під навчанням розуміється використання комп'ютера для навчання деякої дисципліни або предмету. Системи навчання діагностують помилки при вивченні якої-небудь дисципліни за допомогою ЕОМ і підказують правильні рішення. Вони акумулюють знання про гіпотетичного «учня» і його характерні помилки, потім у роботі вони здатні діагностувати слабкості в знаннях учнів і знаходити відповідні засоби для їхньої ліквідації. Крім того, вони планують акт спілкування з учнем залежно від успіхів учня з метою передачі знань.
- *Керування.* Під керуванням розуміється функція організованої системи, що підтримує певний режим діяльності. Такого роду ЕС здійснюють управління поведінкою складних систем відповідно до заданих специфікацій.

- *Підтримка прийняття рішень.* Підтримка прийняття рішень — це сукупність процедур, що забезпечує особу, що приймає рішення, необхідною інформацією і рекомендаціями, що полегшують процес ухвалення рішення. Ці ЕС допомагають фахівцям вибрати і сформувати потрібну альтернативу серед множини виборів при ухваленні відповідальних рішень.

У загальному випадку всі системи, засновані на знаннях, можна поділити на системи, що вирішують задачі аналізу, і на системи, які вирішують задачі синтезу. Основна відмінність задач аналізу від задач синтезу полягає в тому, що якщо в задачах аналізу множина рішень може бути перерахована і включена в систему, то в задачах синтезу множина рішень потенційно необмежена. Задачею аналізу є: інтерпретація даних, діагностика, підтримка прийняття рішення; до завдань синтезу відносять проектування, планування, управління. Комбіновані задачі: навчання, моніторинг, прогнозування.

3. БАЗОВІ ПОНЯТТЯ ШТУЧНОГО ІНТЕЛЕКТУ

3.1. Означення та історія виникнення

Штучний інтелект (ШІ, англ. Artificial intelligence) — *наука та технологія створення інтелектуальних машин, в особливості інтелектуальних комп'ютерних програм.* ШІ пов'язаний з завданням використання комп'ютерів для розуміння людського інтелекту, але не обов'язково обмежується біологічно правдоподібними методами (Джон Маккарті, 1956 р., конференція у Дартмутському університеті). В подальшому було зроблено чимало спроб дати формальне визначення інтелекту взагалі і інтелекту штучного зокрема. Найбільш відомим, очевидно, є визначення предмету теорії штучного інтелекту [1], що було дане видатним дослідником у галузі штучного інтелекту М. Мінські і яке у більш або менш видозміненому вигляді потрапило до словників та енциклопедій: "*штучний інтелект є дисципліна, що вивчає можливість створення програм для вирішення задач, які при вирішенні їх людиною потребують певних інтелектуальних зусиль*". Це визначення зустріло критику, яка полягала в тому, що під нього можна підвести що завгодно, наприклад, виконання простих арифметичних операцій. Відтак до цього визначення додається поправка: "*сюди не входять задачі, для яких відома процедура їх вирішення*". Таке визначення також важко вважати задовільним.

Рассел та Норвіг [2] наводять класифікацію означень ШІ у табличній формі

Системи, які мислять подібно до людини	Системи, які мислять раціонально
Системи, які діють подібно до людини	Системи, які діють раціонально

Єдиної відповіді на питання чим займається ШІ, не існує. Майже кожен автор дає своє визначення. Зазвичай ці визначення зводяться до *наступних*:

- штучний інтелект вивчає методи розв'язання задач, які потребують людського розуміння. Тут мова іде про те, щоб навчити ШІ розв'язувати тести інтелекту. Це передбачає розвиток способів розв'язання задач за аналогією, методів дедукції та індукції, накопичення базових знань і вміння їх використовувати.
- штучний інтелект вивчає методи розв'язання задач, для яких не існує способів розв'язання або вони не коректні (через обмеження в часі, пам'яті тощо). Завдяки такому визначенню інтелектуальні алгоритми часто використовуються для розв'язання NP-повних задач, наприклад, задачі комівояжера.
- штучний інтелект займається моделюванням людської вищої нервової діяльності.
- штучний інтелект — це системи, які можуть оперувати з знаннями, а найголовніше — навчатися. В першу чергу мова ведеться про те, щоби визнати клас експертних систем (назва походить від того, що вони спроможні замінити «на посту» людей-експертів) інтелектуальними системами.

Останнє визначення, що з'явилося у 1990-х рр., засноване на так званому агентно-орієнтованому підході. Цей підхід акцентує увагу на тих методах і алгоритмах, які допоможуть інтелектуальному агенту виживати в оточуючому середовищі під час виконання свого завдання. Тому тут значно краще вивчаються алгоритми пошуку і прийняття рішення.

Джек Коупленд у праці "Що таке ШІ" відзначає, що незважаючи на наявність численних підходів та визначень як до розуміння ШІ, так і до створення інтелектуальних інформаційних систем, можна виділити два **основні підходи щодо розроблення систем ШІ**:

1) низхідний (Top-Down AI, семіотичний, символьний) — створення експертних систем, баз знань та систем логічного виведення, що моделюють та імітують високорівневі психічні процеси: мислення, міркування, мова, емоції, творчість і т.п.;
2) висхідний (Bottom-Up AI, біологічний, конекціоністський) — вивчення нейронних мереж і еволюційних обчислень, які моделюють інтелектуальну поведінку на основі біологічних елементів, а також створення відповідних обчислювальних систем, таких як нейрокомп'ютери.

Другий підхід, власне кажучи не відноситься до визначення ШІ, яке дав Дж. Маккарті. Їх поєднує тільки кінцева мета.

Відсутність чіткого визначення ШІ не заважає оцінювати *інтелектуальність* на *інтуїтивному* рівні. Можна навести як мінімум два методи такої оцінки:

метод експертних оцінок. Рішення про ступінь інтелектуальності приймає досить велика група експертів (незалежно або у взаємодії між собою);

метод тестування. Існує значна кількість так званих інтелектуальних тестів, апробованих практикою, і ці тести широко використовуються для оцінки рівня розумових здібностей людини, а також у психології та психіатрії.

Для прикладу наведемо декілька типових тестових завдань.

Приклад 1. Вставте слово, яке означає те саме, що і два слова поза дужками:

дерево (….) підробка

Приклад 2. Вставте число, яке пропущене:

36 30 24 18 _

Приклад 3. Викресліть зайве слово:

лев лисиця жираф щука собака

І метод експертних оцінок, і метод тестування мають свої недоліки. Головний з них полягає у тому, що оцінка дається виходячи лише з власних уявлень експертів, авторів тестів і т.п. про те, як має бути. Тому ці способи не дуже придатні для оцінки будь-якого інтелекту, крім людського.

Серед психіатрів можна почути вислів: "він міркує логічно вірно, але неправильно". Наприклад, дається тестове завдання: "серед слів соловей, чапля, перепілка, стілець, шпак виділити зайве".

Більшість людей, не задумуючись, дає відповідь стілець, тому що всі інші слова — це назви птахів. І раптом хтось дає відповідь шпак, пояснюючи це тим, що це єдине слово, в якому відсутня літера "л".

Обидві класифікації є логічно вірними і формально рівноправними. Але чому ж перевага віддається одній з них? Тому, що так міркує більшість людей. А чому так міркує більшість людей? Очевидно, тому, що *перша класифікація вважається більш важливою для людської практики*. Це положення приймається на аксіоматичному рівні, без доведення. Але те, що є більш важливим для людської практики, зовсім не обов'язково буде так само важливим для розумного робота або для інопланетянина.

Необхідно підкреслити, що поняття “штучний інтелект” не можна зводити лише до створення пристроїв, які імітують людину в усій повноті її діяльності. Насправді ж, спеціалісти які працюють в цій області вирішують іншу задачу: виявити механізми, які лежать в основі діяльності людини, щоб застосувати їх при вирішенні конкретних науково-технічних задач**.** І це лише одна з можливих проблем.

3.2. Приклади інтелектуальних задач

До переліку інтелектуальних задач можна віднести [1]:

- розпізнавання образів;
- логічне мислення;
- навчання і самонавчання;
- аналіз ситуації;
- розуміння нової інформації;
- планування цілеспрямованих дій;

Більш детально зупинимося на перших трьох задачах.

3.2.1. Розпізнавання

На інтуїтивному рівні можна сформулювати декілька типових задач розпізнавання:

- *ідентифікувати* об’єкт, що спостерігається людиною, тобто вирізнити його серед інших (наприклад, побачивши іншу людину, впізнати у ній свою дружину);
- здійснити розпізнавання у класичній постановці, тобто віднести об’єкт, що спостерігається людиною, до одного з заздалегідь відомих класів об’єктів (наприклад, відрізнити легковий автомобіль від вантажного);
- провести кластеризацію (розбиття множини об’єктів на класи);

Людина робить класифікацію просто. Чоловік, повернувшись додому, відразу ж пізнає свою дитину, собаку і т.д. Але він рідко може *пояснити, як* він це робить. Якби це можна було зробити, алгоритми розпізнавання можна було б легко програмувати і широко застосовувати.

Теорія розпізнавання, яка інтенсивно розвивається, необхідна для того, щоб навчити вирішувати задачі розпізнавання і штучні інтелектуальні системи. Зокрема, сформульовано такий ключовий принцип:

будь-який об’єкт у природі – унікальний; унікальні об`єкти – типізовані.

У відповідності до цього принципу, розпізнавання здійснюється на основі аналізу певних характерних *ознак*. Вважається, що в природі не існує двох об’єктів, для яких співпадають *абсолютно всі* ознаки, і це теоретично дозволяє здійснювати ідентифікацію. Якщо ж для деяких об’єктів співпадають *деякі* ознаки, ці об’єкти теоретично можна об’єднувати в групи, або класи, за цими співпадаючими ознаками. "Близькість" значень ознак дає змогу проводити розбиття множини на кластери.

Проблема полягає у тому, що різноманітних ознак дуже багато. Незважаючи на легкість, з якою людина проводить розпізнавання, вона дуже рідко в змозі виділити ознаки, суттєві для цього. До того ж, об’єкти, як правило, змінюються з часом.

Розпізнавання об’єктів і ситуацій має виняткове значення для орієнтації людини в навколишньому світі і для прийняття правильних рішень. Розпізнавання, як правило, здійснюється людиною на інтуїтивному, підсвідомому рівні, людина навчилася цьому

за мільйони років еволюції. На цьому фоні спроби навчити розпізнаванню складних об'єктів штучні інтелектуальні системи, навіть шляхом автоматизованого виділення характерних ознак, мають досить слабкі досягнення.

3.2.2. Логічне мислення

Логічне мислення — перехід від початкових положень до їх наслідків за формалізованими законами логіки. І тут пересічна людина рідко в змозі пояснити, за якими алгоритмами вона здійснює логічні побудови. Але методики, за якими можна автоматизувати логічне мислення, досить відомі.

Перш за все, це формальна логіка Аристотеля на основі конструкцій, які отримали назву *силогізмів*. Класичний приклад.

Перше положення. *Усі люди смертні.* Друге положення. *Сократ — людина.*

Висновок: *Сократ смертний.*

Якщо перше та друге положення у силогізмі істинні та задовольняють певним загальним формальним вимогам, тоді і висновок буде істинним незалежно від змісту тверджень, що входять до силогізму.

Виконання формальних вимог є важливим, інакше легко припуститися логічних помилок, подібних до таких:

Всі студенти вузу А знають англійську мову. Петров знає англійську мову. Отже, Петров — студент вузу А.

Або: Іванов не готувався до іспиту і отримав двійку. Сидоров не готується до іспиту. Отже, і Сидоров отримає двійку.

Аристотелем було запропоновано декілька формальних конструкцій силогізмів, які він вважав достатньо універсальними. У XIX столітті почала розвиватися сучасна математична логіка, яка розглядає аристотелевські силогізми як один із часткових випадків.

Основою більшості сучасних систем, призначених для автоматизації логічних побудов, є *метод резолюцій* Робінсона. Але практична автоматизація логічного мислення зіткнулася з двома серйозними проблемами.

Перша з них — феномен, який Р. Беллман називав *прокляттям розмірності*. Зовнішній світ являє собою винятково складне переплетіння різноманітних об'єктів та зв'язків між ними. Для того, щоб тільки ввести всю цю інформацію до пам'яті інтелектуального комп'ютера, може знадобитися не одна тисяча років.

Інша проблема — *алгоритмічна нерозв'язність*. Відомо, що в рамках будь-якої досить складної формальної теорії існують положення, які є істинними, але які не можна ні довести, ні спростувати (теорема Геделя про нерозв'язність формальної арифметики.

Реальні програми, які здійснюють логічне виведення (вони часто називаються *експертними системами*) мають досить обмежене застосування. Вони мають обмежений набір фактів та правил з певної, більш-менш чітко окресленої предметної галузі і можуть використовуватися лише у цій галузі.

Що ж стосується людини, то якість її логічного мислення часто буває далеким від бездоганного. Люди рідко проводять логічні побудови до кінця, часто роблять логічні помилки, а інколи взагалі керуються принципами, невірними з точки зору нормальної логіки, а рішення приймається на підсвідомому, інтуїтивному рівні. Зрозуміло, що таке рішення може бути помилковим. Але, якби це було не так, людина була б практично не здатною ні до якої діяльності — ні до фізичної, ні до продуктивної розумової.

Для задач, які розглядалися вище, характерною була спільна риса: їх *погана формалізованість*, відсутність або невизначеність чітких алгоритмів вирішення. Саме такі задачі і являють собою основний предмет розгляду в теорії штучного інтелекту.

До зовсім іншого класу відносяться *обчислювальні задачі*. Важко відповісти на запитання, як саме людина здійснює ті чи інші обчислення. Добре відомими є і низька швидкість, і невисока надійність цього виду людської діяльності. Але були запропоновані ефективні принципи комп'ютерних обчислень.

3.2.3. Навчання

Машинне навчання (machine learning) — це підгалузь штучного інтелекту, яка часто застосовує статистичні прийоми для надання комп'ютерам здатності «навчатися» (тобто, поступово покращувати вміння розв'язувати певну задачу).

Назву було запропоновано 1959 року А. Семюелем [2]. Машинне навчання досліджує вивчення та побудову алгоритмів, які можуть навчатися й робити передбачення та узагальнення на основі наявних даних [3]. Машинне навчання застосовують в обчислювальних задачах, в яких розробка та програмування явних алгоритмів з доброю продуктивністю є складною або нездійсненною; до прикладів застосувань належать фільтрування електронної пошти, виявляння мережевих атак, оптичне розпізнавання символів (ОРС), навчання ранжуванню та комп'ютерний зір.

Задачі машинного навчання, як правило, поділяють на дві широкі категорії, залежно від того, чи доступний системі, що навчається, навчальний «сигнал», або «зворотний зв'язок»:

- *Навчання з учителем* (supervised learning): комп'ютерові надають приклади входів та їхніх бажаних виходів, задані «вчителем», і метою є навчання загального правила, яке відображає входи на виходи. В окремих випадках вхідний сигнал може бути доступним лише частково, або бути обмеженим особливим зворотним зв'язком:

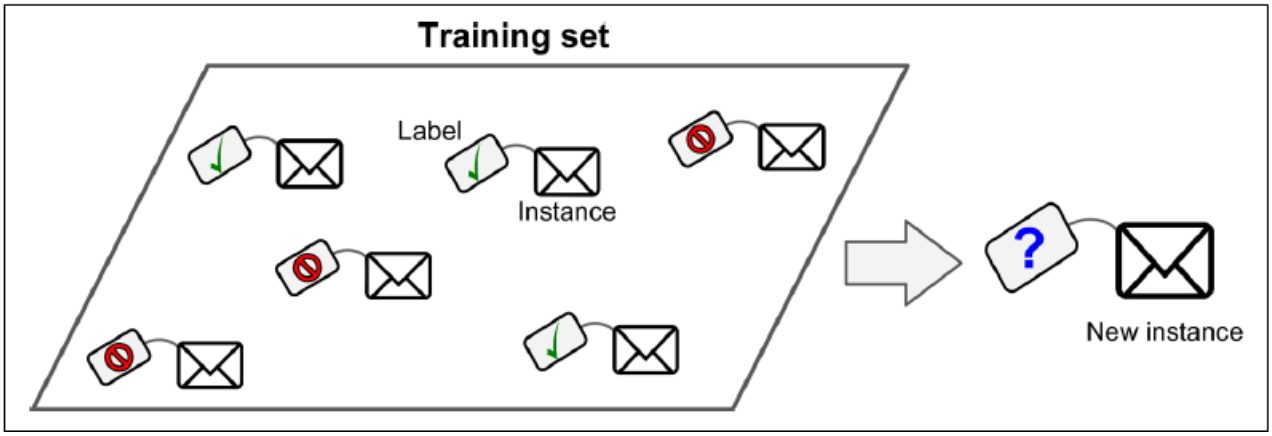


Рис. 1.1. Помічена навчальна вибірка для класифікації електронних листів

- *Напівавтоматичне навчання* (semi-supervised learning): комп'ютерові дають лише неповний тренувальний сигнал: тренувальний набір, в якому відсутні деякі (часто численні) цільові виходи.
- *Навчання з підкріпленням* (reinforcement learning): тренувальні дані (у вигляді винагород та покарань) надаються лише як зворотний зв'язок на дії

програми в динамічному середовищі, як при керуванні автомобілем, або грі з опонентом.

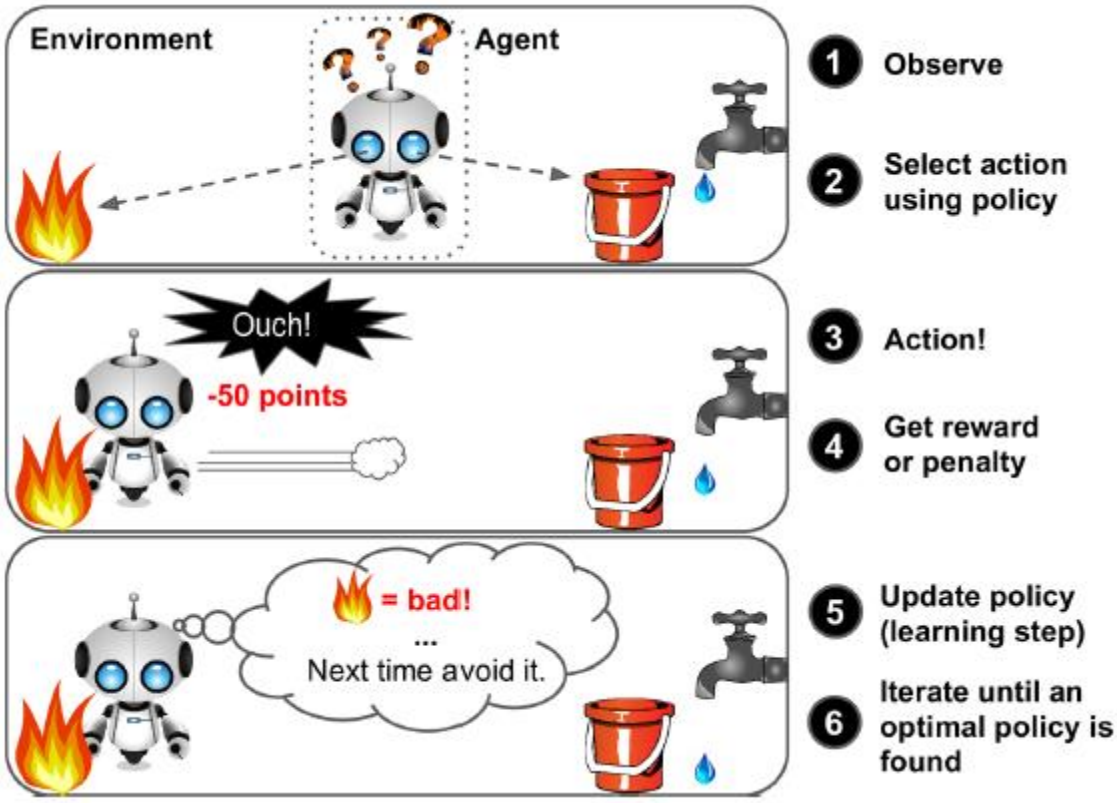


Рис. 1.2. Навчання з підкріпленням

Навчання з підкріпленням застосовується у робототехніці.

- *Навчання без учителя* (unsupervised learning): Алгоритмові навчання не дається міток, залишаючи його самому знаходити приховані закономірності в наборах даних.

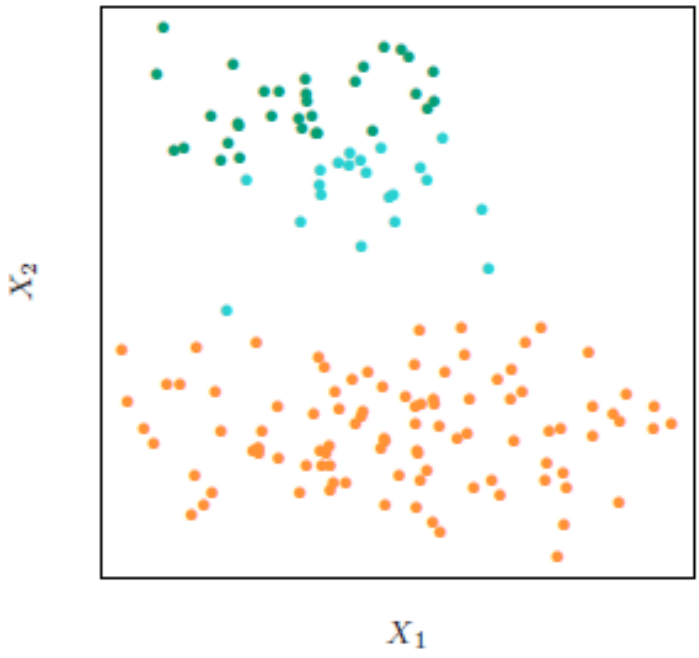


Рис. 1.3. Навчання без учителя

3.3. Огляд популярних інтелектуальних ІС

1. **Google Brain** — дослідницький проект Google по розробці ШІ на основі глибинного навчання (Deep Learning). В ньому поєднуються відкриті дослідження в галузі машинного навчання з розробкою систем та використанням обчислювальних можливостей в масштабах Google.

У жовтні 2016 р. Google Brain провів експеримент щодо шифрування повідомлень. У ньому два набори ШІ розробили свої власні криптографічні алгоритми для захисту своїх повідомлень від іншого ШІ, який в свою чергу спрямовані на розвиток власної системи для злому шифрування, створеного ШІ. Дослідження виявилося успішним, оскільки два первинних ШІ змогли з нуля навчитися спілкуванню один з одним.

В даному експерименті були створені три ШІ: Аліса, Боб і Єва. Мета експерименту полягала в тому, щоб Аліса послала повідомлення Бобу, який зможе його розшифровувати, а Єва спробувала б перехопити дане повідомлення. При цьому ШІ не давалося чіткі інструкції про те, як шифрувати їх повідомлення. Їм була надана тільки функція втрат. Наслідком цього стало те, що якщо під час експерименту спілкування між Алісою і Бобом не увінчалися успіхом (повідомлення Аліси було неправильно витлумачено Бобом або перехоплено Євою), то в наступних раундах криптографічне правило змінюється таким чином, щоб Аліса і Боб змогли безпечно спілкуватися. Дослідження дозволило зробити висновок про те, що ШІ може розробити власну систему шифрування без заздалегідь прописаних алгоритмів шифрування, що може стати проривом в області шифрування повідомлень в майбутньому.

У лютому 2017 р. система Google Brain анонсувала систему поліпшення зображення, що використовує нейронні мережі для заповнення деталей зображень з дуже низькою роздільною здатністю. Зображення з роздільною здатністю 8×8 перетворюються в зображення з роздільною здатністю 32×32.

Було отримано прорив в покращенні зображень з низькою роздільною здатністю. Коли людям показували поліпшене зображення і справжнє зображення, то в 10% випадків для фотографій знаменитостей і в 28% випадків для фотографій спалень вони не могли правильно сказати, де справжнє, а де масштабоване зображення. До цього «звичайне» бікубічне масштабування завжди правильно визначалося людиною.

У вересні 2016 р. команда запустила нову систему — нейронний машинний переклад Google (GNMT), яка являє собою наскрізну систему навчання, здатну вчитися на великій кількості прикладів. Хоча її впровадження значно підвищило якість Перекладача Google для пілотних мов, було дуже складно створити такі поліпшення для всіх 103 підтримуваних мов. Для вирішення даного завдання команда Google Brain змогла розробити багатомовну версію GNMT, яка розширила попередню і дозволила здійснювати переклад між декількома мовами. Більш того, вона дозволила виконувати прямий переклад між мовними парами, які явно не задавалися при навчанні. Нещодавно Google анонсувала, що Перекладач Google може здійснювати переклад за допомогою нейронних мереж без розшифровки тексту. Це означає, що можна перевести запис, записаний на одній мові, в текст на іншій мові без попереднього перетворення мови в текст. Щоб навчити цьому систему, їй подали на вхід багато фрагментів записів іспанською мови з розшифровкою англійською мовою. Різні шари нейронних мереж, які імітують людський мозок, змогли об'єднати відповідні фрагменти і послідовно перетворити звукову хвилю в англійський текст.

В даний час технологія проекту використовується в системі розпізнавання мови в операційній системі Android, пошуку по фотографіях в Google+ і рекомендаціях відео в YouTube.

У роботі над Google Brain використовується TensorFlow — відкрита програмна бібліотека для машинного навчання, розроблена компанією Google для вирішення задачі конструювання і тренування нейронної мережі з метою автоматичного знаходження та класифікації образів, досягаючи якості людського сприйняття. Застосовується як для досліджень, так і для розробки власних продуктів Google. Основне API для роботи з бібліотекою реалізовано для Python, також існують реалізації для C ++, Haskell, Java, Go і Swift.

Серед додатків, для яких TensorFlow є основою — програмне забезпечення автоматизованої анотації зображень DeepDream. У 2015 р. Google на базі TensorFlow реалізувала RankBrain — системе, що самонавчається і використовується пошуковиком Google для забезпечення більш релевантних результатів пошуку для користувачів. На сьогодні RankBrain є третім найбільш важливим фактором в алгоритмі ранжування корпорації Google нарівні з посиланнями і контентом.

RankBrain інтерпретує запити користувачів і надає найбільш релевантні сторінки, які можуть і не містити саме ті слова, які включені в пошуковий запит, але мають аналогічний сенс.

В автономному (офлайн) режимі RankBrain отримує дані про минулі пошукові запити і, аналізуючи їх, дізнається, як налаштувати результати пошуку. Після того як результати RankBrain перевіряються командою Google, система оновлюється і знову працює в режимі реального часу.

2. **IBM Watson** — суперкомп'ютер фірми IBM, оснащений системою штучного інтелекту. Його створення — частина проекту DeepQA. Основне завдання Уотсона — розуміти питання, сформульовані на природній мові, і знаходити на них відповіді в базі даних з використанням великої кількості алгоритмів ймовірнісного пошуку. Він названий на честь першого президента IBM Томаса Уотсона.

Watson складається з 90 серверів IBM p750, кожен з яких оснащений чотирма восьмиядерними процесорами архітектури POWER6. Сумарний обсяг оперативної пам'яті — більше 15 терабайт.

Система мала доступ до 200 млн сторінок структурованої і неструктурованої інформації об'ємом в 4 терабайта, включаючи повний текст Вікіпедії.

У 2014 році IBM оголосила про інвестування 1 млрд доларів в розвиток проекту Watson, і про створення нового підрозділу когнітивних обчислень Watson Business Group, в завдання якого входить розробка і комерціалізація хмарних когнітивних (пізнання, вивчення) сервісів в таких областях як охорона здоров'я, фінанси, подорожі, телекомунікації та роздрібна торгівля.

Продовжуючи успішно розвивати проект IBM Watson, до 2018 року компанія випустила програмні продукти: Watson Studio — для побудови моделей машинного навчання та Watson SDK — для доступу до інтернет-сервісів IBM Watson, які доступні для операційних систем Linux, macOS і Windows.

Для демонстрації роботи Watson прийняв участь в американській вікторині «Jeopardy!», в якій учасники відповідають на питання з області загальних знань: кожне питання — твердження про якесь явище або істоту. Гравець повинен дати свою відповідь у формі запитання. Наприклад, при виборі категорії «Президенти» і питання

вартістю 200 доларів ведучий читає твердження «Цей "батько-засновник" насправді не зрубав вишню», а гравець для отримання суми в 200 доларів повинен буде відповісти «Хто такий Джордж Вашингтон?». У гравця є 5 секунд для відповіді на питання: якщо він дасть правильну відповідь, то він переходить до наступного питання. Суперниками Watson у двох іграх був володар найбільшого виграшу в програмі і рекордсмен по тривалості безпрограшної серії. Під час гри Watson не мав доступу до Інтернету. Комп'ютер здобув перемогу, отримавши 1 млн доларів (системі вдалося виграти в обох іграх).

Приклад ще одного використання проекту Watson — система генерації питань та аналізу відповідей, для діагностики онкологічних захворювань.

3. **Софія** (Sophia) — людиноподібний робот, розроблений компанією Hanson Robotics.

Рис. 1.4. Софія у 2018 році

Софія має СШІ, обладнана функціями обробки візуальної інформації і технологією розпізнавання осіб. Софія може імітувати людські жести і вирази обличчя, а також може відповідати на певні питання і проводити прості бесіди на певні теми (наприклад, про погоду). Всього Софія може імітувати 62 емоції. Робот використовує технологію розпізнавання мови від Alphabet (материнської компанії Google) і вдосконалюється з часом, стаючи розумніший. Програмне забезпечення штучного інтелекту Софії розроблено компанією SingularityNET. Воно аналізує проведені розмови і на підставі нових даних покращує відповіді в майбутньому.

У жовтні 2017 р. Софія стала громадянкою Саудівської Аравії.

3.3. Області застосування інтелектуальних ІС

Фінанси:

- *Алгоритмічна торгівля* — використання складних систем штучного інтелекту для прийняття торгових рішень зі швидкістю, що перевищує швидкість на яку здатний людський організм. Це дозволяє робити мільйони угод в день без будь-якого

втручання людини. Автоматизовані торгівельні системи зазвичай використовуються великими інституційними інвесторами.

- *Дослідження ринку та інтелектуальний аналіз даних*.

Кілька великих фінансових установ вклали кошти в розвиток ШІ, щоб використовувати його в інвестиційній практиці. Банки, такі як UBS і Deutsche Bank, використовують систему ШІ під назвою Sqreem, яка може обробляти дані для розробки профілів споживачів і зіставляти їх з продуктами, які вони, швидше за все захочуть придбати. Goldman Sachs використовує Kensho, платформу аналітики ринку, яка об'єднує статистичні обчислення з великими даними і обробкою природної мови. Його системи машинного навчання використовують дані в Інтернеті і оцінюють кореляції між світовими подіями і їх впливом на ціни фінансових активів. Інформація, витягнута системою ШІ з прямої трансляції новин, використовується в прийнятті інвестиційних рішень.

- *Андерайтинг* (банківській справі — це процедура оцінки банком ймовірності погашення або непогашення кредиту, що запитується).

Онлайн-кредитор Upstart аналізує величезна кількість споживчих даних і використовує алгоритми машинного навчання для побудови моделей кредитного ризику, які прогнозують ймовірність дефолту. Їх технологія буде ліцензована для банків, щоб вони могли використовувати її для оцінки своїх процесів.

Важка промисловість

Роботи стали поширені в багатьох галузях промисловості і часто займаються роботою, яка вважається небезпечною для людей. Роботи виявилися ефективними на робочих місцях, пов'язаних з повторюваними рутинними завданнями, які можуть привести до помилок або нещасних випадків.

Медицина

Штучні нейронні мережі, такі як технологія Concept Processing в програмному забезпеченні EMR, використовуються в якості клінічних систем прийняття рішень для медичної діагностики.

Інші застосування:

- Комп'ютерна інтерпретація медичних зображень. Такі системи допомагають сканувати цифрові зображення, наприклад від комп'ютерної томографії, для типових проявів і для виділення помітних відхилень, таких як можливі захворювання. Типовим застосуванням є виявлення пухлини.
- Аналіз серцевого ритму.
- Роботи-помічники для догляду за людьми похилого віку.
- Обробка медичних записів для надання більш корисної інформації.
- Створення планів лікування.
- Допомога в повторюваних завданнях, включаючи управління прийомом медикаментів, надання консультацій.
- Створення ліків.

В даний час в галузі охорони здоров'я працює понад 100 стартапів, заснованих на застосуванні ШІ.

Управління людськими ресурсами та рекрутинг

Інше застосування ШІ полягає в управлінні людськими ресурсами та рекрутингу. Існує три способи використання ШІ для управління людськими ресурсами та найму

фахівців. ШІ використовується для перегляду резюме і ранжування кандидатів відповідно до їх рівнем кваліфікації. ШІ також використовується для прогнозування успіху кандидата в заданих ролях через платформи зіставлення посад. І нарешті, ШІ використовується при створенні чат-ботів, які можуть автоматизувати повторювані комунікаційні завдання.

Онлайн служби підтримки клієнтів

ШІ реалізується в автоматизованих онлайн-помічників, які можна розглядати як чат-боти на веб-сторінках. Це може допомогти підприємствам знизити витрати на наймання і навчання співробітників. Основною технологією для таких систем є природна обробка мови. Google аналізує людську мову і перетворює їх в текст. Платформа може ідентифікувати сердитих клієнтів через особливості їх мови і реагувати відповідним чином.

Транспорт

Для автоматичних коробок передач в автомобілях були розроблені контролери нечіткої логіки. Автомобілі мають допоміжні функції, засновані на ШІ, такі як засоби круїз-контролю. ШІ використовується для оптимізації додатків управління дорожнім трафіком, що, в свою чергу, скорочує час очікування, споживання енергії і шкідливі викиди на 25%. В майбутньому будуть розроблені повністю автономні автомобілі. Очікується, що ШІ на транспорті забезпечить безпечне, ефективне і надійне транспортування, мінімізуючи згубний вплив на навколишнє середовище і суспільство.

Інші галузі застосування

Різні засоби ШІ також широко використовуються в області забезпечення безпеки, розпізнаванні мови і тексту, інтелектуального аналізу даних і фільтрації спаму в електронній пошті. Також розробляються програми для розпізнавання жестів (розуміння мови жестів машинами), індивідуальне розпізнавання голосу, глобальне розпізнавання голосу (для множини людей в галасливій кімнаті), розпізнавання особи для інтерпретації емоцій і невербальних сигналів.

4. ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ

4.1. Керування складними системами

4.1.1. Алгоритмічний та декларативний підходи до керування

На найбільш загальному рівні, можна виділити два підходи до керування складними системами та до програмування роботів і комп'ютерів, що могли б розв**'**язувати ті чи інші задачі. При програмуванні сучасних комп'ютерів в основному реалізований традиційний **алгоритмічний підхід**, який можна ще назвати імперативним. Цей підхід вимагає заздалегідь продумати та детально розписати, як треба вирішувати певну проблему. Написання програми вимагає задання чітких послідовностей інструкцій. Якщо таку послідовність вдається написати — комп'ютер зможе вирішувати дану задачу, якою б складною вона не була. Але, ясна річ, він виконувати інструкції, абсолютно не розуміючи їх змісту. Якщо зміняться умови, за яких виконується програма, комп'ютер може виявитися безпомічним.

Інший підхід — **декларативний.** Інтелектуальному виконавцеві (людині чи комп'ютерові) досить сказати, **що** треба робити, тобто лише сформулювати завдання, побудувавши всі взаємозв'язки між об'єктами у предметній області. **Як** це завдання буде виконуватися — повинен визначити сам виконавець.

Запустити космічний корабель так, щоб він приземлився на Марсі, взяв зразки ґрунту та привіз їх назад — задача дуже складна, але вона піддається точній алгоритмізації. Математичні методи дозволяють точно розрахувати траєкторію, по якій повинен рухатися такий корабель.

Послати робота до магазину за пляшкою молока — задача, на декілька порядків більш складна. Не кажучи вже про сам процес спілкування з продавцем, робот повинен вирішити ряд не зовсім формалізованих підзадач. Заздалегідь розрахувати траєкторію руху неможливо, оскільки робот повинен уникнути зіткнень з людьми та автомобілями. Якщо магазин закритий на переоблік, він повинен знайти інший магазин. *Неможливо передбачити всі ситуації, які можуть виникнути, а відтак — алгоритмізувати розв'язок задачі.* Тому виконання такого завдання під силу лише інтелектуальній системі, яка вміє *орієнтуватися в зовнішньому світі, аналізувати поточні ситуації та коригувати, адаптувати свою поведінку на основі такого аналізу*.

4.1.2. Формалізація понять алгоритмічності та декларативності

Спробуємо формалізувати деякі з впроваджених раніше понять. Введемо такі позначення:

q — первинні інструкції, записані без будь-яких змін у тому вигляді, в якому їх сформулював автор; аналогічно можна визначити *первинний опис ситуації* як результат її безпосереднього сприйняття;

S — множина факторів, які визначають поточний стан виконавців; розглядатимемо *S* як об'єднання двох множин: S_1 та S_2, де S_1 — множина *контрольованих факторів,* які відомі авторові процедури і на які він може мати вплив, S_2 — множина *неконтрольованих* факторів;

Z — знання, які має виконавець;

r_A — робочий алгоритм, який формується та реалізується виконавцем при алгоритмічному підході. При цьому, як правило, автоматично формулюється і програма p_A, що відповідає цьому алгоритму;

r_D — робочий алгоритм, який формується та реалізується інтелектуальним виконавцем при декларативному підході. Згідно з фундаментальними тезами Тьюринга та Черча (все, що може бути виконано будь-яким виконавцем, може бути промодельоване на машині Тьюринга) сам факт виконання завдання свідчить про існування цього алгоритму. Щоправда, цей алгоритм може і не усвідомлюватися виконавцем, а тому не може бути явно сформульований у вигляді алгоритму чи програми (якщо ж програма, що відповідає алгоритму r_D, повинна бути згенерована, йдеться мова про *задачу синтезу програм*).

Тоді можна записати такі співвідношення:

$$r_A = f\left(q, S_1\right),$$

$$r_D = g\left(q, S_1, S_2, Z\right).$$

Принциповим є наступне. Алгоритм r_A формується на основі первинних інструкцій q однозначно. При цьому також можуть відбуватися зміна і поповнення первинних інструкцій, але цей процес має повністю контрольований і детермінований характер. Як приклад можна навести компіляцію програм, написаних мовами високого рівня. Тому автор процедури може бути впевнений у гарантованому результаті (якщо, звичайно, були дотримані певні формальні вимоги до первинних інструкцій і забезпечено належний стан виконавця).

На противагу цьому, за декларативного підходу такої впевненості немає. Інтелектуальний виконавець певним чином поповнює первинні інструкції, але їх авторові не завжди відомо, яким чином це поповнення відбувається. Тому не можна бути впевненим у результаті виконання інструкцій, і ця втрата гарантованості є неминучою платою за відмову від алгоритмічності. Отже, ми маємо справу з узагальненням поняття алгоритму, яке дістало назву **"квазіалгоритм".**

4.2. Квазіалгоритми

Алгоритмом називається чітка однозначна зрозуміла виконавцеві послідовність інструкцій, виконання яких обов'язково приводить до гарантованого результату за скінченний час. На відміну від цього, інструкції квазіалгоритму можуть бути не зовсім чіткими, і результат виконання квазіалгоритмічної процедури не обов'язково є гарантованим.

Можна виділити як мінімум чотири основні *джерела квазіалгоритмічності*:

1. **Дія випадкових чинників**, що не залежать від виконавця. Строго кажучи, з огляду на цей фактор квазіалгоритмом слід вважати будь-який, навіть найбільш формалізований, алгоритм. Алгоритм розрахований на роботу при певних умовах; якщо ці умови зміняться, алгоритм може не призвести до потрібного результату. *Якщо ж не вдається чітко окреслити межі застосування алгоритму або забезпечити виконання необхідних умов для його роботи, ми маємо справу зі справжнім квазіалгоритмом*.

2. **Недостатнє врахування автором алгоритмічної процедури особливостей виконавця.** Це призводить до того, що виконавець неправильно розуміє, що від

нього вимагається. Процедура може бути в принципі алгоритмічною, за нормальних умов призводити до гарантованого результату, але цей результат може бути не передбачений автором і тому бути для нього несподіванкою.

3. **Нечіткість формулювань**, що фігурують в описі процедури. Розглянемо будь-який кулінарний рецепт, наприклад: “розтерти тісто, змішати з дрібно нарізаними яблуками, додати солі і перцю за смаком і смажити до появи рум’яної скоринки”. Це є типовим прикладом нечіткості формулювань, що призводить до невизначеностей. До якої межі слід розтирати тісто? Що означає “дрібно нарізані”? Що означає “за смаком” і т. д. Неінтелектуальна система, орієнтована на чисто алгоритмічне керування, просто не зрозуміє цього опису і тому не зможе його виконати. Інтелектуальна ж система, наприклад, людина, повинна спробувати поповнити і уточнити цей опис на основі наявних знань і досвіду. Сам по собі факт виконання завдання буде свідчити про те, що квазіалгоритмічний первинний опис процедури був зведений до деякого внутрішнього алгоритму, але навряд чи виконавець зможе чітко пояснити і розписати цей алгоритм. Крім того, ці внутрішні алгоритми для кожного виконавця будуть різними.

4**. "Свобода волі",** яку можуть мати високоорганізовані, по-справжньому інтелектуальні системи. Ці системи можуть мати свої цілі, і, якщо дана процедура суперечить цим цілям, вони можуть відмовитися її виконувати або виконати не так, як це від них вимагається. “Свобода волі” полягає у тому, що інтелектуальна система самостійно приймає рішення про свою поведінку і в залежності від цих рішень може виконувати зовнішні розпорядження, не виконувати їх, або ж виконувати так, як вона вважає за потрібне.

4.3. Характеристика інтелектуальних систем з точки зору кібернетики

4.3.1. Означення інтелектуальної системи

Інтелектуальна система може припускати зовнішнє керування, але для неї характерною є **самокерованість**. Система має **певну мету і прагне так планувати свої дії, щоб досягати цієї мети**. Як вхідні стимули системи можна розглядати поточну ситуацію, що сприймається і аналізується системою. Результатом реакції системи стає зміна зовнішньої ситуації, і поведінка системи коригується в залежності від того, бажаною чи небажаною є ця зміна.

Людина має певну суму **знань про світ**, яка дозволяє їй орієнтуватися в життєвих ситуаціях та приймати правильні рішення. Крім того, людина вміє певним чином **використовувати ці знання**. Ці самі риси мають мати системи штучного інтелекту.

Також можна стверджувати, що **здатність до поповнення первинних знань, є** *однією із ключових рис інтелектуальних систем*. Ця властивість інтелектуальних систем називається **здатністю до навчання.** Розрізняють *зовнішнє навчання* та *самонавчання*.

Підсумувавши усі сказане, можна стверджувати, що **інтелектуальною системою** *називається самокерована кібернетична система, яка має певну суму знань про світ і здатна на основі безпосереднього сприйняття і подальшого аналізу поточної ситуації до планування дій, спрямованих на досягнення мети, а також до навчання*.

4.3.2. Типова схема функціонування інтелектуальної системи

Функціонування інтелектуальної системи можна описати як постійне прийняття рішень на основі аналізу поточних ситуацій для досягнення певної мети. Схема функціонування інтелектуальної системи наведена на рис. 4.1.

Виокремлюють наступні *етапи функціонування інтелектуальних систем*:

1. **Безпосереднє сприйняття зовнішньої ситуації.**
   Результатом є первинний опис ситуації.

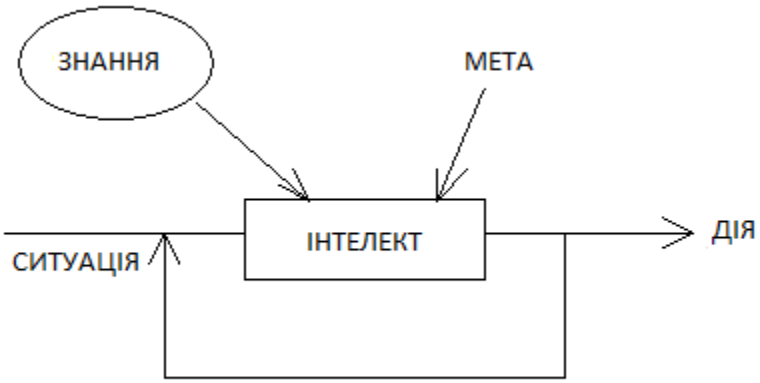


Рис. 4.1. Схема функціонування інтелектуальної системи.

2. **Зіставлення первинного опису зі знаннями системи і поповнення цього опису.**
   Результатом є формування вторинного опису ситуації в термінах знань системи. Цей процес можна розглядати як процес *розуміння ситуації* або як процес перекладу первинного опису на внутрішню мову системи. При цьому можуть змінюватися внутрішній стан системи та її знання.

   Вторинний опис може бути не єдиним, і система має змогу вибирати між різними вторинними описами. Крім того, система в процесі роботи може переходити від одного вторинного опису до іншого. Якщо ми можемо формально задати форми внутрішнього представлення описів ситуацій та операції над ними, то сподіватимемося на певний автоматизований аналіз цих описів.
3. **Планування цілеспрямованих дій та прийняття рішень, аналіз можливих дій та їх наслідків і вибір тих дій, що найкраще узгоджуються з метою системи.**
   Таке рішення зазвичай формулюється певною внутрішньою мовою (свідомо або підсвідомо).
4. **Зворотна інтерпретація прийнятого рішення, тобто формування робочого алгоритму для реагування системи.**
5. **Реалізація реакції системи; наслідками є зміни зовнішньої ситуації і внутрішнього стану системи і т. д.**

Проте не слід вважати, що зазначені етапи є повністю відокремленими у тому розумінні, що наступний етап починається тільки після того, як повністю закінчиться попередній. Навпаки, для функціонування інтелектуальної системи характерним є взаємне проникнення цих етапів. Наприклад, ті чи інші рішення можуть прийматися вже на етапі безпосереднього сприйняття ситуації.

5. ПОДАННЯ ЗНАНЬ В ІНТЕЛЕКТУАЛЬНИХ СИСТЕМАХ

5.1. Підходи до подання знань

У філософії знання визначаються як "відображення об'єктивних властивостей та звя'зків світу".

Знання *є інформаційною основою інтелектуальних систем, оскільки саме вони завжди зіставляють зовнішню ситуацію зі своїми знаннями і керуються ними при прийнятті рішень. Не менш важливим є те, що знання — це систематизована інформація, яка може певним чином поповнюватися і на основі якої можна отримувати нову інформацію, тобто нові знання.*

Існує багато підходів до визначення поняття "знання". На сучасному етапі домінуючою парадигмою, що лежить в основі найвідоміших моделей подання знань у системах штучного інтелекту, можна вважати парадигму (певну сукупність ключових принципів), характерну для **символьного підходу**. Цю парадигму можна охарактеризувати як **вербально-дедуктивну**, або **словесно-логічну**, через певні чинники:

- будь-яка інформаційна одиниця задається вербально, тобто у формі, наближеній до словесної, у вигляді набору явно сформульованих тверджень або фактів;
- основним механізмом отримання нової інформації на базі існуючої є **дедукція**, тобто висновок від загального до часткового. Такий дедуктивний підхід є бездоганним з логічного погляду. В його основі лежать транзитивність імплікації (якщо з a випливає b, з b випливає c, то з a випливає c) і дія квантора загальності (якщо деяка властивість P виконується для будь-якого елемента множини M, а $x \in M$, то P виконується для x.

Але вербально дедуктивне задання знань не є повним, оскільки:

- дедуктивний висновок не виступає єдино можливим. Мислення людини багато в чому є рефлекторним, інтуїтивним. Воно, як правило, спирається на підсвідомі процеси. Людина часто робить висновки за аналогією, асоціацією. Ці висновки не завжди вірні, але вони істотно доповнюють процеси дедуктивного мислення. Без підсвідомого мислення стає неможливим (будь-яке відкриття — як правило, підсвідоме породження гіпотези, яка потім перевіряється дедуктивним або експериментальним шляхом);
- далеко не всі знання є вербальними. Так, жодне твердження не зберігається в пам'яті людини явно. Відомо, що в основі діяльності мозку людини лежить передача сигналів між нервовими клітинами. Часто людина не може сформулювати свої знання. Наприклад, будь-хто знає, що таке "стіл". Але якщо попросити людину дати визначення цього поняття, у неї можуть виникнути проблеми. Таким чином, поняттями можна оперувати і не знаючи чіткого їх визначення. Типовими є слова: "Я не можу пояснити чому, але мені здається".

Тому необхідно розвивати інші моделі знань, окрім вербально-дедуктивних Наприклад, у рамках **конекціоністського підходу** можна розглядати моделі на основі однорідного поля знань. **Однорідним полем знань** називається сукупність простих однорідних елементів, які обмінюються між собою інформацією: нові знання народжуються на основі певних процедур, визначених над полем знань.

5.2. Вербально-дедуктивне визначення знань

У рамках вербально-дедуктивної парадигми можна навести таке визначення знань.

Знаннями *інтелектуальної системи називається трійка <F, R, P>, де F — сукупність явних фактів, які зберігаються в пам'яті системи в явному вигляді, R — сукупність правил виведення, які дозволяють на основі відомих знань набувати нових знань, P — сукупність процедур, які визначають, яким чином слід застосовувати правила виведення.*

Базою знань *(БЗ) інтелектуальної системи називатимемо сукупність усіх знань, що зберігаються в пам'яті системи.*

У базах знань прийнято розрізняти *екстенсіональну* та *інтенсіональну* частини.

Екстенсіональною частиною бази знань називається сукупність усіх явних фактів*, інтенсіональною* частиною — сукупність усіх правил виведення та процедур, за допомогою яких з існуючих фактів можна виводити нові твердження.

Приклад. Розглянемо таку базу знань.

Ф1. Заєць їсть траву.
Ф2. Вовк їсть м'ясо.
ФЗ. Заєць є м'ясом.
ПІ. Якщо *A* є м'ясом, а *B* їсть м'ясо, то *B* їсть *A*.
П2. Якщо *A* їсть м'ясо, то *A* є хижаком.
ПЗ. Якщо *A* їсть *B*, то *B* є жертвою.
П4. Якщо хижак вимирає, жертва швидко розмножується.
П5. Якщо жертва вимирає, хижак вимирає.

Даний приклад є фрагментом неформалізованого опису моделі Лотки-Вольтерра, добре відомої в екології. Літерами "**Ф**" позначені явні факти, які відображають елементарні знання. Їх безпосередній аналіз дозволяє експертній системі відповідати на найпростіші запитання, такі, як "Вовк їсть м'ясо?". Для позитивної відповіді досить просто знайти відповідний факт в екстенсіональній частині бази знань.

Принципово іншим є запитання типу "Вовк є хижаком?". Відповідь на нього на основі простого пошуку знайти неможливо: такого явного факту в екстенсіональній частині бази знань немає. Тому для відповіді на це запитання необхідно застосовувати правила виведення, які дозволяють на основі існуючих явних фактів набувати нових знань, тобто нових фактів. Такі правила позначені у нашому прикладі літерою "**П".** Наприклад, для відповіді на поставлене запитання досить знання факту **Ф2** і правила **П2.**

Безумовно, до бази знань можна було б відразу включити твердження "Вовк є хижаком". Але легко побачити: якщо, крім вовків, розглядати інших хижаків (ведмедів, лисиць та ін.), то безпосередньо включати до бази знань явні факти, що вони є хижаками, недоцільно. Все, що може бути виведене логічним шляхом, як правило, не варто оголошувати фактами. Надалі називатимемо явні факти просто фактами, якщо це не викликає непорозумінь.

Нарешті, **процедури** визначають, яким саме чином слід застосовувати правила. Часто ці процедури є складовою мови, якою програмується експертна система (це стосується насамперед спеціалізованих мов ШІ, таких як Лісп, Пролог, Пленер).

5.3. Експертні системи

База знань, наведена в п. 3.2, може лягти в основу *експертної системи*, тобто інформаційної системи, яка здійснює дедуктивне виведення на основі наявних знань. Існують різні визначення експертних систем, основна суть яких зводиться до такого.

Експертні системи *— це інтелектуальні програмні засоби, здатні у діалогу з людиною одержувати, накопичувати та коригувати знання із заданої предметної галузі, виводити нові знання, розв'язувати на основі цих знань практичні задачі та пояснювати хід їх розв'язку.*

Експертні системи акумулюють знання експертів — провідних спеціалістів у даній предметній галузі. В основі роботи експертних систем лежить дедуктивне виведення нових тверджень з існуючих. Типове застосування експертних систем — консультації для фахівців середньої кваліфікації і неспеціалістів у тій галузі, для якої вона розроблена.

Створити універсальну експертну систему неможливо. По-перше, це пов'язано з "прокляттям розмірності". По-друге, експертна система повинна акумулювати знання людей-експертів, а "універсальних" експертів не існує і не може існувати. По-третє, жодне логічне виведення не може замінити інтуїцію та досвід експерта.

Можна лише експертні системи, які належать до конкретної предметної галузі. Остання передбачає лімітований набір явищ і понять певної сфери людської діяльності та обмежене коло задач, які вирішуються у цій галузі. Існує чимало експертних систем у таких сферах як медична і технічна діагностика, пошук корисних копалин, юриспруденція, аналіз інвестицій і комерційних ризиків і т. п. У створенні експертних систем повинні брати участь фахівці як мінімум двох категорій:

- **експерт,** що є висококваліфікованим фахівцем у даній предметній області, знання якого потрібно передати експертній системі;
- **інженер знань,** завдання якого — формалізувати знання експерта і *привести* їх до вигляду, придатного для занесення до бази знань.

Серйозна проблема у даному випадку пов'язана з тим, що знання експертів важко формалізувати. Експерт часто не може сформулювати свої знання у явному вигляді, робить правильні висновки, але *не може пояснити як саме він їх робить*. Якщо ж експерт і формулює певні правила виведення, вони далеко не завжди відзначаються точністю та адекватністю. З цього випливає ряд інших труднощів, які так чи інакше виявляють себе на етапі формалізації знань або застосування готових експертних систем.

5.4. Дані та знання

Відомо, що сучасний етап розвитку інформатики характеризується еволюцією моделей даних у напрямі переходу від традиційних (реляційна, ієрархічна, мережева) до моделей знань. Знання, як будь-яку інформацію, можна вважати даними. Але *знання є високоорганізованими даними, для яких характерна певна внутрішня структура та розвинуті зв'язки між різними інформаційними одиницями.* Іншим принциповим аспектом є те, що інформаційні системи, які ґрунтуються на знаннях, повинні мати можливість отримувати нові знання на основі існуючих. У цьому ми вже пересвідчилися на прикладах дедуктивного виведення нових знань з явних фактів.

Екстенсіональна частина БЗ складається з фактів, які запам'ятовуються явно, інтенсіональну — з правил, які дозволяють отримувати нові факти. Наявність розвинутої інтенсіональної частини є однією з основних рис, які відрізняють знання від традиційних даних. **Інтенсіональним відношенням,** яке описує базу даних, часто називають правило або сукупність правил, яким підпорядковується кожний запис у базі даних.

Приклад. Розглянемо базу даних (БД) деканату, що містить дані про те, які курси прослухав кожний студент. Відомо, що жоден студент не має права слухати курс "Бази даних", якщо він не прослухав курсу "Основи програмування". Нехай у базі даних зберігається інформація про Іванова, Петрова та Сидорова, які прослухали обидва курси.

Екстенсіональна частина:

Прізвище	Дисципліна
Іванов	Бази даних
Петров	Бази даних
Сидоров	Бази даних

Інтенсіональна БД (або її вже можна назвати БЗ) може мати такий вигляд:

Правило: Якщо Прослухав(*X***,** Бази даних), то Прослухав**(*X*,** Основи програмування).

Наявність такого правила дозволяє скоротити базу знань: твердження, істинність яких можна встановити за допомогою правил, не обов'язково запам'ятовувати в явному вигляді.

Пошук в інтенсіональній БЗ складається з двох етапів:

- пошук потрібного факту в екстенсіональній частині;
- дедуктивне виведення факту на основі правил інтенсіональної частини.

Реалізовувати інтенсіональні правила можна навіть у рамках реляційної моделі даних шляхом програмування відповідних запитів.

Знання можуть бути **неповними.** Це означає, що для доведення або спростування певного твердження може не вистачати інформації. У багатьох системах логічного виведення прийнято **постулат замкненості світу:** на запит про істинність деякого твердження система відповідає "так" тоді і тільки тоді, коли його можна довести; якщо ж довести неможливо, система відповідає "ні". Водночас *"неможливо довести через нестачу інформації"* і *"доведено, що ні"* — це зовсім не те саме. З огляду на це бажано, щоб експертна система запитувала у користувача про факти, яких не вистачає.

Знання можуть бути **недостовірними.** Наприклад, на виготовлення продукції можуть впливати випадкові чинники (об'єктивна невизначеність) або експерт може бути не зовсім упевненим у деякому факті чи правилі (суб'єктивна невизначеність).

Ненадійність знань і недостовірність наявних фактів обов'язково повинні враховуватися в процесі логічних побудов. Звичайно, можна було б просто відкидати факти та правила виведення, які викликають сумнів, але довелося би відмовитися від цінної інформації. Крім того, в експертних системах часто доводиться мати справу з неточно визначеними поняттями, такими, як *"великий", "маленький"* тощо.

5.5. Зв’язки між інформаційними одиницями

У базі знань можуть зберігатися відомості про різні об’єкти, поняття, процеси і т. п., тобто відповідні описи, які називаються **інформаційними одиницями**.

Опис; який утворює ***інформаційну одиницю,*** *розглядається як деяка абстракція реальної сутності, що існує в дійсному світі.*

Це означає що дана абстракція описує певні риси реальної сутності. Бажано, щоб абстракція описувала найважливіші з них.

У реальному світі все тісно взаємопов’язане. Відповідні зв’язки повинні знайти адекватне відображення в БЗ. Існує велика група зв’язків, за допомогою яких можна описувати просторові відношення ("знизу", "зверху", "поруч", "між" і т. п.); часові відношення ("раніше", "пізніше", "під час", "одночасно" і т. п.), причинно-наслідкові відношення тощо. Ці відношення є спільними для всіх предметних областей. Логічні системи, що ґрунтуються на цих відношеннях, називаються **псевдофізичними логіками.** Використання псевдофізичних логік (гак само, як і більш загальних предикатів, таких, як узагальнення та агрегація) дає можливість інтелектуальній системі поповнювати первинні описи.

Так, якщо інтелектуальна система сприймає текст: "У 1985 році почалася перебудова. У 1991 році була проголошена незалежність України", врахування часових властивостей, що визначається часовою логікою, дає змогу відповідати на запитання типу: "Що було раніше: початок перебудови и чи проголошення незалежності України?"

Основними зв’язками між інформаційними одиницями є **узагальнення** та **агрегація.**

Агрегація дозволяє задавати будову об’єкта та його складових. Наприклад, агрегація дає змогу визначити в базі знань аудиторію як об’єкт, що має вікна, двері і т. п. Вікна, в свою чергу, теж можна розглядати як агрегований об’єкт: вони мають ручки, рами і т. п. Інакше кажучи, відношення *"Має",* яке описує агрегацію, означає, що певний об’єкт містить у своєму складі деякий інший об’єкт. За цим відношенням закріпилася спеціальна назва **Has_Part.** По суті, це відношення "ціле — частина".

Таким саме чином ввести і зворотне відношення (*"Є частиною",* або **Is_Part**).

Узагальнення (відношення *"Є"*) задає ієрархію класів **(екземпляр — клас — надклас).** Так, можна говорити, що об’єкт Вася Петров належить до класу Студент. Відповідно інформаційна одиниця, яка описує Васю Петрова, може бути описана на основі більш загальної інформаційної одиниці Студент (точніше, вона задається як екземпляр інформаційної одиниці Студент). Клас Студент, у свою чергу, є *підкласом* типу Людина і т. п. Надклас об’єднує атрибути, або ознаки, спільні для його підкласів; підкласи можуть успадковувати ті чи інші властивості надкласів.

Із відношенням *"Є"* пов’язаний один з основних відомих механізмів логічного виведення — механізм **виведення за успадкуванням** (інша назва — виведення **за наслідуванням).** Суть цього механізму можна сформулювати так: якщо деяка умова виконується для всього класу, то вона виконується і для кожного представника цього класу, а також для всіх підкласів цього класу (якщо інше не задано явним чином). Інакше кажучи, екземпляри успадковують властивості класів, підкласи успадковують властивості надкласів.

В усіх системах керування базами знань повинна бути забезпечена належна підтримка успадкування. Розглянемо *приклад* [8]. У базі знань міститься така інформація:

Усі птахи літають.
Ластівка є птахом.
Юкко є ластівкою.

Маємо загальний клас *"Птахи",* його підклас *"Ластівки"* та об'єкт *"Юкко",* який є екземпляром класу *"Ластівка".* На підставі цих знань будь-яка система, що підтримує успадкування, повинна зробити висновки, що *Юкко є птахом; всі ластівки літають; Юкко теж літає.*

Насправді, замість єдиного відношення *"Є"* необхідно розглядати цілу систему споріднених, але не ідентичних відношень. Інакше легко припуститися логічних помилок, подібних до таких:

Юкко є ластівкою.
Ластівка є видом, який вивчається натуралістами.
Отже, *Юкко є видом, який вивчається натуралістами*.

Слід розглянути такі відношення:

- **"клас — підклас",** яке є відношенням часткового порядку (звичайно строгого). Для цього відношення виконується властивість транзитивності;
- **"екземпляр — клас".** Екземпляри успадковують властивості своїх класів, але саме відношення "екземпляр — клас" не є транзитивним.

"Ластівка" є екземпляром класу "Види, які вивчаються натуралістами", а "Юкко" є екземпляром класу "Ластівка", але "Юкко" у жодному разі не є екземпляром класу *"Види, які* "Види, які вивчаються натуралістами" ("Юкко" взагалі не є видом).

Також необхідно чітко розрізняти поняття **"клас"** і **"множина", "належність до класу"** та **"належність до множини".** Клас об'єднує однотипні елементи, множина — необов'язково. Відношення "елемент — множина" також не є транзитивним. Складні системи можуть бути описані у вигляді певної канонічної форми, яка являє собою композицію двох ієрархій — **ієрархії класів** та **ієрархії об'єктів.**

Ієрархія класів задається відношенням *узагальнення,* **ієрархія об'єктів** — *відношенням агрегації.*

Дійсно, з одного боку, такі об'єкти, як "ручки", "рами"та інші, утворюють новий об'єкт — "вікно". "Вікна", "двері" тощо — це об'єкти, які утворюють ще один новий об'єкт: "аудиторію". Таким чином, можна розглядати певну ієрархію об'єктів. З іншого ж боку, всі ці об'єкти є екземплярами своїх класів. Так, конкретне вікно належить до класу "Вікна", клас "Вікна" є підкласом класу "Дерев'яні вироби" і т. п.

Відношення "клас — підклас" є характерним для ієрархії класів, відношення "елемент — множина" та "підмножина — множина" — до ієрархії об'єктів, а відношення "екземпляр — клас" об'єднує обидві ієрархії.

5.6. Проблема винятків

З успадкуванням пов'язана дуже серйозна проблема — **проблема винятків.** Вона полягає в тому, що деякі підкласи можуть не успадковувати ті чи інші властивості надкласів. Інакше кажучи, характерні риси класу успадковуються всіма його підкласами, **крім** деяких.

Нехай відомо, що *літають всі птахи, крім пінгвінів* (існують деякі інші види птахів, які не літають, але для наших цілей це не має суттєвого значення). Якби це

твердження відразу потрапило до БЗ саме в такому вигляді, особливих проблем не виникало б (хоча і в цьому випадку слід було б передбачити належну обробку винятків).

Але, як було зазначено раніше, експерт не завжди може сформулювати свої знання в явному вигляді. Зокрема, він може не знати або не пам'ятати всіх винятків. Тому він може спочатку включити до БЗ твердження про те, що *всі птахи літають,* а потім пригадати, що *пінгвіни не літають,* і додати це до БЗ. У результаті ми могли б отримати БЗ, подібну до такої:

Усі птахи літають.
Ластівка є птахом.
Юкко є ластівкою.
Пінгвін є птахом.
Пінгвіни не літають.
Бакс є пінгвіном.

Якби три останні твердження **не були** включені до бази знань, системі просто дійшла б хибного висновку, що *Бакс літає.* Але включення даних відомостей до бази знань ще більше ускладнює ситуацію. **Система знань стає суперечливою: з** одного боку, система повинна дійти висновку, **що** Бакс літає, а з іншого — що Бакс не літає. У даному разі кажуть про **втрату монотонності** дедуктивної системи [8].

Система дедуктивного виведення *називається* ***монотонною,*** *якщо для неї виконується така властивість: якщо з набору тверджень* $(q_1, \ldots, q_n)$ *випливає твердження V, то V випливає і з набору тверджень* $(q_1, \ldots, q_n, r)$.

Інакше кажучи, в монотонній системі додавання нових фактів і правил не впливає на істинність висновків, які могли бути отримані без них.

Для вирішення проблеми винятків часто використовують наступне правило: *у разі виникнення суперечностей підклас успадковує відповідну властивість лише від найближчого попередника, тобто найближчого до нього в ієрархії класів.*

5.6. Властивості та моделі знань

У [9] сформульовані такі особливості знань, які відрізняють їх від звичайних даних:

1) **внутрішня інтерпретованість:** кожна інформаційна одиниця повинна мати унікальне ім'я, за яким інформаційна система її знаходить, а також відповідає на запити, в яких це ім'я згадується;
2) **структурованість:** знання повинні мати гнучку структуру; одні інформаційні одиниці можуть включатися до складу інших (відношення типу "частина — ціле", "елемент — клас");
3) **зв'язність*:*** в інформаційній системі повинна бути передбачена можливість встановлення різних типів зв'язків між різними інформаційними одиницями (причинно-наслідкові, просторові та ін.);
4) **семантична метрика*:*** на множині інформаційних одиниць корисно задавати відношення, які характеризують ситуаційну близькість цих одиниць;
5) **активність:** виконання програм в інтелектуальній системі повинно ініціюватися поточним станом бази знань.

Часто виокремлюють інші властивості знань, наприклад **шкальованість,** яка означає, що формально неоднакові поняття насправді відображаються на одній і тій самій **шкалі понять,** різні точки якої відповідають інтенсивності того самого фактора. Так, температура може бути високою або низькою, і це породжує такі поняття, як "холодно", "тепло", "гаряче" тощо.

Моделлю знань *називається фіксована система формалізмів (понять і правил), відповідно до яких інтелектуальна система подає знання в своїй пам'яті та здійснює операції над ними.*

Моделі задання знань необхідні:

- для створення спеціальних мов описів знань і маніпулювання ними;
- для формалізації процедур зіставлення нових знань з уже існуючими;
- для формалізації механізмів логічного виведення.

Найвідомішими з моделей, які ґрунтуються на вербально-дедуктивній парадигмі, є чотири класи моделей:

- семантичні мережі;
- фреймові;
- логічні;
- продукційні.

Вербально-дедуктивні моделі задання знань мають багато спільних рис, Отже, можна вважати, що всі вони мають єдину концептуальну основ).

5.8. Неоднорідність знань. Області і рівні знань

Для успішного здійснення операцій зі знаннями необхідно відокремлювати в БЗ певні фрагменти, які називаються **областями знань.** Ці області знань повинні бути відносно незалежними між собою, що означає таке:

- зміни в одній області знань не повинні приводити до суттєвих змін у інших областях;
- вирішення складної задачі можна, як правило, звести до підзадач таким чином, що для вирішення кожної з цих підзадач достатньо знань з однієї області.

Такий розподіл є важливим для полегшення проектування і використання БЗ. Зокрема, різні області знань можуть проектуватися незалежно одна від одної.

Виходячи з постановки задачі можна по-різному розділяти знанні на області. Так, для експертних систем, які ведуть діалог з користувачем мовою, наближеною до природної, можна виокремити такі області знань [7]:

- **предметна область,** яка містить знання про конкретну предмет, галузь, в якій працює експертна система;
- **область мови,** яка містить знання про мову, якою ведеться діалог;
- **область системи,** яка містить знання експертної системи про власні можливості;
- **область користувача,** яка містить знання про користувача. Наявність їх дозволяє враховувати індивідуальні особливості кожного користувача. Наприклад, пояснення користувачеві можуть надавати залежно від рівня його підготовленості;
- **область діалогу,** яка містить знання про мету діалогу, а також про форми та методи його організації.

Знання можуть різнитися за **рівнем задання** і **рівнем детальності. За рівнями задання** розрізняють знання нульового рівня (конкретні і абстрактні) і знання вищих рівнів — знання про знання **(метазнання).**

У випадку класифікації знань за **рівнями детальності** до уваги береться ступінь деталізації знань. Кількість рівнів детальності залежить від специфіки задачі, обсягу знань і моделі їх задання. Якщо користувач запитує в експертної системи, як здійснити певну операцію. У разі отримання відповіді на найвищому рівні детальності система видає *загальний план.* Якщо цей загальний план не є достатнім, система може спуститися на нижчий рівень і розписати операції детальніше.

5.9. База знань як об'єднання простіших одиниць

БЗ є кон'юнкцією простих тверджень. Можна вважати, що кожне з них відповідає окремому реченню природної мови. Подібні твердження називають **концептуальними одиницями.** Останні можуть бути або фактами, або правилами виведення.

Якщо концептуальна одиниця являє собою факт, вона може бути описана певним **предикатом,** тобто логічною функцією, що залежить від заданої кількості змінних і може приймати одне з двох можливих значень 0 або 1 (false або true). Конкретні твердження утворюються з предикатів шляхом підстановки конкретних значень аргументів. При цьому для опису одного й того самого твердження можна використовуватися різні предикати.

Так, два твердження:

7 + 5 = 12,

8 + 9 = 15

відповідають одному предикату Плюс(*a, b, c*).

Перша підстановка конкретних значень замість змінних *a, b, c* породжує істинне твердження, друга — хибне. Формально ці твердження будуть мати запис: Плюс(7, 5, 12) та Плюс(8, 9, 15).

Можна також розглядати загальніший предикат: Операція (назва_операції, *a, b, c*), який можна використовувати для запису тверджень не тільки про додавання, але й про будь-яку іншу бінарну операцію: множення, ділення і т. п. Можливим є запис: Операція(Плюс, 7, 5, 12).

Який тип розглянутих предикатів слід обирати, залежить від проектувальника БЗ і специфіки конкретної задачі.

5.10. Бінарні предикати і тріада "об'єкт—атрибут—значення"

Концептуальна одиниця може бути достатньо складною. Тому концептуальну одиницю, що є фактом, часто зручно розглядати як кон'юнкцію елементарніших тверджень, а саме — **бінарних фактів,** кожний з яких описується **бінарним предикатом,** тобто предикатом, який залежить від двох змінних. Якщо формалізувати правила об'єднання бінарних предикатів у складніші одиниці, на основі бінарних предикатів можуть бути сконструйовані складні твердження та системи знань.

Розглянемо приклад. Маємо твердження *"Студент Іванов отримав п'ятірку на іспиті зі штучного інтелекту".*

Перепишемо цю фразу у вигляді кон'юнкції бінарних фактів:

Є(Іванов, Студент)
Є(ІспШтІнт, Іспит)
Ісп_ШІ(Іванов, 5)

Предикат Ісп_ШІ був введений для задання зв’язку між різними інформаційними одиницями.

У вигляді бінарного можна переписати й **унарний предикат**, тобто предикат, який залежить від однієї змінної. Наприклад, предикат Птах(*x*), який означає, що *x* є птахом, можна переписати як **Є(***x***,** Птах).

Якщо певний унарний предикат описує **властивість** об’єкта, наприклад Літає**(***X***),** її можна переписати **у** вигляді Літає(*x***,** так).

Бінарним предикатам і бінарним фактам безпосередньо відповідає **тріада "об’єкт — атрибут — значення".** Об’єкт у цій тріаді — це, як правило, назва деякої інформаційної одиниці, атрибут — назва певної ознаки, а значення — конкретне значення цієї ознаки для даного об’єкта. Тріада "об’єкт — атрибут — значення" є просто іншою формою замін бінарного факту. Так, можна переписати наш приклад у вигляді:

Іванов —Є — Студент
Ісп_ШІ — Є — Іспит
Іванов — Ісп_ШІ — 5.

Остання тріада цього прикладу задає зв’язок між різними об’єктами.

6. КОНЕКЦІОНІСТСЬКІ МОДЕЛІ ТА МЕТОДИ

6.1. Загальна характеристика конекціоністського підходу та його місце в теорії інтелектуальних систем

Конекціоністський підхід до побудови систем штучного інтелекту розвинувся на противагу символьному, що є характерним для сучасних моделей знань. В основі конекціоністського підходу лежить спроба безпосереднього моделювання розумової діяльності людського мозку. Відомо, що мозок людини складається з величезної кількості нервових клітин (нейронів), що взаємодіють між собою. Ці "обчислювальні елементи" мозку функціонують набагато повільніше, ніж обчислювальні елементи комп'ютерних систем. Але ефективність людського інтелекту досягається як за рахунок паралельної роботи нейронів, так і за рахунок того, що механізми їх взаємодії були вироблені шляхом тривалої еволюції.

Для багатьох цілей нейрон можна розглядати як елемент з певним критичним значенням. Це означає, що він або ж дає на виході деяку постійну величину, якщо сума його входів досягає певного значення, або ж залишається пасивним.

Мак-Каллок і Піттс довели, що будь-яку обчислювану функцію можна реалізувати за допомогою спеціально організованої мережі ідеальних нейронів, логічні властивості яких з високою достовірністю можна приписати реальному нейрону. Але ця мережа буде мати наступні вади. По-перше, проблема полягає в тому, чи можна знайти якийсь розумний принцип реорганізації мережі, який дозволяв би випадково об'єднаній, спочатку, групі ідеальних нейронів самоорганізовуватись в "обчислювальний пристрій", здатний вирішувати довільну задачу розпізнання. По-друге, потрібно використовувати велику кількість нейронів. Так, модель мурашки потребує використання близько 20000 нейронів, людини — 100 млрд. нейронів, що на практиці неможливо.

Нейрологічна теорія стала також основою системи розпізнавання, яка дістала назву ***перцептрон***. В цьому підході основна увага приділялась встановленню характеристик, приписаних фіксованій множині детекторів ознак. Альтернативний підхід розпізнавання зводиться до пошуку "*добрих"* ознак, на основі яких розпізнавання здійснюється найбільш чітко. Наприклад, перцептрон Розенблатта передавав повідомлення від "ока", яке реалізовувалось системою фотоелементів, в блоки електро-механічних комірок пам'яті, які оцінювали відносні величини електричних сигналів. Ці комірки з'єднувались між собою випадковим чином, створюючи мережу з прямими зв'язками. Зазначимо, що в ній були відсутні зворотні зв'язки між нейроподібними елементами. Перцептрон міг навчатись шляхом спроб і помилок, а також корекцією електричних імпульсів.

Мінські і Пейпертом було математично доведено, що перцептрони не в змозі виконувати багато приписуваних їм функцій, наприклад, розпізнавання частково затулених предметів. Після цього результату розвиток перцептронної теорії призупинився.

Один з сучасних напрямків створення розумних машин — це розробка *нейрокомп'ютерів*. *Нейрокомп'ютер* — це програмно-технічна система (спеціалізована ЕОМ), яка реалізує деяку формальну модель природної мережі нейронів.

В основу машин п'ятого покоління покладено ідею паралельної обробки інформації в нейроподібних системах. Не зважаючи на те, що електронний процесор працює в тисячі разів швидше, ніж його нейронний еквівалент у мозку, мережі нейронів

вирішують багато задач (особливо нечислових) в тисячі раз швидше, ніж електронний процесор.

Причини цього такі:

1) Характер взаємозв'язків між нейронами дозволяє розв'язувати багато задач на основі паралельної обробки;
2) У нейронній мережі пам'ять не локалізована в одному місці (як в послідовних машинах), а розподілена по всій структурі. В біологічних системах пам'ять реалізується підсиленням або послабленням зв'язків між нейронами, а не зберіганням двійкових символів;
3) Біологічні мережі реагують не на всі, а тільки на визначені зовнішні подразнення. Кожний нейрон виступає як елемент прийняття рішення і як елемент зберігання інформації. Перевага такої структури — "життєздатність" (вихід з ладу декількох нейронів не приводить до значної зміни даних, що зберігаються, або ж до руйнування всієї системи).
4) Можливість адресації за вмістом (асоціативної пам'яті) є ще однією важливою характеристикою систем з розподіленою пам'яттю (кожний елемент відшукується за його вмістом, а не зберігається в комірці пам'яті з визначеним номером).

В основу зв'язків в нейрокомп'ютерах покладено *принцип асоціацій*. *Асоціативні зв'язки* пронизують все мислення людини. Існує думка [1], що *процеси мислення є не що інше, як розповсюдження певного збудження, як деяка ланцюгова реакція*. Навіть найбільш примітивні процеси навчання принципово залежать від послідовності подій в часі. Це й закладено в природу нейронних систем. Тому їм притаманне реагування тільки на жорстко визначені зовнішні подразнення. Наприклад, домашні тварини "навчаються" ігнорувати повторні несуттєві зовнішні подразнення ("цокання" годинника), але посилюють сприйняття подразнень, які можуть мати серйозні наслідки (звук автомобільних гальм).

Багато дослідників вважає, що майбутнє належить комп'ютерам, які базуються на аналізі зв'язків, а не обробці символів [11]. Мінські вважав, що якщо комп'ютер повинен діяти подібно мозку, тоді й його конструкція повинна бути також подібна до мозку.

Моделі штучних нейронних мереж (ШНМ) і схеми з адресацією за вмістом мають і недоліки. Внаслідок нефіксованої організації вони можуть плутати різні об'єкти. Але це аналогічно звиканню, посиленню чуттєвості до асоціацій, які лежать в основі психічних особливостей людини. Іншими словами, будь-який комп'ютер, який претендує на "розумність" повинен мати такі особливості.

Штучні нейрони, що також називаються нейронними клітинами, вузлами, модулями, моделюють структуру й функції біологічних нейронів. Архітектура й особливості штучних нейронних мереж, утворених нейронами, залежать від конкретних завдань, які мають бути вирішені з їхньою допомогою [11].

6.2. Модель штучного нейрона

Структуру штучного нейрона, запропоновану у [11], зображено на рис. 6.1.

Вхідними сигналами штучного нейрона x_i $\left(i = \overline{1,N}\right)$ є вихідні сигнали інших нейронів, кожний з яких узятий зі своєю вагою w_i $(i = \overline{1,N})$, аналогічною до синаптичної сили.

Вхідний оператор $f_{\text{вх}}$ перетворює зважені входи й подає їх на оператор активації f_a. Вихідний сигнал нейрона у являє собою перетворений вихідним оператором $f_{\text{вих}}$

вихідний сигнал оператора активації. Таким чином, нелінійний оператор перетворення вектора вхідних сигналів x у вихідний сигнал у може бути записаний у такий спосіб:

$$y = f_{\text{вих}}(f_{\text{а}}\,(f_{\text{вх}}(\boldsymbol{x}, \boldsymbol{w}))) \tag{6.1}$$

Як вже зазначено, вихідний сигнал даного нейрона є вхідним для наступного.

Рис. 6.1. Структура штучного нейрона

Вхідний оператор (вхідна функція) нейрона задає вигляд використовуваного в нейроні перетворення зважених входів. Відмінність гальмуючих входів від збуджувальних відбивається у знаках відповідних ваг. Звичайно використовуються такі вхідні функції:

— сума зважених входів

$$f(\mathbf{x}, \mathbf{w}) = \sum_{i=1} w_i x_i;$$

— максимальне значення зважених входів

$$f(\mathbf{x}, \mathbf{w}) = \max_i(w_i x_i);$$

— мінімальне значення зважених входів

$$f(\mathbf{x}, \mathbf{w}) = \min_i(w_i x_i).$$

6.2.1. Функція активації

Функція активації $f_a(\cdot)$ описує правило переходу нейрона, що перебуває в момент часу k у стані $z(k)$, у новий стан $z(k+1)$ при надходженні вхідних сигналів x

$$z(k+1) = f_a(z(k), f_{\text{вх}}(\mathbf{x}, \mathbf{w})).$$

Надалі позначатимемо функцію активації без індексу *«а»*. Найбільш простими активаційними функціями є

— лінійна

$$f(z) = Kz,\ K = \text{const};$$

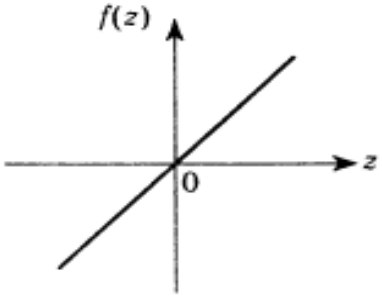


Рис. 6.2. Лінійна функція

— лінійна біполярна з насиченням

$$f(z) = \begin{cases} 1 \text{ при } z > \alpha_2; \\ Kz \text{ при } -\alpha_1 \leq z \leq \alpha_2; \\ -1 \text{ при } z < \alpha_1; \end{cases}$$

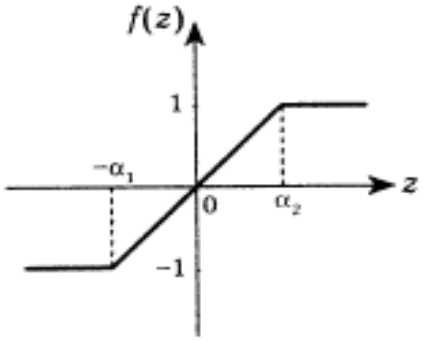


Рис. 6.3. Лінійна біполярна функція з насиченням

— лінійна уніполярна з насиченням

$$f(z) = \begin{cases} 1 \text{ при } z \geq \dfrac{1}{2\alpha}; \\ \alpha z + 0{,}5 \text{ при } |z| < \dfrac{1}{2\alpha}; \\ 0 \text{ при } z \leq -\dfrac{1}{2\alpha}. \end{cases}$$

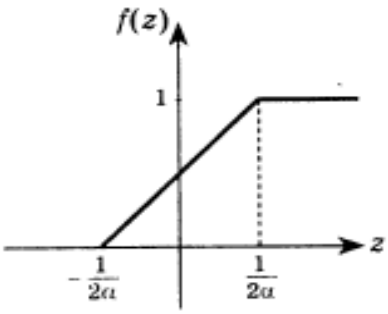


Рис. 6.4. Лінійна уніполярна функція з насиченням

Незважаючи на те, що лінійні функції є найбільш простими, їхнє застосування

обмежене, в основному, найпростішими ШНМ, які не мають у своєму складі прихованих шарів, у яких, крім того, існує лінійна залежність між вхідними й вихідними змінними. Такі мережі мають обмежені можливості. Двошарова лінійна мережа еквівалентна одношаровій з ваговою матрицею, що дорівнює добутку вагових матриць першого й другого шарів. Звідси випливає, що будь-яка багатошарова лінійна, мережа може бути замінена еквівалентною одношаровою. Хоча, використання лінійних активаційних функцій не є зайвим у багатошарових ШНМ, для розширення ж можливостей мережі застосовують нелінійні функції активації.

У роботі У. Мак-Каллока і У. Піттса у якості активаційної використовувалася *функція Хевісайда* — уніполярна *порогова* гранична функція вигляду

$$f(z) = \begin{cases} 1 \text{ при } z \geq \alpha; \\ 0 \text{ при } z < \alpha. \end{cases}$$

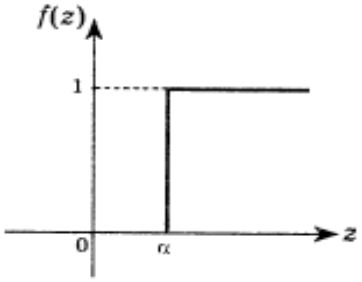


Рис. 6.5. Уніполярна порогова функція

Різновидом даної функції є біполярна порогова функція

$$f(z) = \begin{cases} 1 \text{ при } z \geq \alpha; \\ -1 \text{ при } z < \alpha. \end{cases}$$

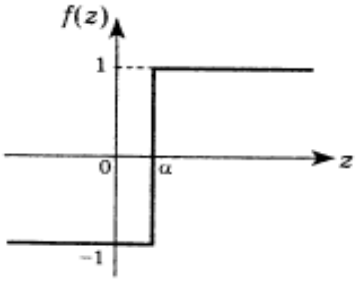


Рис. 6.6. Біполярна порогова функція

Ці функції активації застосовувалися в основному в класичних ШНМ. При побудові нових структур ШНМ найчастіше доводиться працювати як із самою активаційною функцією, так і з її першою похідною. У цих випадках необхідним є використання як активаційної монотонної диференційованої й обмеженої функції. Особливо важливу роль відіграють такі функції під час моделювання нелінійних залежностей між вхідними й вихідними змінними. Це так звані *логістичні*, або *сигмоїдальні* (*S*-подібні), функції.

Функція називається сигмоїдальною, якщо вона є монотонно зростаючою, диференційованою і задовольняє умові

$$\lim_{\lambda \to -\infty} f(\lambda) = k_1, \qquad \lim_{\lambda \to +\infty} f(\lambda) = k_2, \qquad k_1 < k_2.$$

До таких функцій належать:

— логістична (уніполярна)

$$f_{\log}(z) = \frac{1}{1+e^{-kz}}; \tag{6.2}$$

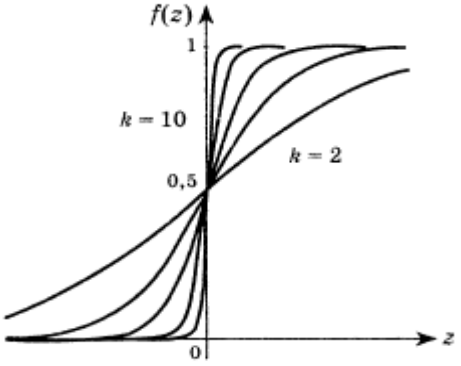


Рис. 6.6. Логістична функція

— гіперболічного тангенса (біполярна)

$$f_{\text{th}}(z) = \tanh(\alpha z) = \frac{e^{\alpha z} - e^{-\alpha z}}{e^{\alpha z} + e^{-\alpha z}};$$

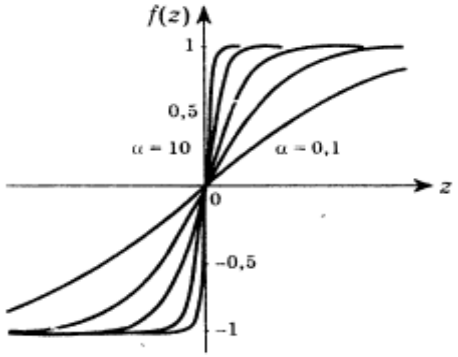


Рис. 6.8. Функція гіперболічного тангенса

Моделі штучних нейронів залежать від конкретних застосувань. Тому синтез моделі в кожному окремому випадку є нетривіальним завданням.

6.2.2. Формальна модель нейрона Маккаллока-Піттса

Формальний штучний нейрон (його називають також нейроном Мак-Каллока-Піттса) [11] може бути поданий як нелінійний перетворювач із ваговими коефіцієнтами w_{ji}, які також називаються синаптичними вагами або підсилювачами. Клітина тіла (сома) описується нелінійною обмежувальною або пороговою функцією $f(u_j)$. Найпростіша модель штучного нейрона додає N зважених входів і здійснює нелінійне перетворення (див. рис. 6.1)

$$y_j = f\left(\sum_{i=1}^{N} w_{ji}x_i + \theta_j\right), \tag{6.3}$$

де y_j — вихідний сигнал j-го нейрона; f — обмежувальна або порогова функція (активаційна); N — кількість входів; w_{ji} — синаптичні ваги; x_i — вхідні сигнали ($i = \overline{1, N}$); θ_j, ($\theta_j \in R$) — пороговий сигнал, що також називається зсувом.

Позначаючи $\theta_j = w_{j0}x_0$ (зазвичай $x_0 = 1$) формулу (6.3) можна переписати у вигляді

$$y_j = f\left(\sum_{i=0}^{N} w_{ji}\, x_i\right) = f\left(\mathbf{w}_j^T x\right),$$

де $x = (1, x_1, \dots, x_N)^T$; $w_j = (w_{j0}, w_{j1}, \dots, x_{jN})^T$ — відповідні вектори входів і ваг розмірності $(N+1) \times 1$.

7. АРХІТЕКТУРА ШТУЧНИХ НЕЙРОННИХ МЕРЕЖ

7.1. Поняття штучної нейромережі

З'єднані між собою нейрони утворюють *штучну нейронну мережу* (ШНМ). Таким чином, ШНМ — пара (N, C), де N — множина нейронів; C — множина зв'язків. Структура мережі задається у вигляді графа, у якому вершини є нейронами, а ребра являють собою зв'язки (з'єднання).

Кожен нейрон мережі має вхідні ланцюги, причому їхня кількість є довільною для кожного нейрона.

У загальному випадку ШНМ складається з декількох шарів, серед яких обов'язково є вхідний, що отримує зовнішні сигнали, вихідний, що відбиває реакцію нейронів на комбінації вхідних сигналів, і в багатошарових ШНМ — приховані шари (рис. 7.1). Така пошарова організація є аналогом шаруватих структур певних відділів мозку.

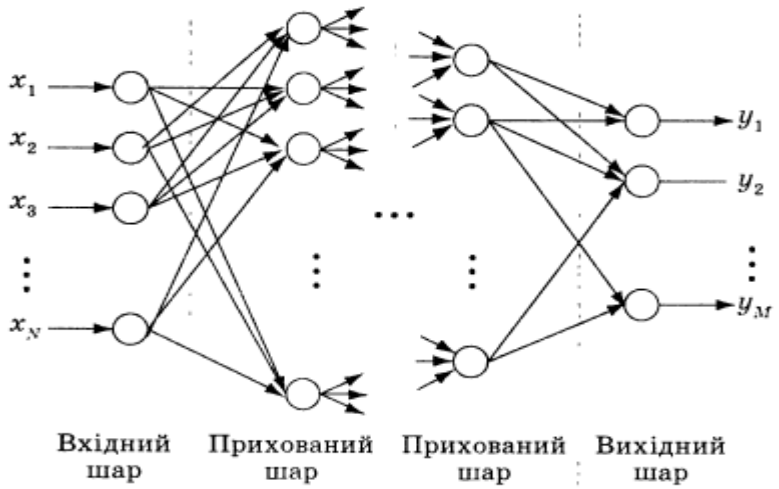


Рис. 7.1. Структура ШНМ

Зв'язки між нейронами задаються у вигляді векторів і матриць. Ваги зручно подавати елементами матриці $W = [w_{ij}]$ розмірності $N \times M$, де N — кількість входів; M — кількість нейронів. Елемент w_{ij} відбиває зв'язок між i-м й j-м нейронами. При цьому, якщо

$w_{ij} = 0$ — зв'язок між i-м й j-м нейронами відсутній;

$w_{ij} < 0$ — гальмуючий сигнал зв'язок;

$w_{ij} > 0$ — прискорювальний сигнал (збуджувальний) зв'язок.

Залежно від того, чи містять ШНМ зворотні зв'язки, чи ні, розрізнюють такі їхні топології:

- ШНМ без зворотних зв'язків (прямого поширення, Feed forward)
- ШНМ зі зворотними зв'язками (зворотного поширення, рекурентні, Feedback)
  — з прямими зворотними зв'язками (direct feedback);
  — з непрямими зворотними зв'язками (indirect feedback);
  — з латеральними зв'язками (lateral feedback);
  — повнозв'язні [1].

7.2. ШНМ прямого поширення

ШНМ прямого поширення припускає наявність декількох шарів зі зв'язками між нейронами різних шарів. У мережах першого порядку існують тільки зв'язки між двома сусідніми шарами, тобто між i-м й (i+1)-м шарами. У цьому випадку говорять, що зв'язки ШНМ пошарові. Приклад такої мережі зображено на рис. 6.10. Якщо в мережі цього типу кожен нейрон шару і пов'язаний з кожним нейроном (i+1)-го шару, мережа називається повнозв'язною прямого поширення.

У мережах другого порядку поряд із зв'язками між нейронами сусідніх i-го й (i+1)-го шарів присутні зв'язки між нейронами шарів i-го й (i+l)-го, де l>1. Такий зв'язок називається "shortcut". Приклад такої ШНМ наведено на рис.7.2.

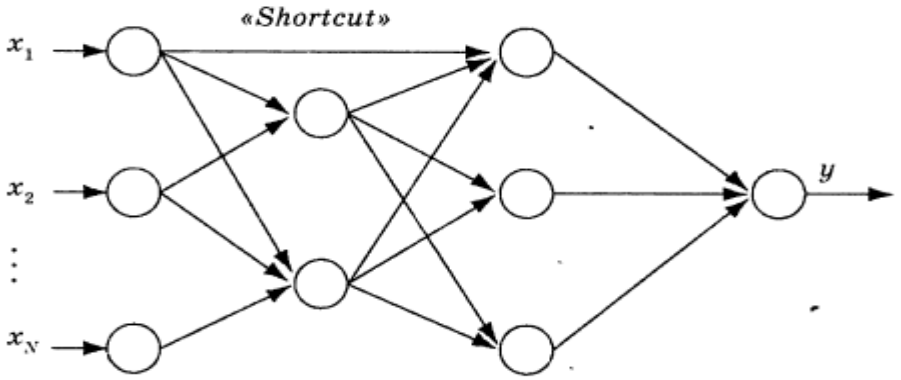


Рис. 7.2. ШНМ прямого поширення другого порядку

Зазначимо, що для мереж прямого поширення матриця зв'язків W є верхньою трикутною матрицею.

7.3. ШНМ зворотного поширення

Мережі цього типу припускають наявність зворотних зв'язків як між нейронами різних шарів, так і між нейронами одного шару. Використання мереж зі зворотними зв'язками необхідне у процесі вивчення складних динамічних об'єктів, наприклад об'єктів, що змінюють свій стан при надходженні нових вхідних сигналів. Такі ШНМ можуть мати властивості, подібні до короткочасної людської пам'яті.

У ШНМ із прямими зворотними зв'язками (рис. 7.3) на вхід нейрона деякого i-го шару подається його вихідний сигнал, тобто даний нейрон підсилює або послаблює сигнал, перетворений його активаційною функцією, завдяки чому досягається його граничний активаційний стан.

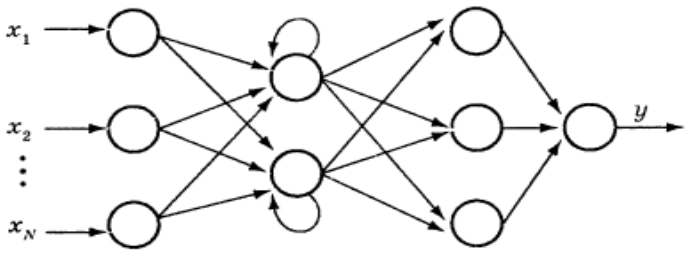


Рис. 7.3. ШНМ із прямими зворотними зв'язками

У ШНМ із непрямими зворотними зв’язками існують зв’язки нейрона *i*-го шару з нейронами (*i*–*k*)-го шару *k* > 0. При цьому одночасно можуть бути прямі зв’язки цього ж нейрона з нейроном (*i*+*l*)-го шару (*l* > 0). Введення таких зворотних звя’зків необхідно, щоб виділити певну особливо важливу для даної ШНМ область вхідних сигналів (рис. 7.4).

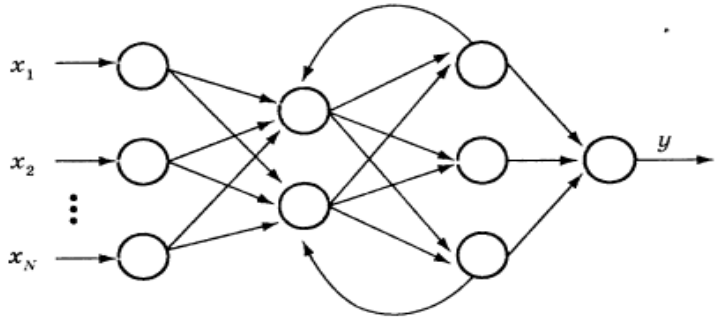


Рис. 7.4. ШНМ із непрямими зворотними зв’язками

ШНМ із латеральними зв’язками має зв’язки між нейронами одного шару (рис. 7.5). Такий тип зворотних зв’язків використовується у тому випадку, якщо тільки один нейрон з даної групи нейронів має бути активним. У цьому випадку на вхід кожного нейрона надходять гальмуючий (послаблюючий, інгібіторний) сигнал від інших нейронів і звичайно збуджувальний (посилюючий, ексгібіторний) сигнал власного зворотного зв’язку. Нейрон із найбільшою активністю (переможець) придушує інші нейрони. Тому цю топологію називають також топологією мережі «переможець отримує все» (WTA–Net).

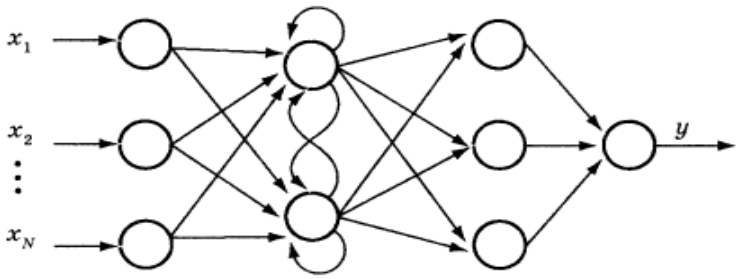


Рис. 7.5. ШНМ із латеральними зв’язками

7.3.1. Повнозв’язні ШНМ

Повнозв’язні ШНМ характеризуються наявністю зв’язків між усіма нейронами мережі (рис. 7.6). Цей вид топології відомий також як мережа Хопфілда.

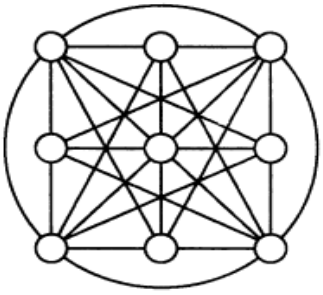


Рис. 7.6. Повнозв’язна ШНМ

8. НАВЧАННЯ ШНМ

8.1. Поняття про навчання ШНМ

Характерною властивістю ШНМ є її здатність до навчання, що полягає *у виробленні правильної реакції на подані їй різні вхідні сигнали*. Існують такі можливості навчання ШНМ:

—зміна конфігурації мережі шляхом утворення нових або виключення деяких існуючих зв'язків між нейронами;

—зміна елементів матриці зв'язку (ваг);

—зміна характеристик нейронів (виду й параметрів активаційної функції й т. д.).

Найбільшого поширення сьогодні отримав підхід, при якому структура мережі задається апріорно, а мережа навчається шляхом настроювання матриці зв'язків (вагових коефіцієнтів) *W*. Від того, наскільки вдало побудована ця матриця, залежить ефективність даної мережі. У цьому випадку навчання полягає у зміні за певною процедурою елементів матриці *W* при послідовному поданні мережі деяких векторів, що навчають.

У зв'язку з цим штучний нейрон може бути представлений у такий спосіб (рис. 8.1).

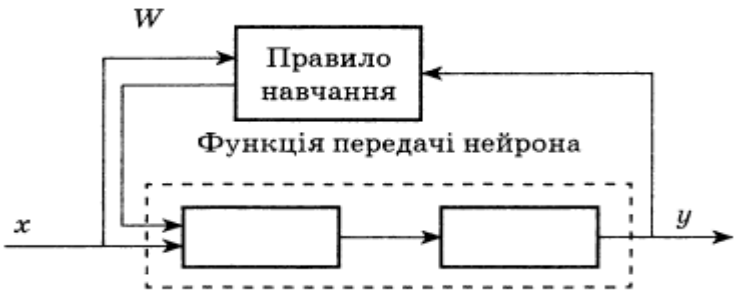


Рис. 8.1. Модель штучного нейрона

У процесі навчання ваги стають такими, що під час надходження вхідних сигналів мережа виробляє відповідні необхідні вихідні сигнали. Розрізняють навчання з учителем і без учителя. Перший тип навчання припускає, що є «учитель», що задає пари, які навчають — для кожного вхідного вектора, що навчає, необхідний вихід мережі. Для кожного вхідного вектора, що навчає, обчислюється вихід мережі, порівнюється з відповідно необхідним, визначається помилка виходу, на основі якої й коректуються ваги. Пари, що навчають, подаються мережі послідовно й ваги уточнюються доти, поки помилка за такими парами не досягне необхідного рівня.

Цей вид навчання неправдоподібний з біологічної точки зору. Дійсно, важко уявити зовнішнього «учителя» мозку, що порівнює реальні й необхідні реакції того, кого навчають, і коригує його поведінку за допомогою негативного зворотного зв'язку. Більш природним є навчання без учителя, коли мережі подаються тільки вектори вхідних сигналів, і мережа сама, використовуючи деякий алгоритм навчання, підстроювала б ваги так, щоб при поданні їй досить близьких вхідних векторів вихідні сигнали були б однаковими. У цьому випадку в процесі навчання виділяються статистичні властивості множини вхідних векторів, що навчають, і відбувається об'єднання близьких (подібних) векторів у класи. Подання мережі вектора з даного

класу викликає її певну реакцію, яка до навчання є непередбаченою. Тому в процесі навчання виходи мережі мають трансформуватися в деяку зрозумілу форму. Це не є серйозним обмеженням, оскільки зазвичай нескладно ідентифікувати зв'язок між вхідними векторами й відповідною реакцією мережі.

Існує ще один вид навчання — з підкріплюванням (reinforcement learning), при якому також передбачається наявність учителя, що не підказує, однак, мережі правильної відповіді. Учитель тільки повідомляє, правильно чи неправильно відпрацювала мережа поданий образ. На основі цього мережа корегує свої параметри, збільшуючи значення ваг зв'язків, що правильно реагують на вхідний сигнал, і зменшуючи значення інших ваг.

Сьогодні існує велика кількість алгоритмів навчання. Деякі з них розглядатимуться пізніше, тут же коротко зупинимося на найбільш відомих.

8.2. Правило навчання Гебба (корелятивне, співвідносне навчання)

Більшість сучасних алгоритмів навчання виросло із правила Гебба. Наприкінці 40-х років XX ст. років Д. О. Гебб теоретично встановив, що асоціативна пам'ять у біологічних системах викликається процесами, що змінюють зв'язки між нервовими клітинами. Відповідно до установленого їм правила, що називається «правилом Гебба», при одночасній активації (порушенні) двох нейронів синаптична сила (вага їхнього зв'язку) зростає. Таким чином, часто використовувані зв'язки в мережі підсилюються, що пояснює феномен звички й навчання повторенням.

У ШНМ зростання синаптичної сили еквівалентне збільшенню ваги зв'язку між нейронами *i* та *j* на величину

$$\Delta w_{ij} = \gamma x_i y_j,$$

де x_i — вихід *i*-го та вхід *j*-го нейронів; y_j — вихід *j*-го нейрона; γ — коефіцієнт, що впливає на швидкість навчання.

Правило Гебба використовується у зв'язках асоціативної пам'яті, а також у деяких інших, заснованих на навчанні без учителя (без підкріплювання). У мережах асоціативної пам'яті приймають **y** = **x**. У гетероасоціативних мережах **x** й **y** в загальному випадку різняться.

8.3. Дельта-правило

Це важливе правило навчання було запропоновано Б. Уідроу й М. Е. Гоффом і найбільше відповідає одношаровим ШНМ прямого поширення. Ідея його полягає в тому, що якщо під час навчання мережі можна встановити розбіжність між її бажаною й наявною реакціями, ця розбіжність може бути усунута або зменшена шляхом зміни певним чином вагових коефіцієнтів зв'язку. Для цього й використовується дельта-правило, відповідно до якого зміна ваги зв'язку між *i*-м й *j*-м нейронами визначається у такий спосіб:

$$\Delta w_{ij} = \gamma x_i (y_j^* - y_j),$$

де x_i — вихід попереднього *i*-го нейрона; y_j^*, y_j — бажана й реальна реакції *j*-го нейрона відповідно; γ — коефіцієнт, що впливає на швидкість навчання.

Якщо різниця $(y_j^* - y_j)$ мала, тобто реакція *j*-го нейрона незначною мірою відрізняється від бажаної, зміна ваги зв'язку між цими нейронами також буде

незначною.

8.4. Градієнтні методи навчання

Багато методів навчання засновано на мінімізації деякої цільової (вартісної, енергетичної й т. д.) функції *I*, що являє собою звичайно деяку опуклу функцію. Якщо використовувані функції активації $f(\cdot)$ диференційовані, зручно застосовувати градієнтні методи мінімізації. У цьому випадку корекція ваг зв'язку між *i*-м й *j*-м нейронами відбувається за правилом

$$\Delta w_{ij} = -\gamma \nabla_{\boldsymbol{w}} I(\boldsymbol{w}),$$

де γ — коефіцієнт, що впливає на швидкість навчання;

$$\nabla_{\boldsymbol{w}} I(\boldsymbol{w}) = \frac{\partial I(\boldsymbol{w})}{\partial \boldsymbol{w}_{ij}}.$$

Ці методи найчастіше використовуються при контрольованому навчанні, коли відома необхідна реакція нейронів $\boldsymbol{y}^*$.

Більшість градієнтних методів засновано на мінімізації квадратичного функціонала

$$I(\boldsymbol{w}) = 0{,}5e^2(k) = 0{,}5(y^*(k) - \boldsymbol{w}^T(k)\boldsymbol{x}(k))^2.$$

У цьому випадку

$$\nabla_{\boldsymbol{w}} I(\boldsymbol{w}) = -e(k)\boldsymbol{x}(k).$$

Таким чином приходимо до алгоритму методу найменших квадратів (МНК)

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \gamma e(k)\boldsymbol{x}(k).$$

Існують різні рекомендації з вибору γ. Так, у теорії стохастичної апроксимації, що вивчає особливості роботи алгоритмів такого типу за наявності завад вимірів, цей коефіцієнт вибирається змінним і таким, що задовольняє умовам Дворецького

$$\lim_{k\to\infty} \gamma(k) = 0, \sum_{k=1}^{\infty} \gamma(k) = \infty, \sum_{k=1}^{\infty} \gamma^2(k) < \infty,$$

зміст яких полягає в тому, що для забезпечення збіжності послідовності у деяку точку $\boldsymbol{w}^*$ довжина кроку $\gamma(k)$ має з одного боку спадати досить повільно (для забезпечення власне збіжності), а з іншого — досить швидко (з метою придушення завад). Таким умовам відповідає, наприклад, ряд $\gamma(k) = \gamma_0 k^{-\alpha}$, де γ_0 — деяка константа, $0{,}5 < \alpha \leq 1$. У теорії стохастичної апроксимації немає рекомендацій з вибору константи γ_0, крім її позитивності. Теорія оптимальної фільтрації, тісно пов'язана з теорією стохастичної апроксимації, дозволяє вибрати цю константу, що виявляється залежною від статистичних характеристик сигналів і завад.

8.5. Одношаровий перцептрон

8.5.1. Будова перцептрона

Значний інтерес до перцептронів викликаний роботою Ф. Розенблатта, у якій він

досліджував нейромережеву модель сітківки (RETINA) — фотоперцептрон. Згодом такий підхід широко використовувався для моделювання обробки оптичних сигналів. Фотоперцептрон зображений на рис. 8.2 і складається, відповідно до концепції Розенблатта, із трьох шарів, що послідовно здійснюють попередню обробку (розбивання) образу, оцінку його характеристик і розпізнавання:
— сітківка (RETINA);
— асоціативний шар;
— вихідний шар.

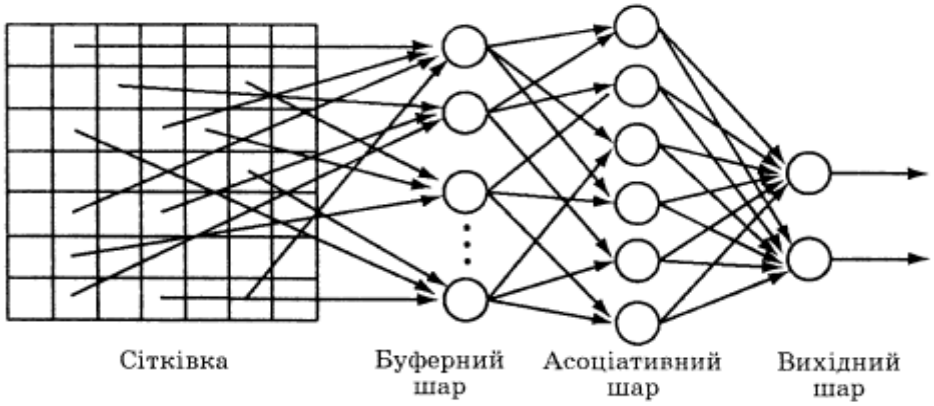


Рис. 8.2. Перцептрон Розенблатта

Попередня обробка образу не залежить від його виду. Однак результат цієї обробки має забезпечити можливість розпізнавання образів на основі аналізу їхніх характеристик. Нарешті, вихідний шар (класифікатор) аналізує характеристики знову пропонованого образу й установлює його відповідність одному з раніше поданих.

Сигнали першого шару, сітківки, подані у двійковій формі, надходять на асоціативний шар, причому в загальному випадку не всі нейрони першого шару пов’язані з усіма нейронами другого шару. При встановленні цих зв’язків виникає можливість структуризації вхідних даних, тобто виділення й об’єднання в так звані рецептивні поля найбільш важливих ознак (областей).

У зв’язку з цим під рецептивним полем розуміють множину всіх нейронів вхідного шару, пов’язаних з одним нейроном асоціативного шару. Зв’язки між нейронами асоціативного й вихідного шарів варіабельні й можуть модифікуватися шляхом зміни вагових коефіцієнтів.

Нейрони асоціативного шару мають лінійні активаційні функції, тому під час надходження із сітківки вхідних сигналів вони посилають імпульси на вихідний шар, де й відбувається додавання зважених імпульсів. У вихідному шарі використовуються уні- або біполярна активаційні функції. Якщо сума зважених імпульсів перевищує деяке задане порогове значення, виробляється одиничний вихідний сигнал, якщо не перевищує — нульовий (для уніполярної) або –1 (для біполярної функції активації). У зв’язку з цим перцептрон може розглядатися як двошарова ШНМ прямого поширення.

8.5.2. Навчання перцептрона

Існують різні шляхи реалізації процесу навчання перцептрона, однак у їхній основі лежить таке правило: вагові коефіцієнти перцептрона змінюються тільки тоді,

коли виникає розбіжність між його фактичною й бажаною реакціями.

Схему навчання перцептрона наведено на рис. 8.3.

Передбачається, що активаційна функція є пороговою. Процес навчання полягає в послідовному поданні множини пар, що навчають $(\mathbf{x}_p, y_p^*)$, $p = \overline{1,P}$, де $\mathbf{x}_p$ — N-вимірний вхідний вектор, y_p^* — бажаний вихідний сигнал p-ї пари, що навчає, відповідно, за допомогою яких визначається необхідний вектор вагових коефіцієнтів $\boldsymbol{w}^*$ такий, що

$$y_p = \operatorname{sgn}\left(\mathbf{w}^{*T}\mathbf{x}_\mathrm{p}\right) = y_p^*, \qquad p = \overline{1,P}.$$

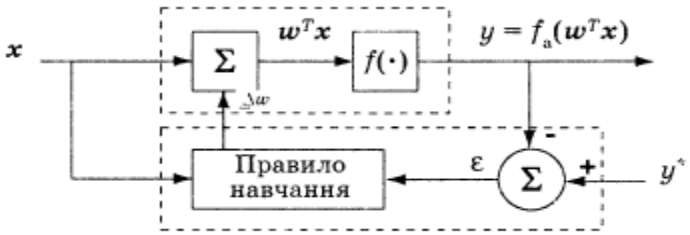


Рис. 8.3. Схема навчання перцептрона

У цьому випадку вектор $\mathbf{w}^*$ забезпечить правильну класифікацію перцептроном усіх пар, що навчають, із поданої множини (завершується цикл навчання).

Гіперплощина $\mathbf{w}^{*T}\mathbf{x}_p = 0$ ділить вхідний простір на два підпростори. Для $y_p^* = \mathbf{1}$ має виконуватися умова $\mathbf{w}^{*T}\mathbf{x}_p > 0$, а для $y_p^* = -1$ — умова $\mathbf{w}^{*T}\mathbf{x}_p < 0$.

Алгоритм навчання може бути записаний у такий спосіб:

$$\mathbf{w}_{p+1} = \mathbf{w}_p + \gamma e_p \mathbf{x}_p,$$

де $e_p = y_p^* - y_p$ — помилка класифікації; γ — параметр, що впливає на швидкість збіжності алгоритму (тривалість процесу навчання). Корекція ваг відповідно може відбуватися в режимах online й offline.

У режимі online корекція відбувається при поданні кожної навчальної пари $\left(\mathbf{x}_p, y_p^*\right)$, $p = \overline{1,P}$.

У режимі offline у M-му циклі (епосі) навчання подаються всі пари $(\mathbf{x}_p, y_p^*)$ й обчислюється середнє значення помилки класифікації

$$\overline{e}_m = \frac{1}{P}\sum_{p=1}^{p}(y_{p,M}^* - y_{p,M}),$$

що й використовується в алгоритмі навчання. Тут $y_{p,M}^*, y_{p,M}$ — бажані й реальні вихідні сигнали при пред'явленні p-ї пари, що навчає, в M-му циклі навчання відповідно.

Теорема збіжності для перцептрона, доведена Розенблаттом, стверджує, що *перцептрон може навчитися правильно класифікувати подані йому образи на скінченному числі навчальних пар*.

8.6. Алгоритм зворотного поширення помилки навчання багатошарових нейромереж прямого поширення

Алгоритм зворотного поширення помилки описаний у [11]. Він реалізує градієнтний метод мінімізації опуклого (звичайного квадратичного) функціонала

помилки в багатошарових мережах прямого поширення, що використовують моделі нейронів з диференціальними функціями активації. Застосування сигмоїдальних функцій активації, які мають відмінні від нуля похідні на всій області визначення, забезпечує правильне навчання й функціонування мережі. Процес навчання полягає у послідовному поданні мережі пар $\left(\mathbf{x}(i), y^*(i)\right)\, i = \overline{1,P}$, що навчають, де $\mathbf{x}(i)$і $y^*(i)$ — вектор вхідних і бажаних вихідних сигналів мережі відповідно, вивчені реакції на них мережі й корекції відповідно до реакції вагових параметрів (елементів вагової матриці).

Перед початком навчання всім вагам присвоюють невеликі випадкові значення (якщо задати всі значення однакові, а для правильного функціонування мережі знадобляться нерівні значення, мережа не навчатиметься).

Для реалізації алгоритму зворотного поширення необхідно:

1. Вибрати із заданої навчальної множини чергову навчальну пару $\left(\mathbf{x}(i), \mathbf{y}^*(i)\right),\ i = \overline{1,P}$ та подати на вхід мережі вхідний сигнал $\mathbf{x}(i)$.
2. Обчислити реакцію мережі $\mathbf{y}(i)$.
3. Визначити помилку $\mathbf{y}^*(i) - \mathbf{y}(i)$.
4. Скорегувати ваги так, щоб помилка була мінімальною.
5. Кроки 1-4 повторити для всієї множини пар $\left(\mathbf{x}(i), \mathbf{y}^*(i)\right)\ i = \overline{1,P}$ доти, поки на заданій множині помилка не досягне необхідної величини.

Таким чином, у процесі навчання мережі подача вхідного сигналу й обчислення реакції відповідає прямому проходу сигналу від вхідного шару до вихідного, а обчислення помилки й корекція вихідних параметрів — зворотному, коли сигнал помилки поширюється по мережі від її виходу до входу. При зворотному проході здійснюється пошарова корекція ваг, починаючи з вихідного шару.

Алгоритм зворотного поширення застосовують також щодо мереж з будь-якою кількістю шарів: як мереж прямого поширення, так і таких, що містять зворотні зв’язки. Обмежимося розглядом випадку навчання двох шарів мережі прямого поширення. Фрагмент такої мережі зображено на рис. 8.4.

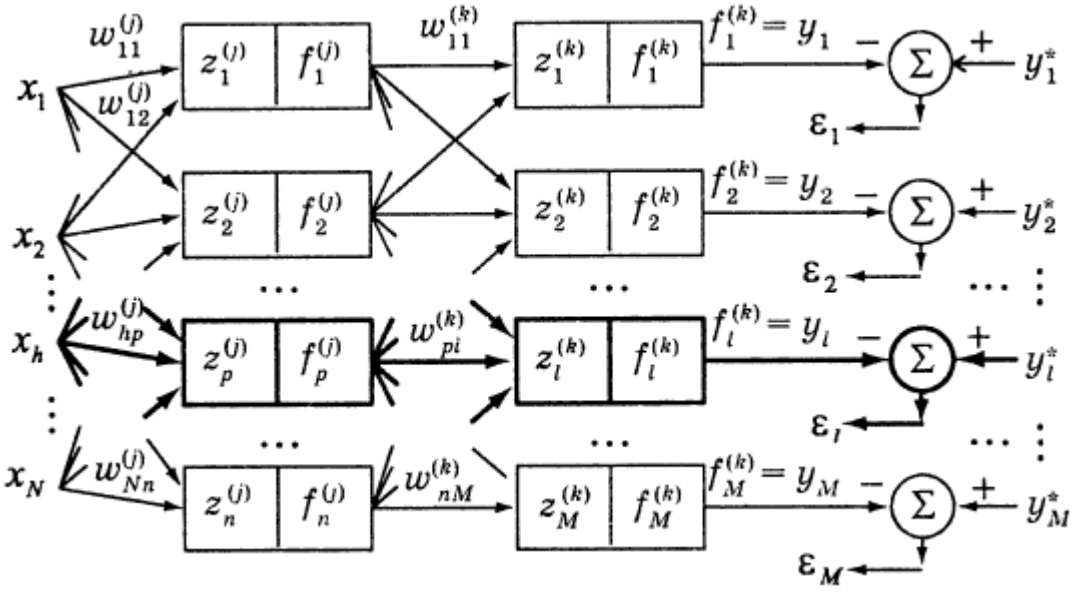


Рис. 8.4. Фрагмент мережі прямого поширення

На рисунку позначено: $w_{pl}^{(k)}$ — вага зв'язку p-го нейрона попереднього шару (j-го) з l-м нейроном наступного (k-го) шару; $f_p^{(j)}$ — активаційна функція p-го нейрона j-го шару; $z_p^{(j)}$ — зважена сума вихідних сигналів попереднього (i-го) шару, що надходять на вхід p-го нейрона наступного (j-го) шару.

Обчислення ваг нейронів вихідного шару

Розглянемо l-й нейрон вихідного (k-го) шару. Вихід даного нейрона є виходом мережі, тому його сигнал $y_i = f_l^{(k)}$ порівнюється з необхідним y_l^* й обчислюється помилка

$$\varsigma_l = y_l^* - f_l^{(k)}.$$

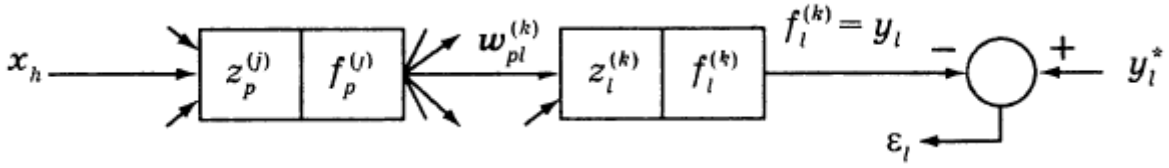


Рис. 8.5. Фрагмент багатошарової мережі

Використання квадратичного критерію якості навчання

$$\varsigma_l^2 = (y_l^* - f_l^{(k)})^2 \tag{8.1}$$

дозволяє отримати градієнтний алгоритм корекції ваг, що у цьому випадку набуває вигляду

$$\Delta w_{pl}^{(k)} = -\gamma_{pl} \frac{\partial \varsigma_l^2}{\partial w_{pl}^{(k)}},$$

де γ_{pl} — коефіцієнт, що впливає на швидкість навчання. Обчислення складної частинної похідної здійснюється за правилом

$$\frac{\partial \varsigma_l^2}{\partial w_{pl}^{(k)}} = \frac{\partial \varsigma_l^2}{\partial f_l^{(k)}} \frac{\partial f_l^{(k)}}{\partial z_l^{(k)}} \frac{\partial z_l^{(k)}}{\partial w_{pl}^{(k)}}. \tag{8.2}$$

З урахуванням (8.1), виду активаційних функцій і того, що $z_l^{(k)} = \sum_{p=1}^{n} w_{pl}^{(k)} f_p^{(j)}$, отримуємо

$$\frac{\partial \varsigma_l^2}{\partial f_l^{(k)}} = -2(y_l^* - y_l) = -2\varsigma_l; \tag{8.3}$$

$$\frac{\partial f_l^{(k)}}{\partial z_l^{(k)}} = \alpha f_l^{(k)}(1 - f_l^{(k)}); \tag{8.4}$$

$$\frac{\partial z_l^{(k)}}{\partial w_{pl}^{(k)}} = f_p^{(k)}. \tag{8.5}$$

Підстановка (8.3)-(8.5) у (8.2) дає

$$\Delta w_{pl}^{(k)} = 2\alpha\gamma_{pl}\varsigma_l f_l^{(k)}(1 - f_l^{(k)})f_p^{(k)} \tag{8.6}$$

або

$$w_{pl}^{(k)}(i+1) = w_{pl}^{(k)}(i) + \gamma_{pl}\delta_{pl}^{(k)} f_p^{(k)}, \tag{8.7}$$

де $\delta_{pl}^{(k)} = 2\alpha\varsigma_l f_l^{(k)}(1 - f_l^{(k)})$, i — номер ітерації.

Аналогічно обчислюються ваги інших нейронів вихідного шару. Всі величини, що входять у (8.7), є відомими. Присутня в (8.6), (8.7) помилка ς_l відмінна від нуля внаслідок того, що нейрони вихідного шару виробляють помилкові вихідні сигнали, тому що, по-перше, їм присвоєнні випадкові вагові коефіцієнти й, по-друге, нейрони прихованих шарів також виробляють помилкові сигнали. Помилка ς_l поширюється назад на попередні приховані шари й використовується для корекції ваг цих шарів.

Обчислення ваг нейронів прихованого шару

Фрагмент мережі для обчислення ваг нейронів прихованого шару наведено на рис. 8.6.

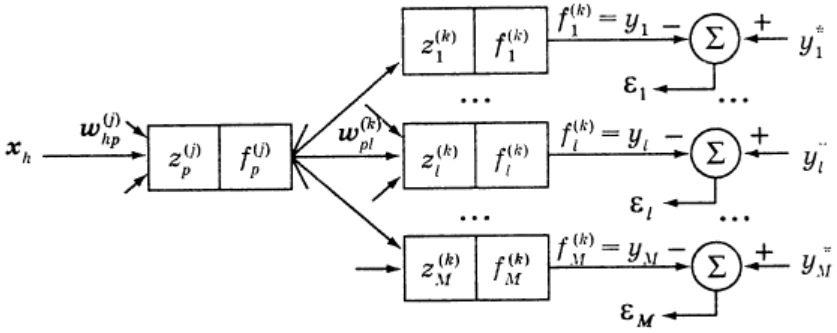


Рис. 8.6. Фрагмент мережі

Розглянемо p-й нейрон прихованого (j-го) шару. Оскільки вихідний сигнал цього нейрона надходить на виходи всіх нейронів вхідного шару, алгоритм настроювання записується в такий спосіб:

$$\Delta w_{hp}^{(j)} = -\gamma_{hp}\frac{\partial\varsigma^2}{\partial w_{hp}^{(j)}} = -\gamma_{hp}\sum_{i-1}^{M}\frac{\partial\varsigma_l^{(2)}}{\partial w_{hp}^{(j)}}. \tag{8.8}$$

Тут ς_l — помилка на i-му виході; γ_{hp} — параметр, що виконує ту саму роль, що й γ_{pl} у (8.7).

Оскільки квадратичний критерій якості в цьому випадку приймає вигляд

$$\varsigma = \sum_{l-1}^{M}[y_l^* - f_l^{(k)}]^2,$$

частинна похідна в (8.8), обчислюється так:

$$\frac{\partial\varsigma^2}{\partial w_{hp}^{(j)}} = \sum_{l=1}^{M}\frac{\partial\varsigma_l^2}{\partial f_l^{(k)}}\frac{\partial f_l^{(k)}}{\partial z_l^{(k)}}\frac{\partial z_l^{(k)}}{\partial f_p^{(j)}}\frac{\partial f_p^{(j)}}{\partial z_p^{(j)}}\frac{\partial z_p^{(j)}}{\partial w_{hp}^{(j)}}.$$

Для розглянутого випадку за аналогією з вищевикладеним отримуємо

$$\frac{\partial \varsigma_l^2}{\partial f_l^{(k)}} = -2\left[y_l^* - f_l^{(k)}\right] = -2\varsigma_l;$$

$$\frac{\partial f_i^{(k)}}{\partial z_i^{(k)}} = \alpha f_i^{(k)}(1 - f_i^{(k)});$$

$$\frac{\partial z_l^{(k)}}{\partial f_p^j} = w_{pl}^{(k)};$$

$$\frac{\partial f_p^{(j)}}{\partial z_p^{(j)}} = \alpha f_p^{(j)}(1 - f_p^{(j)});$$

$$\frac{\partial z_p^{(j)}}{\partial w_{hp}^{(j)}} = x_h.$$

Тут враховано, що $z_l^{(k)} = \sum_{p=1}^{m} w_{pl}^{(k)} f_p^{(j)}$; $z_l^{(j)} = \sum_{h=1}^{N} w_{hp}^{(j)} x_h$, а функції активації є сигмоїдальними. Таким чином,

$$\frac{\partial \varsigma^2}{\partial w_{hp}^{(j)}} = \sum_{i=1}^{m}(-2)\alpha\varsigma_l\left[f_l^{(k)}\left(1 - f_l^{(k)}\right)\right] w_{pl}^{(k)} \alpha\left[f_p^{(j)}\left(1 - f_p^{(j)}\right)\right] x_h =$$

$$= -\sum_{l=1}^{M} \delta_{pl}^{(k)} w_{pl}^{(k)} \frac{\partial f_p^{(j)}}{\partial z_p^{(j)}} x_h.$$

Позначимо $\delta_{hp}^{(j)} = \delta_{pl}^{(k)} w_{pl}^{(k)} \frac{\partial f_p^{(j)}}{\partial z_p^{(j)}}$.

Тоді

$$\frac{\partial \varsigma^2}{\partial w_{hp}^{(j)}} = -\sum_{l=1}^{M} \delta_{hp}^{(j)} x_h$$

і алгоритм корекції ваг приймає вигляд

$$\Delta w_{hp}^{(j)} = -\gamma_{hp} x_h \sum_{l=1}^{M} \delta_{hp}^{(j)},$$

або

$$w_{hp}^{(j)}(i+1) = \Delta w_{hp}^{(j)}(i) + \gamma_{hp} x_h \sum_{l=1}^{M} \delta_{hp}^{(j)}$$

Слід зазначити, що використання випадкових початкових значень вагових параметрів призводить до того, що при повторному моделюванні з іншими початковими значеннями форма поверхонь може бути іншою.

Опис бібліотек, у яких реалізовано різні модифікації методів навчання штучних нейромереж, наведено у [12, 13].

9. МЕРЕЖА ХОПФІЛДА

9.1. Модель Хопфілда

Розроблена Хопфілдом модель асинхронної ШНМ має такі ознаки [11]:

1. Мережа є одношаровою й містить N нейронів, число яких є одночасно числом входів і виходів мережі.
2. Кожен нейрон мережі пов'язаний з усіма іншими нейронами, а також має один вхід, на який подається вхідний сигнал.
3. Кожен нейрон не має власного зворотного зв'язку ($w_{ii} = 0$).
4. Ваги мережі є симетричними, тобто вага зв'язку між i-им й j-им нейронами дорівнює вазі зв'язку між j-им й i-им нейронами $(w_{ij} = w_{ji})$.
5. Кожен нейрон має порогову функцію активації.
6. Вхідними є двійкові сигнали.

Слід зазначити, оскільки «мережева» архітектура мозку містить зворотні зв'язки, то функціонування мережі Хопфілда, яка задовольняє обмеженням 1–6, відповідає природному процесу обробки інформації.

Схема мережі Хопфілда наведена на рис. 9.1. Внаслідок симетрії мережі Хопфілда іноді використовують різні форми її зображення. Так, наприклад, мережі чотирма нейронами, структурна схема якої зображена на рис. 9.1 відповідає графічне зображення, наведене на рис. 9.2.

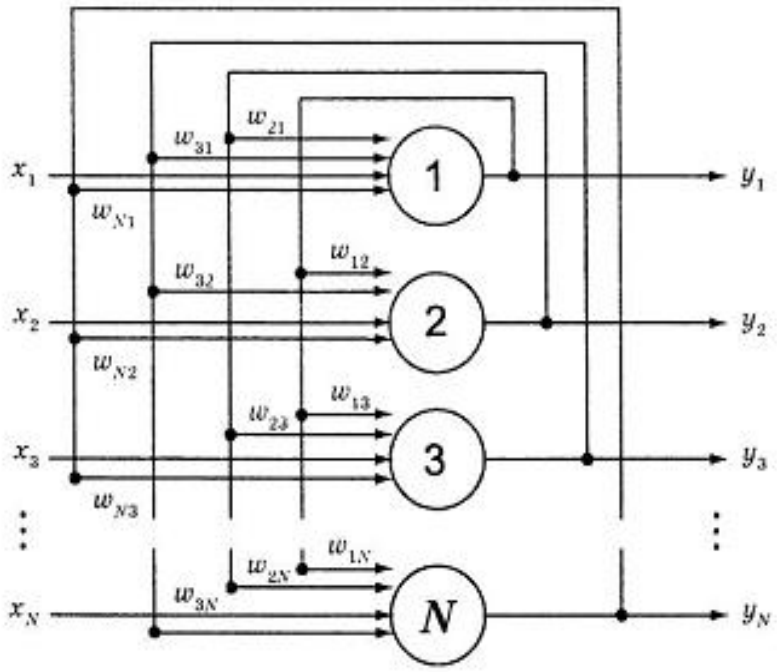


Рис. 9.1. Структурна схема мережі Хопфілда

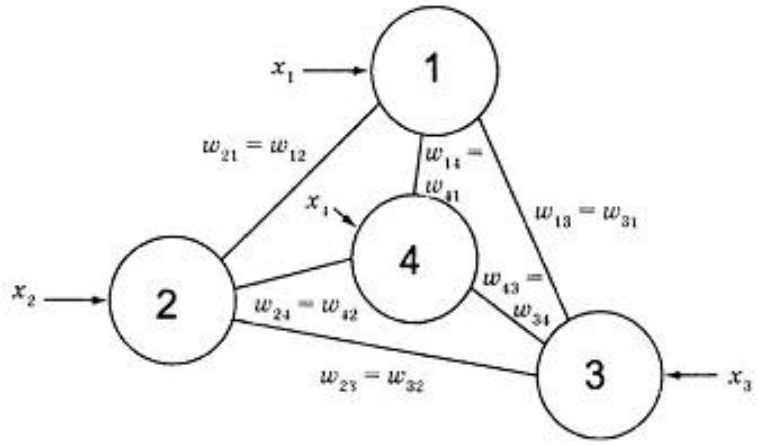


Рис. 9.2. Графічне зображення мережі Хопфілда

Дана мережа не використовує ані навчання з учителем, ані навчання без учителя. Вагові коефіцієнти в ній розраховуються тільки перед початком функціонування мережі на основі інформації про оброблювані дані, і все навчання мережі зводиться саме до цього розрахунку. З одного боку, подання апріорної інформації можна оцінювати як допомогу вчителя, а з іншого – мережа фактично запам'ятовує образи до того, як на її вхід надходять реальні дані, і не може змінювати свою поведінку, тому говорити про зворотний зв'язок із учителем не доводиться.

Слід зазначити, що в мережах зі зворотними зв'язками стан нейронів обчислюється доти, поки вони не виявляться сталими, не змінюваними згодом. Можна сказати, що мережа Хопфілда за певних умов збігається до сталого стану за скінченний час.

Мережі подаються P образів, які зображуються у вигляді N-вимірних векторів $x = (x_1, x_2, \dots, x_N)^T$, де N — кількість нейронів. Компоненти векторів приймають двійкові (0 та 1) або біполярні значення, наприклад (–1 та 1). Позначимо вектор вхідних сигналів, що описує p-й образ, як x^p $(p = \overline{1, P})$. Коли мережа розпізнає поданий образ, вона формує вектор вихідних сигналів, компоненти якого співпадають з відповідними компонентами образа. Тобто $y = x^p$, де $y = (y_1, y_2, \dots, y_N)$ — вектор вихідних сигналів мережі. В іншому випадку вихідний сигнал не збігатиметься з жодним із поданих.

Розрізняють *три стадії (фази) функціонування мережі* [11]:

- Ініціалізація;
- Подання вхідного образу;
- Обчислення стану нейронів.

На стадії *ініціалізація мережі* встановлюють значення вагових коефіцієнтів [11, 12]:

$$w_{ij} = \begin{cases} \sum_{p=0}^{P} x_i^p x_j^p, & i \neq j; \\ 0, & i = j \end{cases} \qquad (9.1)$$

Тут x_i^p й x_j^p — i-та j-та компоненти вектора x^p.

Цю стадію можна розглядати як стадію навчання мережі, після завершення якої вона здатна правильно відтворювати подані їй образи.

Подання мережі вхідного образу фактично здійснюється безпосереднім початковим (нульовим) присвоєнням компонент вихідних сигналів

$$y_i(0) = x_i, \quad i = \overline{1,N}, \tag{9.2}$$

Тому позначення на схемі мережі вхідних сигналів у явному вигляді носить суто умовний характер. Далі обчислюються величини

$$z_j(k+1) = \sum_{i=1}^{N} w_{ij} y_i(k) + x_j. \tag{9.3}$$

Визначаються послідовні стани нейронів, на підставі чого розраховуються відповідні значення вихідних сигналів

$$y_j(k+1) = f_a\left(z_j(k+1)\right) = \begin{cases} 1, & \text{якщо } z_j(k+1) > \theta_j; \\ 0, & \text{якщо } z_j(k+1) < \theta_j; \\ y_j(k), & \text{в інших випадках,} \end{cases} \tag{9.4}$$

де f_a — порогова активаційна функція; θ_j – заданий поріг.

Якщо вихідні значення змінилися, то за формулами (9.3) і (9.4) знову розраховуються стани й вихідні сигнали мережі, якщо не змінилися — робота мережі завершується (мережа перебуває в стійкому стані). У цьому випадку на виходах мережі сформувався вектор, що якнайкраще відповідає поданому образу.

Приклад 9.1. Розглянемо, до чого призведе порушення умов $w_{ii} = 0$. Припустимо, що є мережа, яка складається з двох нейронів, з матрицею ваг

$$w = \begin{pmatrix} -1 & -1 \\ -1 & -1 \end{pmatrix}$$

і порогами $\theta_l = -1 \quad (i = 1,2)$. Нехай у деякий момент часу k при подачі сигналу $x_i = 0 \ (i = 1,2)$ нейрони перебували в стані $z_i(k) = 1 \ (i = 1,2)$, тобто згідно з (9.4) на виході мережі було отримано сигнали $y_i = 1 \quad (i = 1,2)$. Відповідну мережу наведено на рис. 9.3.

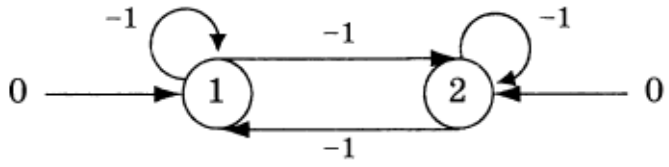


Рис. 9.3. Мережа, для якої порушується умова $w_{ii} = 0$

У наступний, (k+1)-й, момент часу, стани й вихідні сигнали нейронів дорівнюватимуть

$$z_i(k+1) = (-1) \cdot 1 + (-1) \cdot 1 + 0 = -2;$$

$$y_i(k+1) = 0, \quad i = 1,2.$$

Аналогічно для (k+2)-го моменти часу отримаємо

$$z_i(k+2) = (-1)\cdot 0 + (-1)\cdot 0 + 0 = 0;$$

$$y_i(k+1) = 1, \;\; i = 1{,}2.$$

Далі процес повторяється.
Таким чином, при порушенні умови $w_{ii} = 0$ мережа осцилює або зациклюється, тобто не досягає стійкого стану.

Приклад 9.2. Розглянемо наслідки порушення умови $w_{ij} = w_{ji}$ на прикладі мережі, що складається із двох нейронів і має матрицю ваг

$$w = \begin{pmatrix} 0 & -2 \\ -2 & 0 \end{pmatrix}$$

і нульовий поріг ($\theta_i = 0$).

Нехай у k-й момент часу при подачі на вхід нейронів сигналів $x_1 = 1, \; x_2 = -1$ на їхніх виходах були сигнали $y_1 = 1, \;\; y_1 = 0.$ Цю мережу зображено на рис. 9.4.

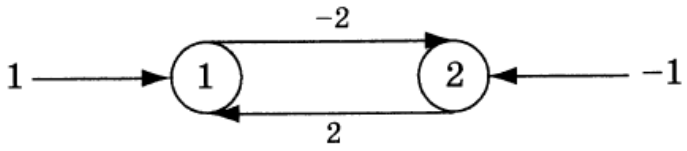


Рис. 9.4. Мережа, для якої порушується умова $w_{ij} = w_{ji}$

Скориставшись формулами (9.3) і (9.4), отримуємо, що в наступні моменти часу стану мережі й значення її вихідних сигналів зміняться у такий спосіб:
$k+1$:

$$z_1(k+1) = -2\cdot 0 + 1 = 1 \qquad y_1(k+1) = 1,$$
$$z_2(k+1) = 2\cdot 1 - 1 = \;\; 1 \qquad y_2(k+1) = 1;$$

$k+2$:

$$z_1(k+2) = -2\cdot 1 + 1 = -1 \qquad y_1(k+2) = 0,$$
$$z_2(k+2) = \;\; 2\cdot 1 - 1 = \;\; 1 \qquad y_2(k+2) = 1;$$

$k+3$:

$$z_1(k+3) = -2\cdot 1 + 1 = -1 \qquad y_1(k+3) = 0,$$
$$z_2(k+3) = \;\; 2\cdot 0 - 1 = -1 \qquad y_2(k+3) = 0;$$

$k+4$:

$$z_1(k+4) = -2\cdot 0 + 1 = 1 \qquad y_1(k+4) = 1,$$
$$z_2(k+4) = 2\cdot 0 - 1 = -1 \qquad y_2(k+4) = 0.$$

Мережа повернулася в початковий стан, і далі процес повторюється. Таким чином, при порушенні умови $w_{ij} = w_{ji}$ мережа також не досягає стійкого стану, тобто осцилює.

Активація мережі Хопфілда може здійснюватися асинхронно, коли на кожному такті змінює свій стан тільки одним випадковим способом обраний нейрон, і синхронно, коли всі нейрони змінюють свій стан одночасно.

9.2. Навчання в мережі Хопфілда

Робота мережі Хопфілда може бути пояснена термінами енергетичного ландшафту [11]. Є ландшафт, що являю собою гористу місцевість, на вершині якої перебуває куля. Потім куля котиться по схилу, поки не зупиниться в якій-небудь низині (западині). Ці низини відбивають стійкі стани мережі, і кожна з них відповідає певному поданому образу (образу навчання). У цій моделі куля, що має велику потенційну енергію, котиться вниз з меншою потенційною енергією, досягаючи локального мінімуму. Щоб знову опинитися в початковому стані, вона повинна здійснити у фізичному значенні роботу, тобто витратити енергію. Таким чином, робота мережі Хопфілда може бути охарактеризована деякою енергетичною функцією.

Якщо в процесі аналізу перцептрона вибір такої функції (функціонала) особливих ускладнень не викликає [11], оскільки відомі реальні й бажані значення виходів мережі, то в цьому випадку ситуація дещо інша. По-перше, структура мережі Хопфілда не дозволяє заздалегідь визначити бажану або необхідну послідовність станів нейронів. По-друге, наявність зворотних зв'язків призводить до того, що виходи мережі після кожного такту функціонування у свою чергу подаються на її входи. Крім того, має бути врахована відсутність у нейронів власних зворотних зв'язків. Зазначені особливості мережі відображаються в енергетичній функції вигляду

$$E = -\frac{1}{2}\sum_{i}\sum_{j\neq i} w_{ij} y_i y_j - \sum_{i} y_i x_i + \sum_{i} y_i \theta_i, \qquad (9.5)$$

де θ_i — значення порога i-го нейрона; w_{ij} – ваги зв'язку i-го й j-го нейронів; x_i — зовнішній вихідний сигнал i-го нейрона (постійна величина на протязі кожного моменту часу або такту); y_i — значення вихідного сигналу i-го нейрона.

Хопфілд довів [11], що при активації мережі функція (9.5) не зростає й досягає локального мінімуму в деякому сталому стані. А оскільки кількість таких стійких станів обмежена, мережа за певних умов досягає одного з них за скінченну кількість ітерацій. При цьому, як уже зазначалося, низини енергетичного ландшафту (енергетичні мінімуми) відповідають збереженим образам. Щоб мережа Хопфілда правильно класифікувала образи, ці низини не мають перекриватися.

9.2.1. Накопичення образів у мережі Хопфілда

Функціонал (9.5) можна розглянути як критерій помилки (чим далі від бажаного образу, тим більше значення він приймає). Щоб зберігати подані на вхід мережі образи, необхідно мінімізувати значення енергетичної функції для кожного з них, тобто визначати локальні мінімуми енергетичного ландшафту. Однак додавання нових образів не має знищувати вже наявну інформацію. А оскільки вся інформація щодо образів, які зберігаються, міститься у ваговій матриці, задача навчання зводиться до визначення значень ваг, що мінімізують енергетичний функціонал.

При мінімізації даного функціонала необхідно враховувати наступне. Як видно з (9.5), мінімум даного виразу перебуває у від'ємній області. Для виконання умови $\sum_i y_i\, \theta_i < 0$ необхідно, щоб компоненти y_i й θ_i постійно мали протилежні знаки, тому доцільно покласти $\theta_i = 0$, що призводить до критерію вигляду

$$E = -\frac{1}{2}\sum_i \sum_{j \neq i} w_{ij} y_i y_j - \sum_i x_i y_i . \quad (9.6)$$

Позначимо за аналогією з вищевикладеним x_m, y_m — компоненти m-го образу. Тоді матрицю ваг w_{ij} можна розбити на дві підматриці, перша з яких (w_{ij}^1) відображає зв'язки всіх образів, крім m-го, а друга (w_{im}) — тільки m-ий образ

$$E = -\frac{1}{2}\sum_i \sum_{j \neq i} w_{ij}^1 x_i x_j - \sum_{i \neq m} x_i y_i - \frac{1}{2}\sum_{i \neq m} w_{im} y_i y_m - x_m y_m . \quad (9.7)$$

Перші два доданки (9.7) можна розглядати як збурення, що відповідає загальному впливу на функціонал усіх образів, крім m-го, а інші — як корисний сигнал. Отже, щоб мінімізувати функціонал (9.7) для заданого m-го образу, необхідно розглянути другий доданок, тобто

$$E_m = -\frac{1}{2}\sum_{i \neq m} w_{im} y_i y_m - x_m y_m . \quad (9.8)$$

Мінімізація (9.8) еквівалентна максимізації

$$\sum_{i \neq m} w_{im} y_i y_m + x_m y_m \quad (9.9)$$

Розглянемо перший доданок. Оскільки $y_m \in \{-1,1\}$, то завжди $(y_m)^2 > 0$. Якщо вибрати $w_{im} = y_i y_m$, то

$$\sum_{i \neq m} w_{im} y_i y_m = \sum_{i \neq m} (y_i)^2 (y_m)^2 . \quad (9.10)$$

Отже, вибір $w_{im} = y_i y_m$ забезпечує максимум функціоналу (9.9). Неважко побачити, що при нульових початкових умовах подача на вхід образу x призводить відповідно у (9.3) і (9.4) до появи на виході мережі вектора $y = x$. Цим пояснюється вибір значень вагових коефіцієнтів на стадії ініціалізації у вигляді (9.1).

9.2.2. Виклик образу

Після того, як усі подані образи будуть збережені мережею (побудовано енергетичний ландшафт), інформація про них може бути отримана шляхом мінімізації функціонала помилки на фазі навчання. Використання енергетичного ландшафту дозволяє визначити вплив окремої вершини на енергію, а мінімізація енергії — шлях переходу мережі в будь який стійкий стан.

Функціонування мережі Хопфілда полягає у наступному:

1. Навчання.

Подаються образи x^p $(p = \overline{1, P})$. За формулою (9.1) обчислюються елементи w_{ij} матриці ваг і для активних нейронів (що перебувають у стані «+1») на основі їх поточного й попереднього стану обчислюється енергія кожного образу

$$E = -\frac{1}{2}\sum_{i=1}^{N} x_i^p (k+1) \sum_{j=1}^{N} w_{ij} x_j^p (k).$$

2. Розпізнавання.

Подаються образи, можливо спотворені, й за формулами (9.3), (9.4) обчислюються послідовні стани й значення вихідних сигналів нейронів до досягнення ними стійких станів.

Приклад 9.3. Розглянемо функціонування мережі Хопфілда, що складається з 7 нейронів і біполярної порогової функції активації з порогом $\theta = 0$, для навчання якої використовуються образи

$$x^1 = (1, -1, -1, 1, -1, 1, 1)^T \text{ і } x^2 = (-1, -1, 1, -1, 1, -1, 1)^T$$

і яка має розпізнавати створений образ $x^3 = \tilde{x}^2 = (-1, -1, \underline{-1,} - 1, 1, -1, 1)^T$ (спотворений біт підкреслений).

На рис. 9.7 наведена ця мережа й позначені деякі її зв'язки.

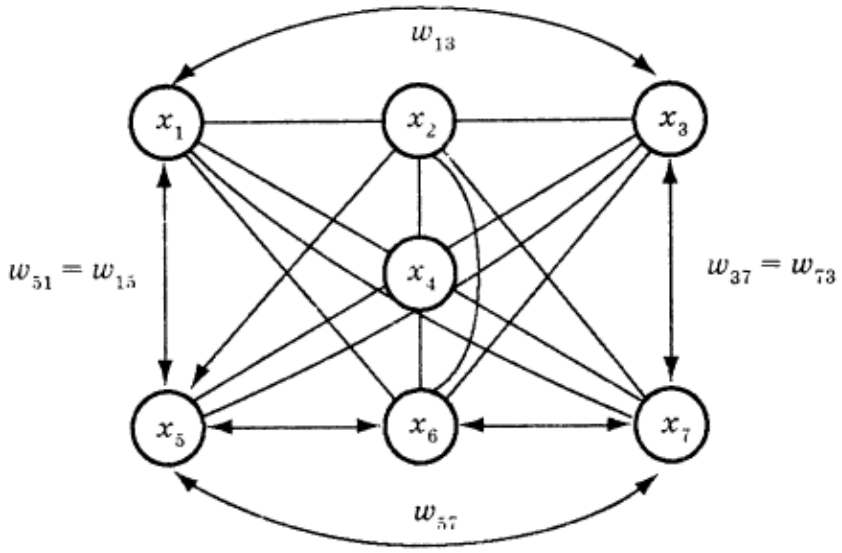


Рис .9.5. Мережа Хопфілда, що складається з 7 нейронів

Елементи w_{ij} матриці ваг W розмірності 7×7, що має вигляд

$$W = W^1 + W^2 = \begin{pmatrix} w_{11} = 0 & w_{12} & w_{13} & \dots & w_{17} \\ w_{21} & w_{22} = 0 & w_{23} & \dots & w_{27} \\ \dots & \dots & \dots & \dots & \dots \\ w_{71} & w_{72} & w_{73} & \dots & w_{77} = 0 \end{pmatrix}$$

де W^i — матриця ваг для *i*-го (*i*=1,2) образу, що навчає, визначають за формулою (2.3)

$$w_{12} = 1 \cdot (-1) + (-1) \cdot (-1) = 0; \quad w_{13} = 1 \cdot (-1) + (-1) \cdot 1 = -2;$$
$$w_{14} = 1 \cdot 1 + (-1) \cdot (-1) = 2; \quad w_{15} = 1 \cdot (-1) + (-1) \cdot 1 = -2;$$
$$w_{16} = 1 \cdot 1 + (-1) \cdot (-1) = 2; \quad w_{17} = 1 \cdot 1 + (-1) \cdot 1 = 0 \text{ і т. д.}$$

Після визначення всіх ваг матриця набуває вигляду

$$W = \begin{pmatrix} 0 & 0 & -2 & 2 & -2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 \\ -2 & 0 & 0 & -2 & 2 & -2 & 0 \\ 2 & 0 & -2 & 0 & -2 & 2 & 0 \\ -2 & 0 & 2 & -2 & 0 & -2 & 0 \\ 2 & 0 & -2 & 2 & -2 & 0 & 0 \\ 0 & -2 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Обчислимо енергію кожного образу, беручи до уваги тільки активні нейрони. Для першого образу

$$E^1 = \sum_{i=1}^{7} E_i^1 = E_1^1 + E_4^1 + E_6^1 + E_7^1,$$

$$E_1^1 = -\frac{1}{2}[x_1(w_{13}x_3 + w_{14}x_4 + w_{15}x_5 + w_{16}x_6)] =$$

$$= -\frac{1}{2}[(-2)\cdot(-1) + 2\cdot 1 + (-2)\cdot(-1) + 2\cdot 1] = -4;$$

$$E_4^1 = -\frac{1}{2}[x_4(w_{41}x_1 + w_{43}x_3 + w_{45}x_5 + w_{46}x_6)] =$$

$$= -\frac{1}{2}[2\cdot 1 + (-2)\cdot(-1) + (-2)\cdot(-1) + 2\cdot 1] = -4;$$

$$E_6^1 = -\frac{1}{2}[x_6(w_{61}x_1 + w_{63}x_3 + w_{64}x_4 + w_{65}x_5)] =$$

$$= -\frac{1}{2}[2\cdot 1 + (-2)\cdot(-1) + 2\cdot 1 + (-2)\cdot(-1)] = -4;$$

$$E_7^1 = -\frac{1}{2}[x_7(w_{72}x_2)] = -\frac{1}{2}[(-2)\cdot(-1)] = -1.$$

Тоді $E^1 = -13$.

Аналогічно отримуємо для другого образу

$$E^2 = \sum_{i=7}^{7} E_i^2 = E_3^2 + E_5^2 + E_7^2 = -9.$$

Розпізнавання образу $x^3 = \tilde{x}^2$ починаємо з визначення його енергії, що після проведення аналогічних обчислень дорівнюватиме

$$E^3 = \sum_{i=1}^{7} E_i^3 = E_5^3 + E_7^3 = -5.$$

За формулами (9.3) і (9.4) обчислюємо нові стани нейронів і відповідні значення вихідних сигналів, які з урахуванням біполярної активаційної функції дорівнюватимуть

$$z_j^3 = \sum_{i=1}^{7} w_{ij}x_i^3, \quad j = \overline{1,7}$$

$$z_1^3 = w_{31}x_3^3 + w_{41}x_4^3 + w_{51}x_5^3 + w_{61}x_6^3 + x_1^3 =$$

$$= (-2)\cdot(-1) + 2\cdot(-1) + (-2)\cdot 1 + 2\cdot(-1) - 1 = -5 < 0; \; y_1^3 = -1;$$

$$z_2^3 = w_{72}x_7^3 + x_2^3 = (-2)\cdot 1 - 1 = -3 < 0; \; y_2^3 = -1;$$

$$z_3^3 = w_{13}x_1^3 + w_{43}x_4^3 + w_{53}x_5^3 + w_{63}x_6^3 + x_3^3 =$$

$$= (-2)\cdot(-1) + (-2)\cdot(-1) + 2\cdot 1 + (-2)\cdot(-1) - 1 = 7 > 0; \; y_3^3 = -1;$$

$$z_4^3 = w_{14}x_1^3 + w_{34}x_3^3 + w_{54}x_5^3 + w_{64}x_6^3 + x_4^3 =$$

$$= 2 \cdot (-1) + (-2) \cdot (-1) + (-2) \cdot 1 + 2 \cdot (-1) - 1 = -5 < 0;\ \ y_4^3 = 1;$$
$$z_5^3 = w_{15}x_1^3 + w_{35}x_3^3 + w_{45}x_4^3 + w_{65}x_6^3 + x_5^3 =$$
$$= (-2) \cdot (-1) + 2 \cdot (-1) + (-2) \cdot (-1) + (-2) \cdot (-1) + 1 = 5 > 0;\ \ y_5^3 = -1;$$
$$z_6^3 = w_{16}x_1^3 + w_{36}x_3^3 + w_{46}x_4^3 + w_{56}x_5^3 + x_6^3 =$$
$$= 2 \cdot (-1) + (-2) \cdot (-1) + 2 \cdot (-1) + (-2) \cdot 1 - 1 = -5 < 0;\ \ y_6^3 = -1;$$
$$z_7^3 = w_{27}x_2^3 + x_7^3 = (-2) \cdot (-1) + 1 = 3 > 0;\ \ \ y_7^3 = 1.$$

Таким чином, після подання $x^3 = \widetilde{x^2}$ мережа перейде в стійкий стан

$$(-1, -1, 1, -1, 1, -1, 1)^T,$$

що відповідає образу x^2 з енергією, як неважко переконатися, $E = -9$. Отже, мережа правильно розпізнала (відновила створений образ).

10. НЕЙРОМЕРЕЖА КОХОНЕНА

У багатьох моделях ШНМ вирішальну роль відіграють зв'язки між нейронами, які визначаються ваговими коефіцієнтами й зазначають місце нейрона в мережі. Однак у біологічних системах, наприклад, у мозку, сусідні нейрони, отримуючи аналогічні вхідні сигнали, реагують на них подібним чином, тобто групуються, утворюючи деякі області. Оскільки під час обробки багатовимірного вхідного образу здійснюється його проектування на область меншої розмірності зі збереженням його топології, нерідко подібні мережі називають *мапами (self-organizing feature map).* У таких мережах істотним є врахування взаємного розташування нейронів одного шару.

Мережа Кохонена відноситься до мереж, що самоорганізуються, які під час надходження вхідних сигналів, на відміну від мереж, що використовують навчання із учителем, не отримують інформацію про бажаний вихідний сигнал. У зв'язку з цим неможливо сформувати критерій настроювання, заснований на неузгодженості реальних і необхідних вихідних сигналів ШНМ, тому вагові параметри мережі корегують, виходячи з інших міркувань. Усі подані вхідні сигнали із заданої навчальної множини мережа у процесі навчання розділяє на класи, будуючи так звані топологічні мапи.

10.1. Структура мережі Кохонена

Мережа Кохонена використовує таку модель (рис. 10.1): мережа складається з M нейронів, що утворюють прямокутні гратки на площині — шар. До нейронів, розташованих в одному шарі, що є двовимірною площиною, підходять нервові волокна, по яких надходить N-вимірний вхідний сигнал. Кожен нейрон характеризується своїм розміщенням у шарі й ваговим коефіцієнтом. Розміщення нейронів, у свою чергу, характеризується деякою метрикою й визначається топологією шару, при якій сусідні нейрони під час навчання впливають один на одного сильніше, ніж розташовані далі. Кожен нейрон утворює зважену суму вхідних сигналів. Наявність зв'язків між нейронами призводить до того, що при збудженні одного з них можна обчислити збудження інших нейронів у шарі, причому це збудження зі збільшенням відстані від збудженого нейрона зменшується. Тому центр реакції шару, що виникає у відповідь на отримане подразнення, відповідає місцезнаходженню збудженого нейрона. Зміна вхідного сигналу, що навчає, призводить до максимального збудження іншого нейрона й відповідно — до іншої реакції шару.

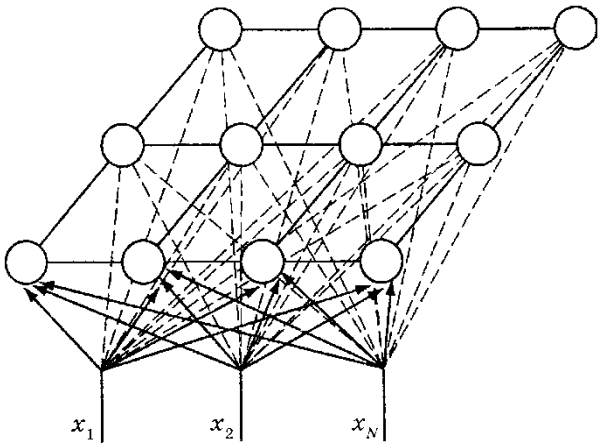


Рис. 10.1. Модель мережі Кохонена

Мережа Кохонена може розглядатися як подальший розвиток мережі векторного квантування. Відмінність їх полягає у способах навчання.

10.2. Навчання мережі Кохонена

Замість того, щоб шукати місцезнаходження нейрона шляхом розв'язання загальних рівнянь збудження, Кохонен істотно спростив розв'язання задачі, виділяючи зі всіх нейронів шару лише один *с-й* нейрон, для якого зважена сума вхідних сигналів максимальна

$$c = \arg\min_j \left(\mathbf{x}^T \mathbf{w}_j \right) \tag{10.1}$$

Зазначимо, що корисною операцією попередньої обробки вхідних векторів є їхня нормалізація $\overline{x}_i = \frac{x_i}{\|\mathbf{x}\|}$, $i = 1,...,N$, яка перетворює вектори вхідних сигналів в одиничні з тим же напрямком.

Отже, буде активований тільки той нейрон, вектор ваг якого ш найближчий до вхідного вектора *x.* А оскільки перед початком навчання невідомо, який саме нейрон активуватиметься при поданні мережі конкретного вхідного вектора, мережа навчається без учителя, тобто *самонавчається.*

Вводячи потенційну функцію — функцію відстані f_{ij} («сусідства») між *i-м* й *j*-м нейронами з місцезнаходження r_i й r_j відповідно, що монотонно спадає зі збільшенням відстані між цими нейронами, Кохонен запропонував такий алгоритм корекції ваг [11]:

$$\mathbf{w}_j(k+1) = \mathbf{w}_j(k) + \alpha(k)\, f_{cj}\left(\mathbf{x}(k) - \mathbf{w}_j(k)\right), \tag{10.2}$$

де $\alpha(k) \in (0,1]$ — коефіцієнт підсилення, що змінюється в часі (звичайно вибирають $\alpha = 1$ на першій ітерації, поступово зменшуючи в процесі навчання до нуля); $f_{ij}(k)$ — монотонно спадна функція

$$f_{ij}(k) = f\left(\|r_i - r_j\|, k\right) = f(d,k) = f(d,\sigma),$$

де r_i та r_j — вектори, що визначають положення нейронів *i* й *j* у ґратках. При прийнятій метриці $d = \|r_i - r_j\|$ функція $f_{ij}(k)$ із зростанням часу *k* прямує до нуля. На практиці замість параметра часу *k* використовують параметр відстані σ , що задає величину області «сусідства» і зменшується із часом до нуля.

Вибір функції $f_{i_i}(k)$ також впливає на величини ваг усіх нейронів у шарі. Очевидно, що для нейрона-переможця *c* $f_{cc}\left(\|r_c - r_c\|\right) = f_c(0) = 1$.

Зважаючи на те, що визначення нейронів-переможців у мережі Кохонена й у LVQ відбувається ідентично, то й корекція ваг цих нейронів здійснюється однаково.

На рис. 10.2 наведено приклад зміни двовимірних ваг мапи $w_j = \left(w_{j1}, w_{j2}\right)^T$, що утворює ланцюг. З появою вхідного образу **x** найбільш сильно змінюється ваговий вектор нейрона-переможця 5, менш сильно — ваги розташованих поруч із ним нейронів 3, 4, 6, 7. А оскільки нейрони 1,2,8,9 перебувають поза областю «сусідства», їхні вагові коефіцієнти не змінюються.

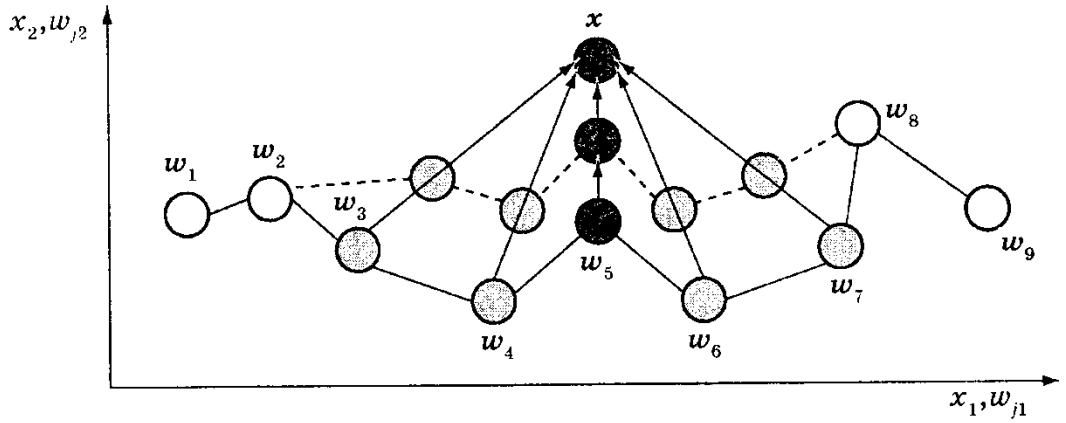


Рис. 10.2. Зміна ваг мапи Кохонена

Отже, алгоритм навчання мережі Кохонена може бути описаний так:

1. *Ініціалізація.*

Ваговим коефіцієнтам усіх нейронів привласнюються малі випадкові значення й здійснюється їхня нормалізація. Вибирається відповідна потенційна функція $f_{ij}(d)$ і призначається початкове значення коефіцієнта підсилення α_0.

2. *Вибір сигналу, що навчає.*

Із усієї множини векторів навчальних вхідних сигналів відповідно до функції розподілу $P(\mathbf{x})$ вибирається один вектор $\mathbf{x}$, що представляє «сенсорний сигнал», поданий мережі.

3. *Аналіз відгуку (вибір нейрона)*

За формулою (1.4) визначається активований нейрон.

4. *Процес навчання.*

Відповідно до алгоритму (1.5) змінюються вагові коефіцієнти активованого й сусідніх з ним нейронів доти, поки не буде отримано необхідного значення критерію якості навчання або не буде подане задане число вхідних векторів, що навчають. Остаточне значення вагових коефіцієнтів збігається з нормалізованими векторами входів.

Оскільки мережа Кохонена здійснює проектування N-вимірного простору образів на M-вимірну мережу, аналіз збіжності алгоритму навчання є досить складною задачею. Однак властивості алгоритму можуть бути продемонстровані на простих прикладах.

Приклад 10.1. Розглянемо одновимірний випадок. Припустимо, що мережа складається з одного нейрона, а вхідний сигнал $x \in [a, b]$. Нехай початкове значення ваги дорівнює x_1 (рис. 10.3)

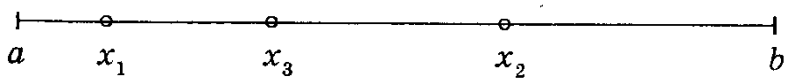


Рис. 10.3. Навчання мережі, що складається з одного нейрона

Кожна зміна x залежить тільки віддаленості від a й b, тобто

$$\frac{dx}{dt} = \frac{\alpha}{2}[(b-x)+(a-x)] = \alpha\left(\frac{a+b}{2} - x\right),$$

де $\alpha \in (0,1]$.

Тоді переміщення x_1 у x_2 відбуватиметься за правилом

$$x_2 = x_1 + \alpha\left(\frac{a+b}{2} - x_1\right) = \frac{a+b}{2} + (1-\alpha)\left(x_1 - \frac{a+b}{2}\right).$$

Аналогічно

$$x_3 = x_2 + \alpha\left(\frac{a+b}{2} - x_2\right) = \frac{a+b}{2} + (1-\alpha)^2\left(x_1 - \frac{a+b}{2}\right).$$

Після n-ї ітерації отримуємо

$$x_n = \frac{a+b}{2} + (1-\alpha)^{n-1}\left(x_1 - \frac{a+b}{2}\right),$$

звідки випливає, що

$$\lim_{n\to\infty} x_n = \frac{a+b}{2},$$

тобто навчання завершується, коли значення ваги ділитиме інтервал $[a,b]$ навпіл. Отже, x_n займе стійке розташування в середині інтервалу $[a,b]$.

Приклад 10.2. Розглянемо одновимірний шар, що складається з нейронів, які ділять заданий інтервал $[a,b]$ на N рівних частин. Нехай ваги нейронів упорядковані, тобто $a < x_1 < x_2 < \cdots < x = b$ (рис. 10.4).

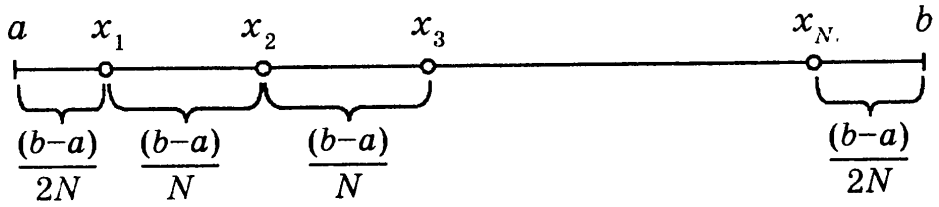


Рис. 10.4. Навчання мережі

Тоді кожна вага мережі може бути обчислена за формулою

$$x_i = a + (2i-1)\frac{b-a}{2N}, \qquad i = \overline{1,N}.$$

Оскільки ці ваги ділять інтервал $[a,b]$ на N рівних частин, то вони займають на цьому інтервалі стійкі розміщення.

Приклад 10.3. Розглянемо двовимірний шар, що складається з $N \times N$ нейронів. Вхідний вектор складатиметься із двох компонентів $x = (x_1 x_2)^T$, $x_1 \in [a,b]$, $x_2 \in [c,d]$. Відповідно вектор вагових коефіцієнтів ij-нейрона має вигляд $\omega^{ij} = \left(\omega_1^{ij}, \omega_2^{ij}\right)^T$. Припустимо, що всі компоненти вагового вектора впорядковані $\omega_1^{ij} < \omega_1^{ik}$, $j < k$, $\omega_2^{ij} < \omega_2^{mj}$, $i < m$. Визначаючи для кожного стовпця нашої матриці ваг середнє значення

$$\omega_1^j = \frac{1}{N}\sum_{i=1}^{N} \omega_1^{ij}, \quad j = \overline{1,N},$$

з урахуванням упорядкованості компонентів отримуємо

$$a < \omega_1^1 < \omega_1^2 < \cdots < \omega_1^N < b.$$

Аналогічно отримуємо для кожного рядка матриці ваг

$$c < \omega_2^1 < \omega_2^2 < \cdots < \omega_2^N < d.$$

За аналогією із прикладом 1.3 слід зазначити, що в процесі навчання вагові коефіцієнти розіб'ють інтервали $[a, b], [c, d]$ на рівні частини, тобто нейрони розташуються у вузлах ґраток.

Якби з кожним нейроном шару асоціювався один вхідний вектор, то вектор ваг будь-якого нейрона шару Кохонена міг би бути навчений за допомогою одного обчислення, оскільки вага нейрона-переможця корегувалися б з $\alpha = 1$ (для одновимірного випадку вага відразу б потрапляла в центр відрізка $[a, b]$). Однак звичайно навчальна множина містить багато подібних між собою вхідних векторів, і мережа Кохонена має бути навчена активувати той самий нейрон для кожного з них. Це досягається усередненням вхідних векторів шляхом зменшення величини α при поданні кожного наступного вхідного сигналу. Отже, ваги, асоційовані з нейроном, усередняться й приймуть значення поблизу «центра» вхідних сигналів, для яких даний нейрон є «переможцем».

Приклад 10.4. Нехай мережа, що складається із трьох нейронів, має класифікувати такі пропоновані їй нормалізовані вектори:

$$x(1) = (0{,}8;\ 0{,}6)^T; \qquad x(2) = (0{,}7071;\ 0{,}7071)^T;$$
$$x(3) = (0{,}6;\ -0{,}8)^T; \qquad x(4) = (0{,}1961;\ -0{,}9806)^T;$$
$$x(5) = (-0{,}9806;\ -0{,}1961)^T; \qquad x(6) = (-0{,}9806;\ 0{,}1961)^T.$$

Випадковим способом обрані нормалізовані початкові значення ваг дорівнюють
$w_1(0) = (1{,}000; 0{,}000)^T;\ w_2(0) = (0; -1)^T;\ w_3(0) = (-0{,}707; 0{,}707)^T.$
Розміщення векторів x й $w_i(0)$ зображено на рис. 10.5

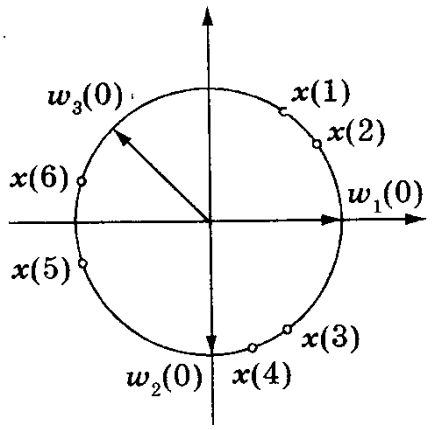


Рис. 10.5. Розміщення векторів x і $\omega(0)$

Подання мережі вектора $x(0)$ дає

$$w_1^T(0)x(1) = \begin{bmatrix} 1 & 0 \\ 0 & -1 \\ -0,7 & 0,7 \end{bmatrix}^T \begin{bmatrix} 0,8 \\ 0,6 \end{bmatrix} = \begin{bmatrix} 0,8 \\ -0,6 \\ -0,14 \end{bmatrix}.$$

Оскільки $max[w_1^T(0)x(1)] = 0,8$, переможцем буде перший нейрон. Скорегуємо його ваги відповідно до алгоритму (1.5), прийнявши $\alpha = 0,5$,

$$w_1(1) = w_1(0) + 0,5\big(x(1) - w_1(0)\big) = \begin{bmatrix} 1,0 \\ 0 \end{bmatrix} + 0,5\left(\begin{bmatrix} 0,8 \\ 0,6 \end{bmatrix} - \begin{bmatrix} 1 \\ 0 \end{bmatrix}\right) = \begin{bmatrix} 0,9 \\ 0,3 \end{bmatrix}.$$

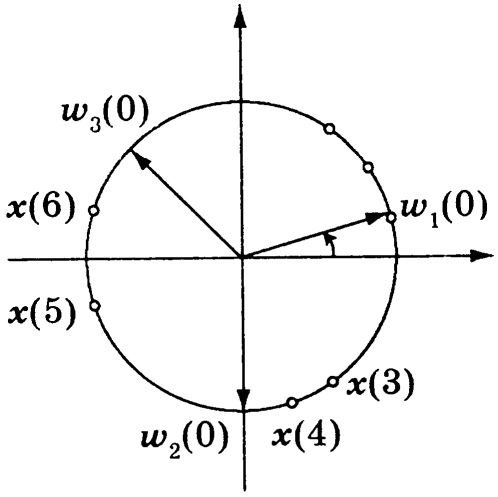


Рис. 10.6. Корекція вагового вектора w_1

Ваги змінилися таким чином, що вектор перемістився до вхідного вектора $x(1)$ *(див.* рис. 10.7). Оскільки порядок подання вхідних векторів довільний, то при поданні $x(4)$ отримуємо

$$w_1^T(1)x(4) = -0,11769;\ w_2^T(0)x(4) = 0,9806;\ w_3^T(0)x(4) = -0,8319.$$

Оскільки переможцем виявився другий нейрон, скорегуємо його вагові коефіцієнти, міняючи його розміщення (рис. 1.14) переміщенням до $x(4)$

$$w_2(1) = w_2(0) + 0,5\big(x(4) - w_2(0)\big) =$$

$$= \begin{bmatrix} 0 \\ -1 \end{bmatrix} + 0,5\left(\begin{bmatrix} 0,1961 \\ -0,9806 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix}\right) = \begin{bmatrix} 0,1 \\ -0,99 \end{bmatrix}.$$

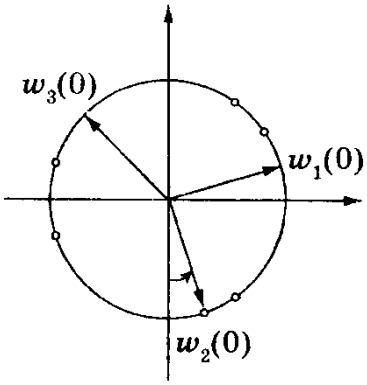


Рис. 10.7. Корекція вагового вектора w_2

Процес навчання завершиться, коли кожен ваговий вектор стане прототипом різних поданих мережі вхідних сигналів, тобто відповідатиме своїй групі вхідних сигналів. В остаточному підсумку вектори ваг займуть положення приблизно такі ж, які наведено на рис. 10.8.

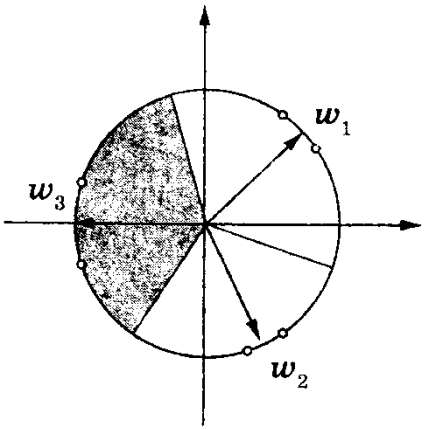


Рис. 10.8. Остаточне розміщення вагових векторів

10.3. Вибір функції «сусідства»

На рис. 10.9 зображено шари нейронів з нейроном-переможцем, позначеним чорним кільцем. Оскільки ваги всіх затемнених нейронів змінюються по-різному, залежно від їхньої далекості від нейрона-переможця, найбільш простим є вибір як f_{ij} деякої величини, що дорівнює одиниці при $j = i$, меншої одиниці для затемнених нейронів, тобто нейронів, що лежать у безпосередній близькості від активованого нейрона, і нуль для інших, позначених світлими кружками.

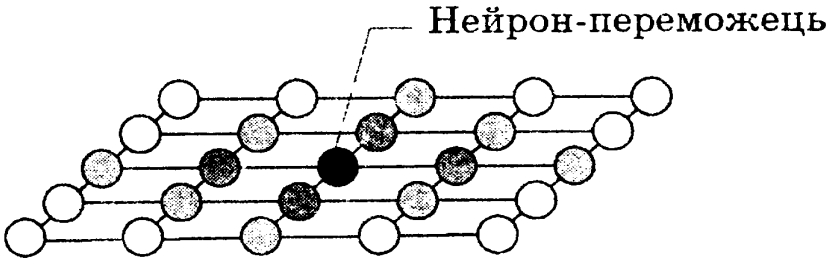


Рис. 10.9. Шар нейронів Кохонена

На практиці ж як f_{ij} вибирають функції, що використовують евклідову метрику

$$d = \sum_k \left(r_{ik} - r_{jk}\right)^2,$$

де r_{ik}, r_{jk} – координати i-го й j-го нейронів.

До найбільш поширених потенційних функцій відносяться:

а) функція Гаусса

$$f_{\text{gauss}}(d,\sigma) = e^{-\frac{a^2}{2\sigma^2}},$$

де σ^2— дисперсія відхилення;

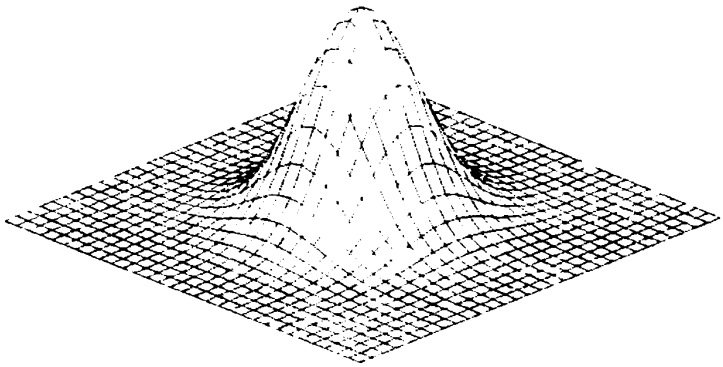


Рис. 10.10. Функція Гаусса

б) функція «Мексиканський капелюх»

$$f_{gauss2}(d,\sigma) = \left(1 - \left(\frac{d}{\sigma}\right)^2\right) e^{-\left(\frac{d}{\sigma}\right)^2},$$

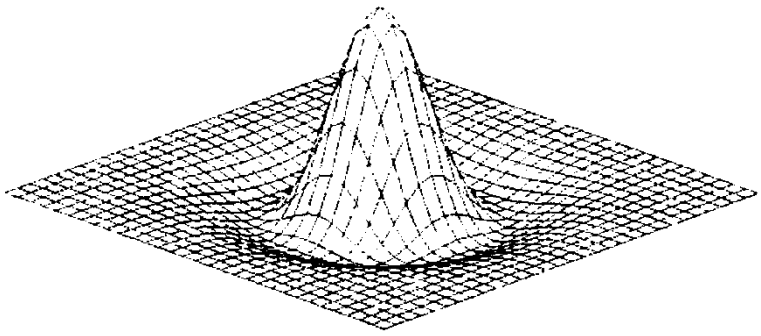


Рис. 10.11. Функція «Мексиканський капелюх»

в) косинусоїдна функція

$$f_{cos}(d,\sigma) = \begin{cases} cos\left(\frac{d\pi}{2\sigma}\right), d < \sigma; \\ 0, \qquad\qquad d \geq \sigma; \end{cases}$$

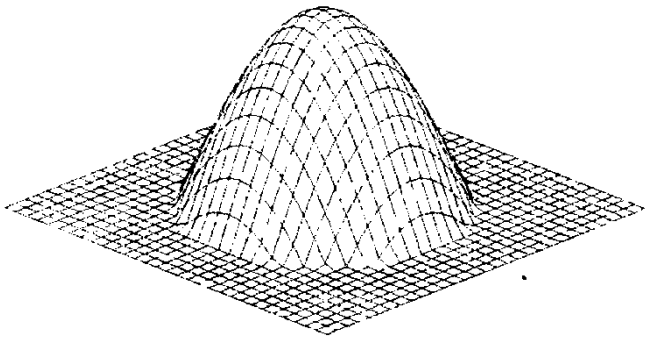


Рис. 10.12. Косинусоїдна функція

г) конусоподібна функція

$$f_{cone}(d,\sigma) = \begin{cases} 1 - \dfrac{d}{\sigma}, & d < \sigma; \\ 0, & d \geq \sigma; \end{cases}$$

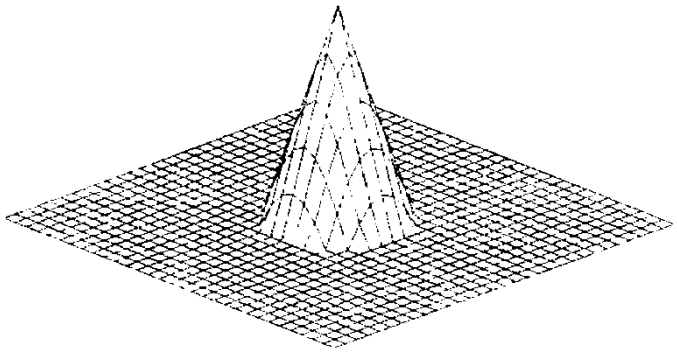


Рис. 10.13. Конусоподібна функція

д) циліндрична функція

$$f_{cylinder}(d,\sigma) = \begin{cases} 1, & \text{при } d < \sigma; \\ 0, & \text{при } d \geq \sigma; \end{cases}$$

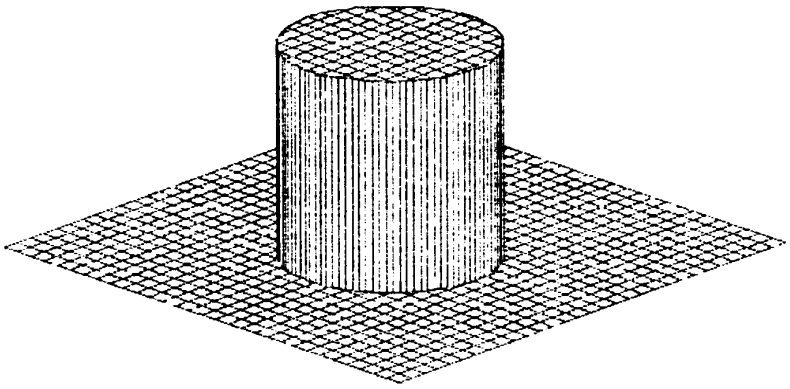


Рис. 10.14. Циліндрична функція

10.4. Побудова мапи Кохонена

Як правило, спочатку будують досить грубу мапу (модель розбиття), поступово уточнюючи її в процесі навчання. Для цього необхідно повільно змінювати не тільки параметр α, але й, наприклад, параметр σ у формулі (10.2). Одним з ефективних способів зміни цих параметрів є такий:

$$\alpha(k) = \alpha(0)\left[\frac{\alpha_{min}}{\alpha(0)}\right]^{\frac{k}{k_{max}}},$$

$$\sigma(k) = \sigma(0)\left[\frac{\sigma_{min}}{\sigma(0)}\right]^{\frac{k}{k_{max}}},$$

де $\alpha(0) \approx 0{,}8;\ \alpha_{min} \ll 1;\ \sigma(0) = 0{,}2;\ \sigma_{min} = 0{,}5$ — параметри, що визначають крутизну функції $f_{ij};\ k_{max}$ — кількість ітерацій, що задаються.

На рис. 10.15 зображено процес побудови мапи Кохонена, що являє собою двовимірну ґратку, утворену шляхом з'єднання сусідніх нейронів, які перебувають у вузлах ґраток. Виходячи з початкових умов і використовуючи алгоритм навчання , мережа в міру збільшення кількості навчальних вхідних образів розвивається й набуває вигляду ґраток. Внизу кожного рисунка зображено кількість образів, на основі яких отримана відповідна мережа [1].

При реалізації мережі Кохонена виникають такі проблеми:

1. Вибір коефіцієнта навчання α.

Цей вибір впливає як на швидкість навчання, так і на стійкість отримуваного розв'язку. Очевидно, що процес навчання прискорюється (швидкість збіжності алгоритму навчання збільшується) при виборі α близьким до одиниці. Однак у цьому випадку подання мережі різних вхідних векторів, що відносяться до одного класу, призведе до змін відповідного вектора вагових коефіцієнтів. Навпаки, при $\alpha \to 0$ швидкість навчання буде повільною, однак вектор вагових коефіцієнтів за умови досягнення центра класу при подачі на вхід мережі різних сигналів, що відносяться до одного класу, залишатиметься поблизу цього центра.

Тому одним зі шляхів прискорення процесу навчання при одночасному забезпеченні отримання стійкого розв'язку є вибір змінного α з $\alpha \to 1$ на початкових етапах навчання й $\alpha \to 0$ — на завершальних. На жаль, такий підхід не може бути застосований у тих випадках, коли мережа має безупинно підлаштовувати-ся до поданих їй нових вхідних сигналів.

2. Рандомізація ваг.

Рандомізація ваг шару Кохонена може породити серйозні проблеми під час навчання, оскільки в результаті цієї операції вагові вектори розподіляються рівномірно по поверхні гіперсфери. Як правило, вхідні вектори розподілені нерівномірно й групуються на відносно малій частині поверхні гіперсфери. Тому більшість вагових векторів виявляться настільки віддаленими від будь-якого вхідного вектора, що не будуть активовані й стануть марними. Більш того, активованих нейронів, які залишилися, може виявитися занадто мало, щоб розбити близько розташовані вхідні вектори на кластери.

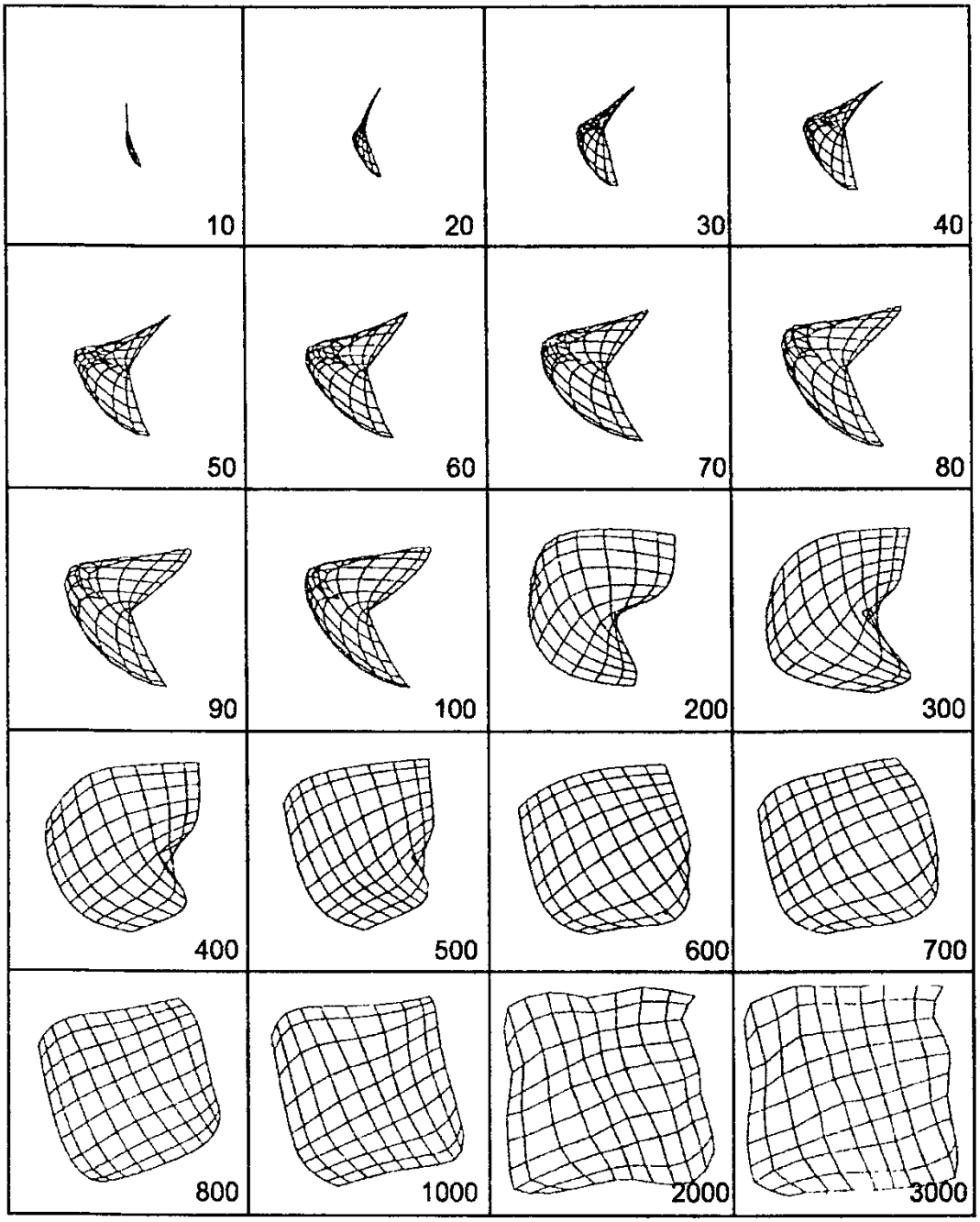
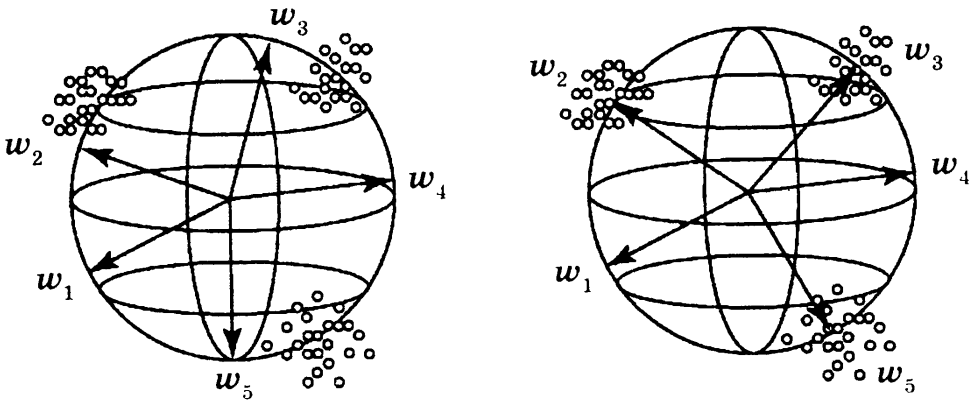


Рис. 10.15. Побудова мапи Кохонена

3. Вибір початкових значень векторів вагових коефіцієнтів і нейронів.

Якщо початкові значення обрані невдало, тобто розташованими далеко від поданих вхідних векторів, то нейрон не виявиться переможцем ні при яких вхідних сигналах, а, отже, не навчиться.

На рис. 10.16 наведено випадок, коли початкове розташування ваг w_1 й w_4 призвело до того, що після навчання мережі всі пропоновані образи класифікують нейрони з вагами w_2, w_3 і w_5. Ваги ж w_1 й w_4 не змінилися й можуть бути вилучені з мережі.



а б

Рис. 10.16.

4. Кількість нейронів у шарі.

Кількість нейронів у шарі Кохонена має відповідати кількості класів вхідних сигналів. Це може бути неприпустимо в тих задачах, коли кількість класів заздалегідь невідома.

5. Класи вхідних сигналів.

Шар Кохонена може формувати тільки класи, що являють собою опуклі області вхідного простору.

ЛІТЕРАТУРА

1. Глибовець М.М., Олецький О.В. Системи штучного інтелекту. — К.: КМ Академія, 2002. — 366 с.
2. Рассел С., Норвиг П. Искусственный интеллект. Современный поход. — М.: Вильямс, 2006. — 1408 с.
3. Люгер Дж. Искусственный интеллект. Стратегии и методы решения сложных проблем. — М.: Вильямс, 2003. — 864 с.
4. Смолин Д.В. Введение в искусственный интеллект: конспект лекций. — М.: ФИЗМАТЛИТ, 2004. — 208 с.
5. Братко И. Алгоритмы искусственного интеллекта на языке Пролог. — М.: Вильямс, 2004. — 640 с.
6. Девятков В.В. Системы искусственного интеллекта. — М.: Изд-во МГТУ им. Баумана, 2001. — 352 с.
7. Субботін С.О. Подання й обробка знань у системах штучного інтелекту та підтримки прийняття рішень. Запоріжжя: ЗНТУ, 2008. — 341 с.
8. Представление и использование знаний / Под ред. Уэно Х., Исидзука М. — М.: Мир, 1989. — 220 с.
9. Искусственный интеллект: Справочник: В 3-х т. — М.: Радио и связь, 1990.
10. Нильсон Н. Принципы искусственного интеллекта. — М.: Радио и связь, 1985. — 376 с.
11. Руденко О. Г., Бодянський Є. В. Штучні нейронні мережі: Навчальний посібник. — Харків: ТОВ "Компанія СМІТ", 2006. — 404 с.
12. Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. — СПб.: Питер, 2018. — 480 с.
13. Жерон О. Прикладное машинное обучение с помощью Scikit-Learn и TensorFlow. — М.: Вильямс, 2018. — 688 с.