Міністерство освіти і науки України
Міжнародний гуманітарний університет
Кафедра комп’ютерних наук

Русу О.П.

ШТУЧНІ НЕЙРОННІ МЕРЕЖІ

Опорний конспект лекцій
для здобувачів вищої освіти,
які навчаються за спеціальностями
галузі «Інформаційні технології»

Одеса 2025

Затверджено Вченою Радою Міжнародного гуманітарного університету.
Протокол № 5 від 20 січня 2025 р.

Русу О.П. Штучні нейронні мережі. Опорний конспект лекцій для здобувачів вищої освіти, які навчаються за спеціальностями галузі «Інформаційні технології» [Електронне видання]. Кафедра комп'ютерних наук Міжнародного гуманітарного університету. Одеса, 2025. 141 с.

Навчальна дисципліна «Штучні нейронні мережі» присвячена вивченню існуючих та перспективних штучних нейронних мереж. При вивченні дисципліни особливу увагу приділяється методам програмної реалізації штучних нейронних мереж, зокрема алгоритмам та математичному апарату, який використовується для їх реалізації. Завдання вивчення дисципліни полягає в отриманні навичок аналізу, вибору та реалізації штучних нейронних мереж з метою впровадження їх в програмному забезпеченні.

ЗМІСТ

Лекція 1. Введення в дисципліну

У сучасному світі прогрес продуктивності фахівця в галузі інформаційних технологій досягається лише тоді, коли частину інтелектуального навантаження беруть на себе комп'ютери. Одним із способів досягти максимального прогресу в цій сфері, є «штучний інтелект», коли комп'ютер спроможний виконувати не тільки однотипні операції по фіксованому алгоритму, але і може сам навчатися. Тому головна задача штучного інтелекту полягає у тому, щоб зробити обчислювальні машини ще кориснішими і перетворити їх на справжніх помічників людини. Але на цьому функція штучного інтелекту не обмежується. Річ у тому, що намагаючись створити машини, що здатні мислити, людина потроху починає розуміти як працює її власний мозок, тобто починає краще зрозуміти сама себе.

Навіщо потрібен штучний інтелект?

Прогрес продуктивності фахівця в галузі інформаційних технологій досягається лише тоді, коли частину інтелектуального навантаження беруть на себе комп'ютери

Створюючи штучний інтелект, людина починає краще розуміти сама себе!

Тому існує два головних напрямку розвитку штучного інтелекту: прагматичний та біонічний. Що той, що інший напрямки наразі знаходяться і стадії розвитку, тому у вас є унікальна можливість зробити свій внесок у розвиток штучного інтелекту і стати затребуваним фахівцем, що ніколи не залишиться без роботи.

Напрями розвитку штучного інтелекту

- Прагматичний
- Біонічний

Прагматичний напрямок розвитку штучного інтелекту не потребує створення копії людського мозку. Тобто у цьому напрямку розумова діяльність людини приймається як якась «чорна скринька», і її не потрібно якось докладно вивчати. Головне щоб результат роботи штучного інтелекту був подібний до результату роботи людини. Тому якщо кінцевий результат функціонування штучної системи хоча б приблизно збігається з результатом діяльності людини, то таку систему вже визнають інтелектуальною незалежно від способів отримання цього результату. За такого підходу не ставиться задача відтворити у штучній комп'ютерній системі структури та процеси, які відбуваються у мозку людини. Головне, щоб кінцевий результат розв'язання задачі був такий же, як і випадку розв'язання її людиною.

Прагматичний напрямок розвитку
штучного інтелекту

Розумова діяльність людини – «чорний ящик»

Цільові галузі
- створення інтелектуальних інструментів (лінгвістичні процесори, спеціалізовані мови, бази знать, прототипи систем)
- розробка методів подання й обробки знань
- інтелектуальне програмування (ігрові програми, системи автоматичного перекладу, розпізнавальні програми)

Наразі можна виділити три цільові галузі застосування прагматичного напрямку. По-перше, це інструментарій, необхідний для створення самого штучного інтелекту, тобто штучний інтелект заради штучного інтелекту. Без цього не можна обійтись, оскільки системи з кожним роком стають усе складнішими, тому створення їх без відповідних інструментів займає багато часу. До цього напрямку відносяться мови для систем штучного інтелекту, методи автоматичного синтезу програм, лінгвістичні процесори, системи аналізу й синтезу мови, бази знань, оболонки та прототипи систем, системи когнітивної графіки та багато інших.

До другого напрямку відноситься створення методів подання й обробки знань. Цей напрямок, фактично є основою штучного інтелекту – його головною теоретичною базою, як мінімум на даному етапі розвитку цієї галузі.

А от третій напрямок розвитку має конкретну прикладну задачу – створення програм або пристроїв, що мають певний рівень інтелекту. Це можуть бути ігрові програми, системи машинного перекладу, системи автоматичного реферування, програми, що спроможні автоматично генерувати тексту або створювати зображення, програми розпізнавання образів та багато інших.

Прагматичний напрямок розвитку
штучного інтелекту

Пристрої та системи не схожі на людину

Основні задачі
- Розпізнавання образів
- Пошук оптимальних варіантів
- Обробка великої кількості інформації

Головною особливістю прагматичного напрямку є те, що кінцеві інтелектуальні застосунки зовсім не схожі на людину. Це можуть бути якісь автомати, наприклад автомат для продажу квитків, або якісь інструменти, наприклад, автоматизована штукатурна станція. Та частіше за все кінцеві застосунки – це програми, які взагалі не мають якоїсь конкретної форми.

Частіше за все інтелектуальні програми, що відносяться до прагматичного напрямку розвитку штучного інтелекту, вирішують задачі, пов’язані із розпізнаванням образів, пошуком оптимальних варіантів, обробкою великих масивів інформації та інших подібних задач. Задач, на які людина завжди витрачає багато свого часу.

Іншим напрямком розвитку штучного інтелекту є біонічний. Біонічний напрямок базується на припущенні про те, що якщо в штучній системі відтворити структури й процеси людського мозку, то й результати розв’язку задач такою системою будуть подібні до результатів, які отримує людина. У біонічних системах виділяють три головні напрямки: нейробіонічний, структурно-еврістичний та гомеостатичний.

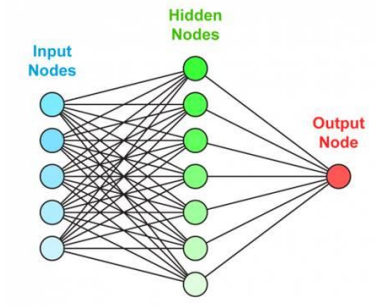
Біонічний напрямок розвитку
штучного інтелекту

Відтворення структур і процесів людського мозку

Основні підходи
- Нейробіонічний
- Структурно-евристичний
- Гомеостатичний

Системи, побудовані за нейробіонічними принципами, імітують елементи нервових систем живих істот і здатні відтворювати деякі інтелектуальні функції, у тому числі і розпізнавання сигналів, що генерує мозок людини. Цей принцип лежить в основі функціонування біонічних протезів, які дозволяють керувати штучною механічною системою так само, як і природним органом. Найбільш відомим прикладом використання елементів, побудованих за цим підходом, є нейронні мережі, які можуть використовуватися практично у всіх сферах діяльності людини. Наразі нейронні системи активно використовуються в медицині, економіці, зв'язку, охоронних та воєнних системах, а також у системах обробки інформації.

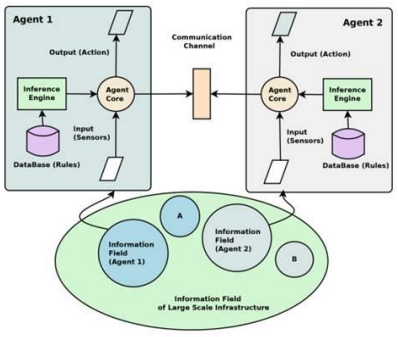
Нейробіонічні системи



- Економіка
- Медицина
- Зв'язок
- Охоронні системи систем
- Обробка інформації

Другим напрямком розвитку біонічних інтелектуальних сисем є системи, побудовані на основі структурно-евристичного підходу. В основі цього напрямку лежать знання про поведінку одного або групи об'єктів. Під час спостереження за цими об'єктами намагаються відтворити ті структури, які б могли забезпечити таку ж саму реакцію на зовнішні події як і природний об'єкт. Прикладом систем побудованих за даними принципом є мультиагентні системи, які активно використовуються в системах оборони, в онлайн-торгівлі, транспорті а також при моделюванні соціальних структур.

Структурно-евристичні системи



- Онлайн-торгівля
- Ліквідація надзвичайних ситуацій
- Моделювання соціальних структур
- Системи оборони
- Транспорт
- Логістика
- Графіка
- Геоінформаційні системи

В основі систем, побудованих за гомеостатичними принципами, лежить відтворення реальності у вигляді популяції віртуальних організмів, що постійно еволюціонує, тобто її

елементи народжуються, ворогують, взаємодіють і нарешті помирають. Результатом подібного функціонування є створення певної рівноваги (стійкості) усієї системи в умовах впливів середовища, яке постійно змінюється. Прикладом подібного рішення є генетичні алгоритми, які активно використовуються для рішення задач, що потребують пошуку найбільш оптимального рішення серед великої кількості можливих варіантів. Прикладом практичного використання генетичних алгоритмів є програми для формування розкладів, а також логістичні системи, у яких генетичні алгоритми використовуються для пошуку найбільш оптимальних маршрутів.

Отже ви вже зрозуміли, що області практичного застосування штучного інтелекту практично безмежні і дуже складно знайти сферу, де його використання неможливе. До недавнього часу розвиток штучного інтелекту стримувався відсутністю необхідної апаратної основи, зокрема потужних комп'ютерів та мікроконтролерів. Проте зараз вже доволі велика кількість виробників електронних компонентів виготовляють елементну базу, на якій створення інтелектуальних систем вже не визиває особливих технічних проблем. Отже у вас є реальний шанс стати частиною цього процесу і, розібравшись у всіх тонкощах створення інтелектуальних програм, отримати цікаву та високооплачувану роботу.

Поняття штучного інтелекту

Термін інтелект (англ. intelligence) походить від латинського intellectus, що означає розум або розумові здібності людини. У рамках дисципліни «Методи та системи штучного інтелекту» під штучним інтелектом ми будемо розуміти здатність мозку вирішувати різні інтелектуальні задачі – задачі які не мають якогось конкретного алгоритму вирішення і єдиної відповіді. Людина такі задачі вирішує на основі отриманих знань та власного досвіду, тому результат вирішення однієї й тої ж задачі у однієї й тої ж людини може бути різним. Та ж сама властивість притаманна і штучному інтелекту (Artificial Intelligence – AI) – вони також при рішенні однієї й тої самої задачі у різний час можуть отримувати різні рішення. Тому потрібно розуміти головне. На відміну від автоматизованих систем, поведінка яких завжди детермінована, інтелектуальні системи важко прогнозувати. Але тільки вони можуть вирішувати задачі, які не можна вирішити за допомогою автоматики.

Штучний інтелект

- **Інтелект** (англ. Intelligence, лат. Intellectus) – розум або розумові здібності людини
- **Інтелект** (у рамках дисципліни) – здатність вирішувати інтелектуальні задачі
- **Штучний інтелект** (англ. Artificial Intelligence – AI) – властивість автоматичних систем брати на себе окремі інтелектуальні функції людини

Зверніть увагу, що у визначенні поняття інтелекту використовується термін «знання», під яким мається на увазі не лише інформація, яка надходить через первинні датчики. Ця інформація надзвичайно важлива, але для інтелектуальної діяльності її недостатньо. Річ у тому, що об'єктам нашого навколишнього середовища притаманна властивість не лише впливати на органи чуття, але й перебувати один із одним у певних відносинах. Тому щоб здійснювати в навколишньому середовищі, що постійно змінюється, інтелектуальну діяльність, або хоча б просто існувати, необхідно мати спершу створити модель навколишнього світу.

Тому для того, щоб практично використовувати інтелектуальні системи, в них необхідно спочатку створити інформаційну модель навколишнього середовища, у якій реальні об'єкти, їх властивості й відносини між ними повинні не лише відображатися і запам'ятовуватися, але ще і «цілеспрямовано перетворюватися». Тому інтелектуальні системи потрібно часто не тільки створювати, а ще й навчати, бо їх поведінка формується саме «в процесі навчання на досвіді і адаптації до різноманітних обставин».

Знання

Знання – не тільки інформація, яка надходить до через органи чуття (або датчики)

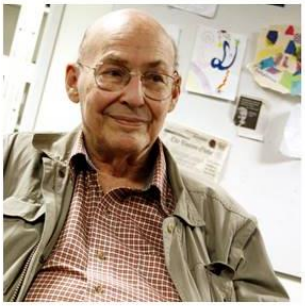
Знання – модель цього світу

Модель зовнішнього середовища формується в процесі навчання на досвіді і адаптації до різноманітних обставин

Штучний інтелект не має якогось сталого конкретного визначення. Кожен фахівець, що працює у цій галузі, напевно, спроможний сформулювати власне трактування цього феномену. І у цьому немає нічого дивного, оскільки тут є певна специфіка – вирішення нечітких задач приводить і до появи нечітких визначень.

Найбільш відомими є визначення предмету теорії штучного інтелекту було дане видатним дослідником у цій галузі Марвіном Мінські. Він казав, що «штучний інтелект є дисципліною, 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, у тому числі і виконання простих арифметичних операцій. Тому до нього дуже швидко додали поправку: «сюди не входять задачі, для яких відома процедура їх вирішення». Проте, навіть з цією поправкою таке визначення також важко вважати задовільним.

Штучний інтелект



Штучний інтелект є дисципліною, що вивчає можливість створення програм для вирішення задач, які при вирішенні їх людиною потребують певних інтелектуальних зусиль. Сюди не входять задачі, для яких відома процедура їх вирішення.

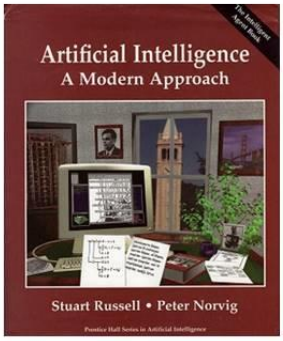
Марвін Лі Мінскі

Відомі світові дослідники в області штучного інтелекту Стюарт Рассел та Пітер Норвіг, що написали найбільш відому книжку «Штучний інтелект: сучасний підхід» (Artificial Intelligence: A Modern Approach), наводять класифікацію означень штучного інтелекту у табличній формі. Відповідно до їх визначення головною ознакою систем із штучним інтелектом є можливість прийняття рішень та виконання дій або раціонально, або так, як це робить людина. Зверніть увагу, що людина не завжди діє раціонально. Тому не слід дивуватись тому, що система під керуванням штучного інтелекту також може прийняти рішення, яке буде далеко від оптимального.

Штучний інтелект

Пітер Норвиг

Стюарт Расел



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 штучний інтелект, не існує. Зазвичай, ці визначення зводяться до наступних:

1. Штучний інтелект вивчає методи розв’язання задач, які потребують людського розуміння. Тут мова іде про те, щоб навчити штучний інтелект розв’язувати тести інтелекту. Це передбачає розвиток способів розв’язання задач за аналогією, методів дедукції та індукції, накопичення базових знань і вміння їх використовувати.
2. 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.
3. Штучний інтелект займається моделюванням людської вищої нервової діяльності.
4. 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.

Чим займається штучний інтелект?

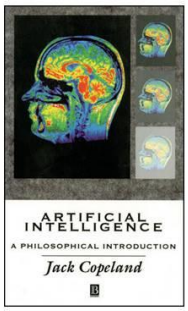
- Штучний інтелект вивчає методи розв'язання задач, які потребують людського розуміння
- Штучний інтелект вивчає методи розв'язання задач, для яких не існує способів розв'язання або вони не коректні
- Штучний інтелект займається моделюванням людської вищої нервової діяльності
- Штучний інтелект – це системи, які можуть оперувати з знаннями, а найголовніше – навчатися

Останнє визначення, із цього переліку що з'явилося у 90-х роках, коли почали бурно розвиватися агентні інтелектуальні системи. Це визначення акцентує увагу на методах і алгоритмах, які дають можливість інтелектуальному агенту виживати в середовищі, яке постійно змінюється. Тому у цьому напрямку особливу увагу приділяють алгоритмам пошуку та прийняття рішень.

Джек Коупленд у праці «Штучний інтелект» відзначає, що незважаючи на наявність численних підходів та визначень як до розуміння цього феномену, так і до створення інтелектуальних інформаційних систем, можна виділити лише два основні підходи щодо розроблення систем із штучним інтелектом:

- 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.;
- 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.

Два основні підходи щодо розроблення систем із штучним інтелектом



- **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)
- **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

Через те, що визначити, що таке штучний інтелект однозначно неможливо, то і виникає проблема із оцінкою рівня інтелектуальності розробленої системи. Тобто ми щось створили, а от як визначити чи дійсно розумним виявився наш витвір? У цьому випадку можна застосувати 2 підходи.

Перший підхід отримав назву «Метод експертних оцінок». При використанні цього методу рішення про ступінь інтелектуальності приймає досить велика група експертів. Вони можуть приймати рішення незалежно один від одного, так і колегіально – обговорюючи результати дослідження. Зрозуміло, що така оцінка завжди буде суб'єктивною, тому для більшої об'єктивності експертну групу необхідно ретельно підбирати як за кількістю, так і за якістю.

Як оцінити рівень інтелекту?

- **Метод експертних оцінок** (суб'єктивна оцінка людини або групи людей)
- **Метод тестування** (на основі групи типових тестів, які також можуть бути недосконалі)

Другий підхід полягає у проходженні інтелектуальною системою певного набору заздалегідь розроблених завдань. Наразі існує доволі велика кількість так званих інтелектуальних тестів. Деякі з них є доволі новими, а деякі вже давно апробовані на практиці і широко використовуються для оцінки рівня розумових здібностей людини, а тому числі в психології та психіатрії.

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 або авторів тестів. 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».

Як оцінити рівень інтелекту?

- Підкресліть слово в нижньому рядку, яке підходить до всіх слів у верхньому рядку:
  ДАЧА, ДАТЧИК, ГОНКА
  хвиля, вода, палиця, птиця, страх, макака
- Вставте число, яке пропущене:
  36 30 24 18 __
- Викресліть зайве слово:
  лев лисиця жираф щука собак

Давайте розглянемо приклад. Нехай серед слів «соловей, чапля, перепілка, стілець та шпак» необхідно 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.

Як оцінити рівень інтелекту?

Видаліть зайве слово:
соловей, чапля, перепілка, стілець, шпак

Правильна відповідь:
Стілець (не птах)

Правильна відповідь:
Шпак (немає букви «л»)

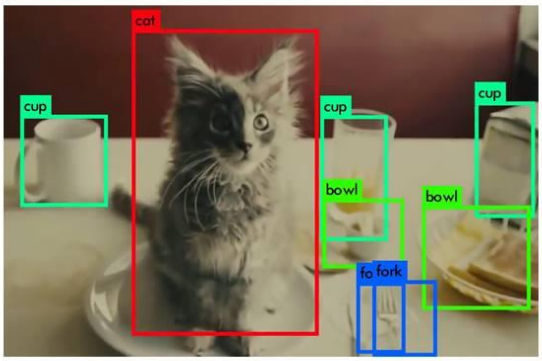
Це положення приймається на аксіоматичному рівні, без доведення. Але те, що є більш важливим для людської практики, зовсім не обов'язково буде так само важливим для

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.

Давайте розглянемо приклади задач, для яких необхідно застосування інтелектуальних можливостей людини або машини. Насправді їх набагато більше, проте найпоширенішими є ті, які ви зараз бачите на екрані. Усі детально ми зараз розглядати не будемо, оскільки зараз необхідно зрозуміти, що таке штучний інтелект та навіщо він взагалі потрібен.

Найбільш поширеною задачею, яку неможливо вирішити без використання штучного інтелекту є задача розпізнавання образів. 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, тобто розбити однин складний об’єкт на складові частини.
Людина робить класифікацію просто. Чоловік, повернувшись додому, відразу ж пізнає свою дитину, собаку та інші об’єкти. 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 зараз дуже інтенсивно розвивається, необхідна для того, щоб навчити штучні інтелектуальні системи вирішувати задачі розпізнавання. Ключовим принципом теорії розпізнавання є те, що будь-який об’єкт у природі є унікальним, і при цьому всі унікальні об`єкти – типізовані.

Розпізнавання образів



- Ідентифікувати об’єкт
- Вирізнити його серед інших
- Розбити об’єкт на складові (провести кластерізацію)

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 теоретично любого об’єкту.

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 образів

- Будь-який об’єкт у природі – унікальний
- Унікальні об`єкти – типізовані

Різноманітних ознак дуже багато, тому системи розпізнавання складних об’єктів поки що знаходяться на початковій стадії розвитку!

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.

Ще однією актуальною задачею, де використовуються системи штучного інтелекту і застосунки, пов’язані із логічним мисленням. Логічне мислення – це перехід від початкових положень до їх наслідків за допомогою формалізованих законів 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.

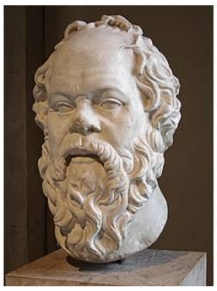
Логічне мислення

Логічне мислення — перехід від початкових положень до їх наслідків за формалізованими законами логіки

Пересічна людина рідко коли може пояснити, за якими алгоритмами вона здійснює логічні побудови

Перш за все, це формальна логіка Аристотеля на основі конструкцій, які отримали назву силогізмів. Розглянемо класичний приклад, який складається із двох положень. Перше положення звучить наступним чином: «Усі люди смертні». Другим положенням є те, що «Сократ – це людина». Зв'завши ці два положення вкупу за допомогою логіки можна зробити на наступний висновок: «Сократ смертний». Отже, 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.

Формальна логіка Аристотеля



- Усі люди смертні
- Сократ – людина

Висновок: Сократ смертний

Виконання формальних вимог є важливим, інакше легко припуститися логічних помилок, подібних до таких: всі студенти університету знають англійську мову; Петренко знає англійську мову, отже, Петренко є студентом університету. Або: Петренко не готувався до іспиту і не здав його; Коваль не готується до іспиту, отже, і Коваль не здасть іспит.

Формальна логіка Аристотеля

- Всі студенти університету знають англійську мову
- Петренко знає англійську мову

Висновок: Петренко навчається в університеті

- Петренко не готувався до іспиту і не здав його
- Коваль не готується до іспиту

Висновок: Коваль не здасть іспит

Аристотелем було запропоновано декілька формальних конструкцій силогізмів, які він вважав достатньо універсальними. У XIX столітті почала розвиватися сучасна математична логіка, яка розглядає аристотелевські силогізми лише як один із часткових випадків.
Основою більшості сучасних систем, призначених для автоматизації логічних побудов, є метод резолюцій Робінсона, який дозволяє доводити різні теореми через пошук протиріч.

Але практична автоматизація логічного мислення зіткнулася з двома серйозними проблемами.

Правило резолюцій

Джон Алан Робінсон

Правило резолюцій — метод доведення теорем через пошук протиріч

Застосування
- логіка висловлювань
- логіка першого порядку

Перша з них отримала завдяки назву, який Річард 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.

Прокляття розмірності

Річард Ернест Беллман

Прокляття розмірності (англ. curse of dimensionality) вказує на різні феномени, які виникають при аналізі та роботі з даними у багатовимірних просторах (часто це сотні або тисячі вимірів)

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. Тобто є вхідні дані, є правильні відповіді на кожну комбінацію вхідних даних, проте немає ніякого способу перетворення цих вхідних даних на відповіді.

Алгоритмічно нерозв'язна задача

Курт Гедель

Теореми Геделя
- Якщо формальна арифметика є несуперечливою, то в ній існує невивідна і неспростовна формула
- Якщо формальна арифметика є несуперечливою, то в ній є невивідною деяка формула, яка змістовно стверджує несуперечливість цієї арифметики

Алгоритмічно нерозв'язною задачею називається задача, що має відповідь **Так** чи **Ні** для кожної комбінації із обмеженого набору вхідних даних, але не має алгоритму отримання цих відповідей

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.

Експертна система

Експертна система — методологія адаптації алгоритму успішних рішень однієї сфери науково-практичної діяльності в іншу

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 у рамках дослідження штучного інтелекту були запропоновані ефективні алгоритми, які дозволяли вирішувати задачі, що мають кількість обчислень, які фізично неможливо виконати навіть на самому продуктивному комп'ютері.

Однією з таких задач є задача комівояджера, яка полягає у тому, що торговому агенту необхідно об'їхати певний перелік міст таким чином, що побувати у кожному місті лише один раз, потім повернутися у точку виїзду, і при цьому довжина шляху повинна бути найкоротша. Алгоритму, що дозволяв би швидко вирішити цю задачу, у природі не існує – цю задачу можна вирішити лише шляхом повного перебору варіантів. Та річ у тому, що кількість можливих варіантів рішення має факторіальну залежність від кількості міст. Сучасний комп'ютер шляхом прямого перебору зможе визначити маршрут, у якому налічується 10 – 11 міст. Для того, що визначити найкоротшій маршрут, що має 12 міст, вже необхідно витратити приблизно один робочий день. А для того, щоб шляхом прямого перебору визначити маршрут, що містить 20 міст, потрібно кілька тисяч років. А людина подібну задачу може вирішити за кілька хвилин.

Задача комівояджера

Кількість міст	Кількість варіантів
2	2
3	6
4	24
5	120
6	720
7	5040
8	40320
9	362880
10	3628800

Кількість міст	Кількість варіантів
11	39916800
12	479001600
13	6227020800
14	87178291200
15	1307674368000
16	20922789888000
17	355687428096000
18	6402373705728000
19	121645100408832000

Таким чином, навіть після такого стислого огляду можна зробити висновки, що штучний інтелект має велику користь для людини. Проте величина його користі така ж сама, як величина його складності. Але у будь якому випадку, галузь штучного інтелекту буде бурхливо розвиватися. Тому, якщо ви зможете освоїти цей напрямок, то ви ніколи не залишитесь без роботи.

Висновки

- Штучним інтелектом вважається властивість автоматичних систем брати на себе окремі функції інтелекту людини
- Існує два напрями розвитку штучного інтелекту
- Штучний інтелект спрямований на вирішення задач, які не мають алгоритму розв'язання або потребують проведення великої кількості обчислень
- **Галузь штучного інтелекту поки ще знаходиться у самому початку свого розвитку** ~~(і це престижна та високооплачувана робота)~~

Лекція 2. Принципи побудови штучних нейронних мереж

Я певен, що ви вже переглянули попередні навчальні матеріали і знаєте, що штучний інтелект має велику користь для людини. Так само як і знаєте те, що рівень складності штучних інтелектуальних систем набагато перевищує рівень складності більшості існуючих технічних засобів.

Але якими б складними не були сучасні інтелектуальні пристрої та системи, їх усе одно створюють люди. А це означає, що їх можна вивчити, повторити та вдосконалити. І саме цим ми й будемо займатися під час вивчення дисципліни «Методи та системи штучного інтелекту». А шлях вивчення штучних інтелектуальних систем краще всього починати із вивчення нейронних мереж, оскільки саме вони свого часу дали поштовх для розвитку штучного інтелекту. Саме цим ми й будемо займатися протягом наступних кількох тижнів. А розпочнемо ми з невеликої історичної екскурсії, яка допоможе нам зрозуміти, чому ці системи називаються «нейронними», звідки вони з'явилися і головне – чому людина досі не створила штучного аналога самого себе.

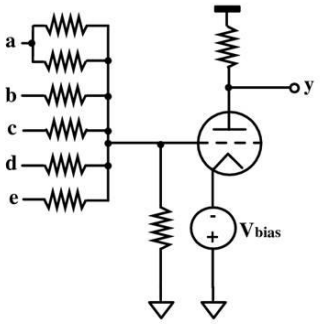
Історія розвитку нейронних мереж

Гіпотезу про можливість машинного моделювання діяльності людського мозку запропонував американський математик та кібернетик Норберт Вінер, якого обґрунтовано вважають одним із «батьків» штучного інтелекту. У 1942 році на одній із конференцій він зробив доповідь про механізми зворотних зв'язків у біології, на якій, зокрема, було висловлене припущення, що діяльність людського мозку можна моделювати штучно. На цій доповіді був присутній Воррен Маккалок, який також був захоплений ідеями штучного моделювання процесів, що протікають у нервових системах, але мав певні сумніви щодо можливості практичної реалізації цієї теорії. Впевнившись, що не тільки він мріє відтворити біологічні розумові процеси в електронних системах, Маккалок вирішив спробувати якось формалізувати електронні системи, які зможуть виконувати інтелектуальні функції. Менше ніж за рік Воррен Маккалок зі своїм колегою Уолтером Пилттсом підготували статтю, у якій вперше був використаний термін «нейронна мережа» та запропонований алгоритм її функціонування. Після цього дуже швидко з'явилося кілька робіт, що стали своєрідним фундаментом для нейронних мереж. Однією з цих робіт була робота Френка Розенблата, у якій був запропонований одношаровий персептрон – одна із найбільш популярних моделей нейронних мереж того часу.

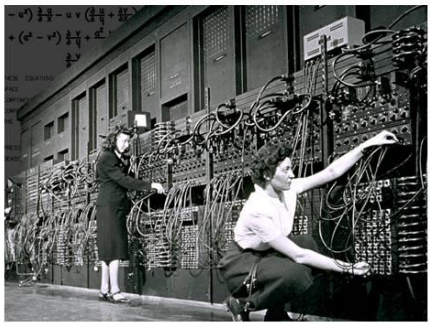
Та до створення першої нейронної мережі ще було далеко, оскільки елементна база для електронних пристроїв того часу була досить обмежена. Спочатку нейронні мережі намагалися створити на аналогових електронних схемах, де усі елементи мережі мали «жорсткий» фізичний зв'язок між собою. Проте дуже швидко стало зрозуміло, що для їх реалізації краще підходить електроніка із «м'якими» зв'язками між елементами, тобто

електроніка, що працює під керівництвом програми. Але на той час електронно-обчислювальні машини тільки починали з'являтися і були дуже дорогі та недосконалі. Тому практична перевірка теорії нейронних мереж була відкладена на певний час.

Початок нейронних мереж



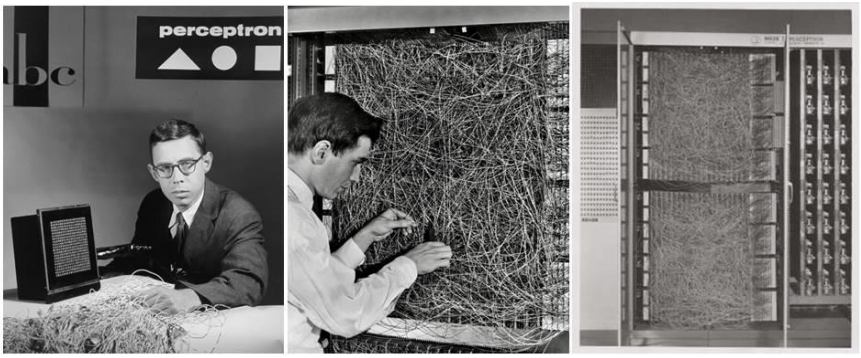
Штучний нейрон та вакуумній лампі



Перша ЕОМ (ENIAC) (1946 р)

Однією із перших практичних реалізацій нейронних мереж став персептрон Френка Розенблата, реалізований на комп'ютері «Марк-1». Цей комп'ютер підключили до чотирьохсот сульфід-кадмієвих фотоелементів, які утворили світлочутливу матрицю із роздільною здатністю 20 на 20 елементів. Зв'язки всередині нейронної мережі налаштовувались за допомогою патч-панелі, а сила зв'язків – за допомогою спеціальних потенціометрів. Така система могла розпізнавати деякі літери, які підносили до «ока» комп'ютера на спеціальних картках, і обіцяла практично безмежні перспективи для подальшого розвитку. На той час людям здавалося, що до реалізації штучного мозку залишається лише один крок – потрібно лише створити нейронну мережу відповідних розмірів.

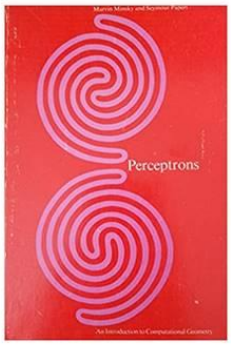
Френк Розенблат з першою реалізацію нейронних мереж
на комп'ютері «Марк-1» (1960 р.)



Проте дуже швидко виявилося, що реальні результати, які показують нейронні мережі, м'яко кажучи, далекі від очікуваних. І виникли дві проблеми. Практично одразу виявилось, що потужності електронно-обчислювальних машин того часу було явно недостатньо для створення навіть невеликої нейронної мережі. Це, звісно, було неприємно, але не критично – потрібен був лише час для того, щоб вони вийшли на необхідний технічний рівень. А от розв'язати другу проблему здавалося принципово неможливо.

Період занепаду нейронних мереж

Марвін Лі Мінскі
1927 – 2016



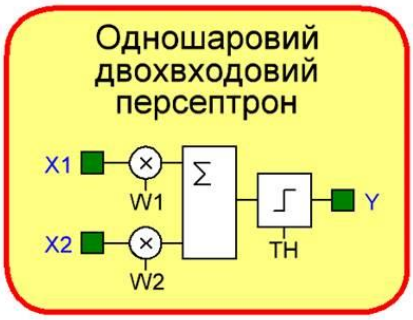
Книга «Персептони»
(Perseptons) (1969)

Основні проблеми нейронних мереж у 1969 – 1981 роках

- Потужність комп'ютерів того часу була недостатня для створення нейронних мереж
- **Не всі функції можуть бути реалізовані за допомогою одношарових персептронів** (М. Мінські, 1969)

Проблема полягала у тому, що нейронні мережі, реалізовані на одношарових персептронах, виявилися нездатними вирішувати деякі прості задачі. Наприклад, двохвходовий одношаровий персептрон можна легко налаштувати для роботи у режимі диз'юнкції (логічне АБО) або кон'юнкції (логічне І), але неможливо налаштувати для роботи у режимі виняткової диз'юнкції – тобто реалізувати на його основі функцію виняткового АБО. Довгий час інженери вважали, що вони просто неспроможні правильно налаштувати подібну систему і вперто перевіряли різні варіанти, але усі їх спроби виявилися марними. І лише у 1969 році Марвін Лі Мінські у своїй книзі «Персептрони» чітко довів, що цю задачу вирішити неможливо.

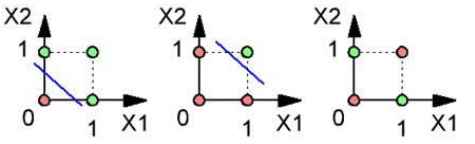
Лінійна сепарабельність (розділяємість)



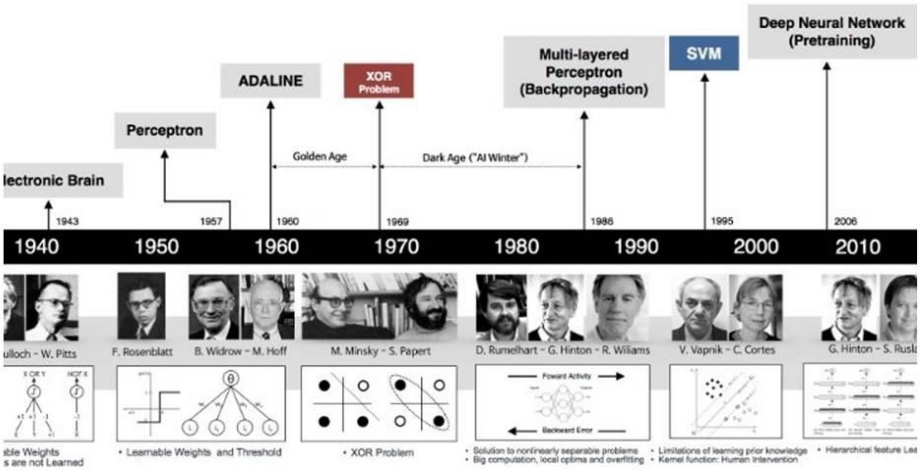
OR		
X1	X2	Y
0	0	0
1	0	1
0	1	1
1	1	1

AND		
X1	X2	Y
0	0	0
1	0	0
0	1	0
1	1	1

XOR		
X1	X2	Y
0	0	0
1	0	1
0	1	1
1	1	0



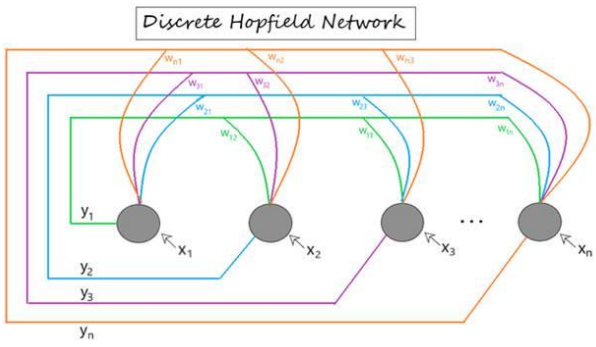
Блиск та суворість аргументації Мінського, а також його престиж породили величезну довіру до книги – її висновки були невразливими. Розчаровані дослідники припинили роботи над створенням штучного інтелекту і перейшли у більш перспективні галузі. Уряди практично усіх країн знизили фінансування досліджень цього напрямку, і штучні нейронні мережі забули майже два десятиліття. Проте кілька найбільш наполегливих вчених, таких як Кохонен, Гроссберг, Андерсон все ж таки не здалися і продовжили дослідження нейронних мереж. У той час вчені, що працювали зі штучним інтелектом, постійно стикалися із браком фінансування, скептицизмом зі сторони колег і труднощами у публікаціях результатів досліджень. Тому дослідження, опубліковані в сімдесятих і на початку вісімдесятих років, розпорошені у різних журналах, більшість із з яких були та залишаються маловідомими.



Цей період часто називають «першою зимою штучного інтелекту». Він закінчився аж у 1982 році, коли в Джон Гопфілд написав статтю, у якій представив нейронну мережу, яка зараз відома як мережа Гопфілда. З'ясувалося, що оцінка Мінського виявилася занадто песимістичною, і задачі, які не можна вирішити за допомогою одношарових персептронів, можна вирішити за допомогою більш складних багатошарових нейронних мереж. Більше того, окрім рекурентних мереж, якими є мережі Гопфілда, на той час вже були розроблені нейронні мережі із зворотним розповсюдженням інформації разом із алгоритмами їх навчання. Крім того, не слід забувати, що обчислювальна потужність комп'ютерів того часу, побудованих вже на напівпровідникових мікропроцесорах, а не на електронних лампах, набагато перевищувала потужність електронно-обчислювальних машин 60-х років, а отже іще одне технічне обмеження було усунете.

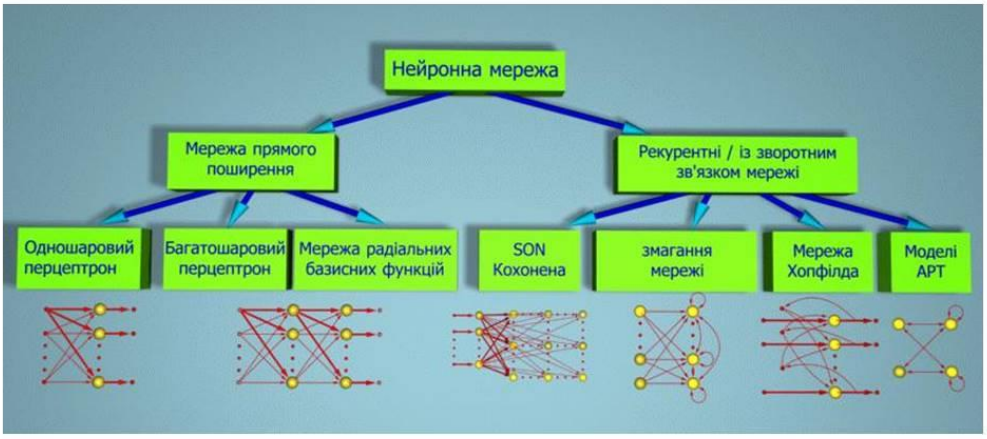
Нейронна мережа Гопфілда

Джон Джозеф Гопфілд
1933 рн



Усе призвело до того, що у 1982 році Японія на одній із американо-японських конференції оголосила про поновлення досліджень в області нейронних мереж. Сполучені Штати Америки, побоюючись, що Японія значно випередить їх у цій галузі була змушена також поновити фінансування досліджень, і з цього часу нейронні мережі знову почали бурхливо розвиватися. Однак цей період вже значно відрізнявся від періоду 60-х років. У той час дослідження нейронних мереж проводились на рівні інтуїції без якого системного підходи. А у 80-х роках вже була готова доволі ґрунтовна теоретична база, яка дозволяла не тільки ефективно обходити обмеження, виявлені Мінським, а й вирішувати нові задачі. Отже Мінські, з одного боку, штучно загальмував розвиток нейронних мереж, опублікувавши у 1969 році книжку із хибними висновками, а з другого – дав можливість цій області дозріти теоретично, що значно зменшило кількість помилок під час створення у 1980-х роках нейронних мереж нового покоління.

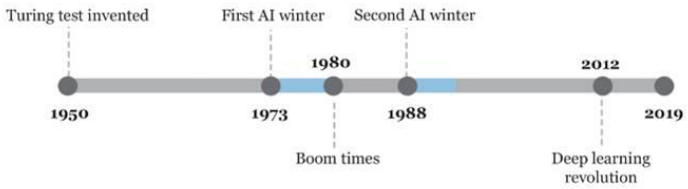
Типи нейронних мереж



Наступний період затишшя в історії розвитку нейронних мереж розпочався у 1988 році, коли стало зрозуміло, що штучний інтелект та нейронні мережі знову виявилися набагато складнішими, ніж це вважалося того часу. «Друга зима штучного інтелекту» розпочалася у 1984 році, коли Джон Маккарті виступив із критикою експертних систем – що на той час були пріоритетним напрямком розвитку нейронних мереж.

Історія розвитку нейронних мереж та штучного інтелекту

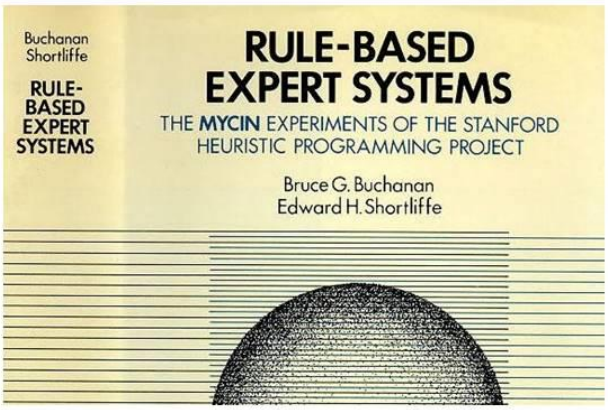
Джон Маккарті
1927 – 2011



У цій роботі основні акценти Маккарті зробив на те, що експертним системам того часу не вистачало здорового глузду та знань про власні обмеження. Зокрема, він розповів про експертну систему MYCIN, створену для допомоги лікарям. У якості прикладу її недосконалості Маккарті привів реакцію системи на випадок, коли у кишечнику пацієнта виявлено холерний вібріон. Програма прописала пацієнту двотижневий курс тетрацикліну, який, можливо, і вилікував би його, проте пацієнт до кінця лікування, скоріш за все, не дожив би. У його звіті також було багато зауважень відносно складності завдань, які пов’язані із штучним інтелектом та нейронними мережами, причиною яких є неймовірно велика кількість варіантів їх рішення.

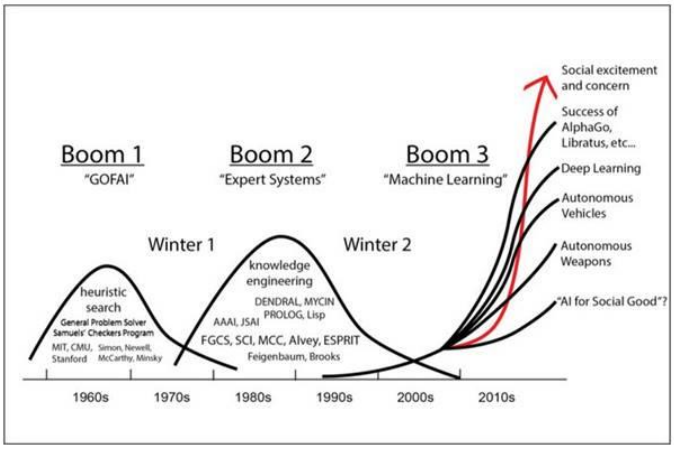
Подібна ситуація склалася і в інших галузях застосування нейронних мереж – майже усюди первинний ентузіазм та неймовірні очікування від їх застосування змінився на розчарування. Директор Агентства перспективних оборонних дослідницьких проектів Управління інформаційних наук і технологій США (DARPA ISTO) Шварц якось написав, що дослідження в галузі штучного інтелекту завжди мають «…обмежений успіх у певних галузях, за яким негайно йде невдача у досягненні ширшої мети, на яку ці початкові успіхи, напевно, натякають…».

Експертна система MYCIN для допомоги докторам



Усе це призвело до поступового охолодження цікавості до питань нейронних мереж і, відповідно, до штучного інтелекту, а це, в свою чергу, призвело до зменшення фінансування цієї галузі. Мінімальний рівень наукових статей, пов'язаних із розвитком штучного інтелекту був зафіксований у 1995 році. Проте роботи у цьому напрямку повністю не припинялися і нейронні мережі постійно вдосконалювалися і поступово досягли рівня, за якого їх стало можливо використовувати на практиці. Відбулося це завдяки реалізації технології глибокого навчання, яка була розроблена у 2006 році.

Етапи розвитку нейронних мереж



На сьогоднішній день нейронні мережі знову знаходяться у стані бурхливого розвитку. Повний перелік галузей, де можна використовувати нейронні мережі, є доволі великим, тому можна привести лише основні напрямки їх використання. Наразі немає такого напрямку, який, з одного боку, потребує інтелектуального втручання людини, а з другого, не дозволяє використовувати нейронні мережі. Іншими словами, нейронні мережі можна використовувати всюди, де необхідне інтелектуальне керування. Проте не слід забувати, що, не заважаючи на приголомшливі досягнення, нейронні мережі ще не досконалі, і тому до їх практичного використання поки що слід ставитися доволі обережно.

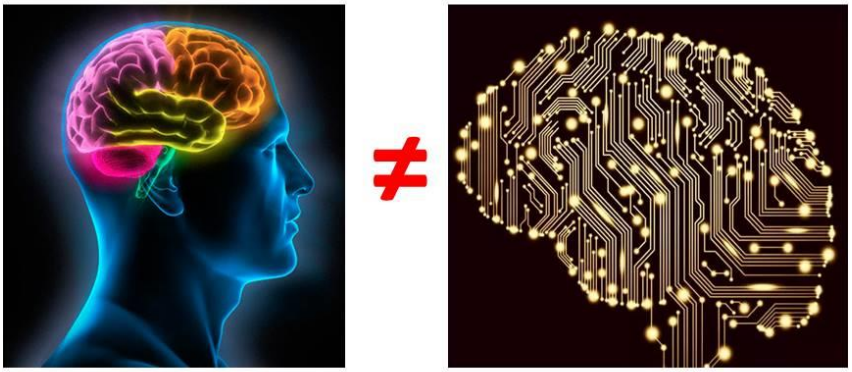
Сучасне використання нейронних мереж

- Розпізнавання образів і класифікація
- Прийняття рішень та керування
- Кластерний аналіз
- Прогнозування
- Апроксимація
- Стиснення даних та асоціативна пам'ять

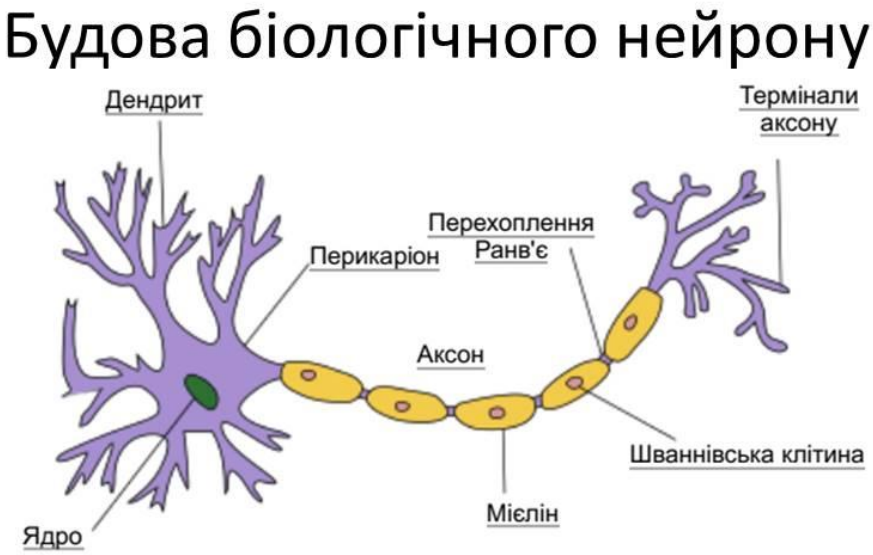
Принцип роботи біологічних нейронних систем

Як ви вже зрозуміли, нейронні мережі є копією реальних біологічних систем, у тому числі і копією головного мозку людини, тому перед тим як перейти до вивчення штучних нейронних мереж, доцільно спочатку стисло розглянути як працюють їх біологічні прототипи. Але тут слід обов'язково зазначити, що навіть зараз, попри усі досягнення нейробіології, наші знання про роботу мозку настільки обмежені, що існує дуже мало якихось орієнтирів, на які можуть спиратися розробники штучних нейронних систем. Тому людям, що працюють у цій галузі, у пошуках структур, здатних виконувати необхідні функції, доволі часто доводиться виходити за межі сучасних біологічних знань, відмовлятися від біологічної правдоподібності і навіть створювати системи, які не можуть існувати у живій природі. У цьому випадку мозок стає простою метафорою, і процеси, що протікають у штучних системах, не мають нічого спільного із процесами, що протікають у системах, що мають біологічну природу.

Біологічна vs. Штучна мережа



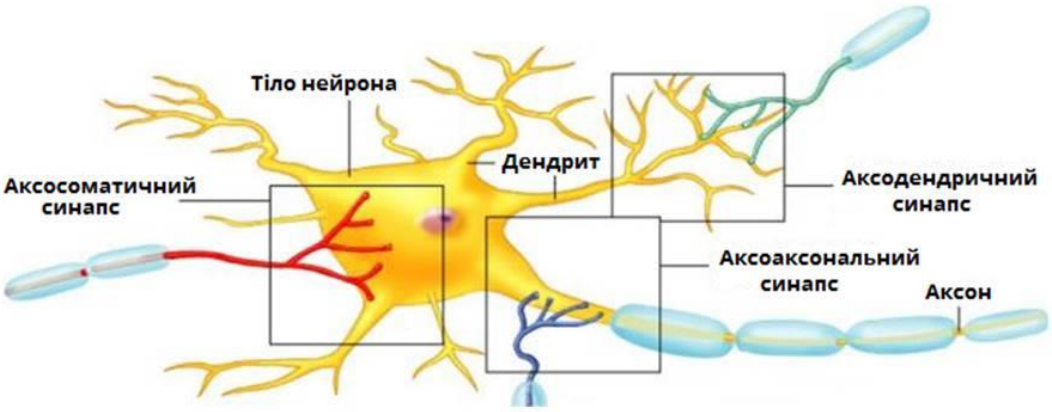
Основним елементом нервової системи живих істот є нейрон. Нейрон є електрично збудливою клітиною, що обробляє та передає інформацію у вигляді електричного або хімічного сигналу. Типовий нейрон складається з тіла клітини, дендритів та аксону. Дендрити є відростками, що виходять з тіла клітини. Вони можуть тягнутися на сотні мікрон та багаторазово розгалужуватися, утворюючи складне «дендритне дерево». Аксон побудований із особливого клітинного волокна, що виходить з тіла і простягається на відстань приблизно 1 м, а у деяких тварин навіть і більше. З тіла нейрону зазвичай виходить декілька дендритів, але не більше одного аксону, хоча останній може розгалужуватись сотні разів.



Зв’язок між нейронами відбувається за допомогою синапсів – спеціалізованих контактів між двома нейронами, або між нейроном та клітинами іншого типу. Нервова система людини, містить близько 86 мільярдів ($86 \cdot 10^9$) нейронів, які мають приблизно один більярд (10^{15}) зв’язків. Кожен нейрон має багато властивостей, притаманним іншим тканинам тіла, але його унікальною здатністю є прийом, обробка та передача електрохімічних сигналів по нервових шляхам, які власне і утворюють комунікаційну систему мозку.

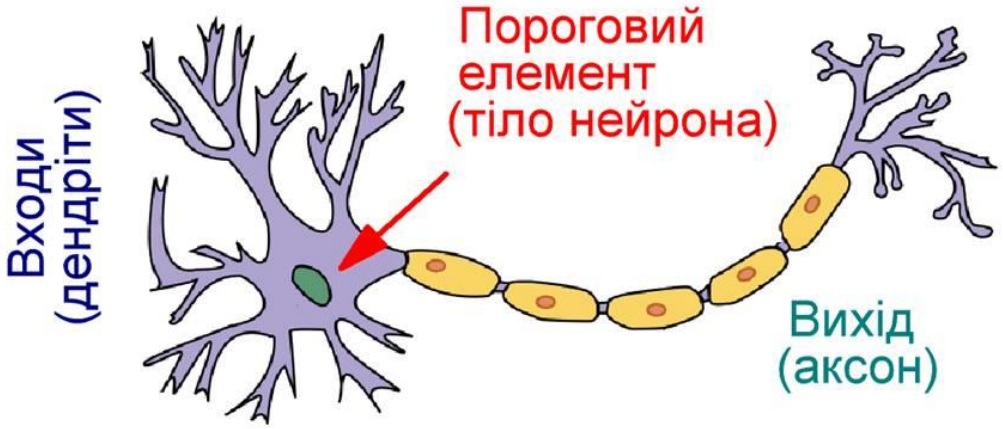
Дендрити йдуть від тіла нервової клітини до інших нейронів, де вони приймають сигнали з точок з'єднання, які називають синапсами. Прийняті синапсами сигнали від інших клітин підводяться до тіла нейрона, у якому вони підсумовуються. Особливістю цієї операції є те, що одні сигнали прагнуть перевести нейрон у збуджений стан, а інші навпаки – намагаються перешкодити його збудженню. Коли сума вхідних сигналів стає більшою за певний поріг, у тілі нейрона починається певна хімічна реакція і він переходить у збуджений стан. У цьому стані на його аксоні з’являється певний електричний потенціал, який передається іншим нейронам або клітинам.

З’єднання нейронів у мережу



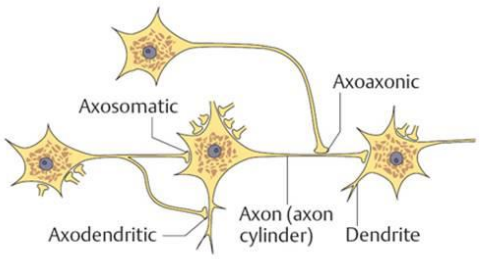
Таким чином, біологічний нейрон можна представити як певний пороговий елемент, функцію якого виконує безпосередньо сама клітина нейрона. Цей елемент може мати декілька входів, функцію яких виконують дендрити, та один вихід, утворений аксоном. Ця основна функціональна схема дуже спрощена – реальні нейрони можуть мати і трошки іншу будову, і трішки інший принцип роботи. Проте більшість нейронів, наприклад, у головному мозку людини, мають саме таку будову. Тому для розуміння роботи штучних нейронних мереж такої моделі цілком достатньо.

Будова біологічного нейрону



Кожен із нейронів головного мозку зв’язується із іншими нейронами. У біологічних системах один нейрон може бути зв’язаним із кількома тисячами інших нейронів. Зверніть увагу на те, що синапси мають певний електричний опір і певні частотні характеристики. Тому сигнал, що передається через синапс, надходить ослабленим на певну величину і можливо із певною затримкою у часі. Це є дуже важливим моментом, оскільки саме у характеристиках синапсів зберігається інформація та пам’ять людини. І процес навчання штучної нейронної мережі, так само, як і процес навчання її біологічного аналога, зводиться до встановлення такого коефіцієнту передачі синапсів, за якого забезпечується мінімальний рівень помилок.

Види синапсів



Синапси мають певний електричний опір, тому сигнал, що передається по аксону до кожного дендріта надходить ослабленим на певну величину, яка залежить від стану синапса

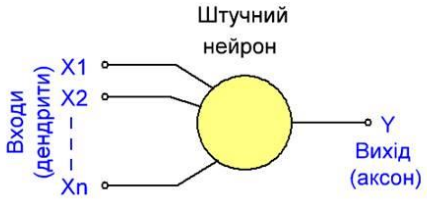
Принципи побудови штучних нейронних систем

Штучні нейронні системи, так само як і біологічні, складаються із окремих нейронів, зв'язаних між собою електрично або програмно. Штучні нейрони, зазвичай, позначаються на схемах у вигляді кружечків. З лівої сторони кружечка, зазвичай, розташовують входи нейрона, які є аналогами дендритів. Кількість входів штучного нейрона, так само, як кількість дендритів, теоретично необмежена.

З правої сторони кружечка, зазвичай, розташовують єдиний вихід, який є аналогом аксона. Вихідний сигнал штучного нейрона, теоретично, може приймати любі значення, але розрізнюють лише два стани: нормальний та збуджений. У нормальному стані рівень вихідного сигналу нейрона близький до рівня, що відповідає логічному нулю, а у збудженому стані його вихідний сигнал зростає до рівня, що наближається до рівня логічної одиниці.

Ще раз зверніть увагу, що рівні сигналів на входах і виході штучного нейрона можуть приймати будь яке значення. Тобто, не зважаючи на те, що система складається із дискретних компонентів і описується дискретними рівняннями, вона фактично є аналоговою системою, що працює із безперервними сигналами.

Штучний нейрон



Штучний нейрон має

- Кілька входів (необмежена кількість) (аналог дендритів)
- Один вихід (аналог аксона)
- Пороговий елемент (аналог тіла біологічного нейрона)

Може знаходитися у двох станах

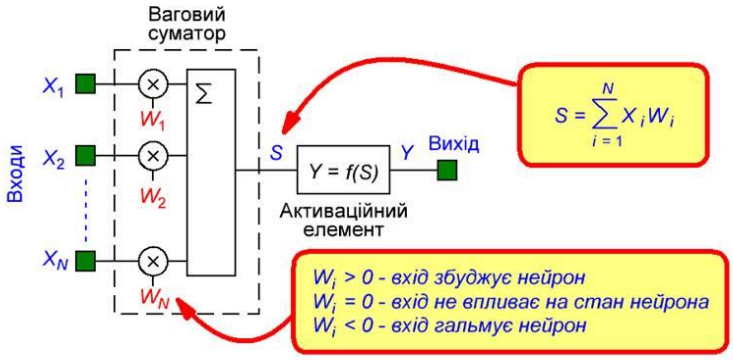
- **Незбудженому** (на виході рівень сигналу близький до рівня логічного нуля)
- **Збудженому** (на виході рівень сигналу близький до рівня логічної одиниці)

Сигнали на входах на виході можуть приймати будь яке значення!!!

Кожен штучний нейрон складається із двох основних елементів – вагового суматора та активаційного елемента. Ваговий суматор арифметично складає рівень сигналів на входах штучного нейрона і передає обчислену суму на вхід активаційного елемента. Активаційний елемент аналізує сигнал на своєму вході і формує вихідний сигнал, відповідно до правила, яке називається активаційною функцією.

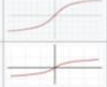
Зверніть увагу, що вхідні сигнали надходять до суматора не безпосередньо, а через помножувачі, які обчислюють добуток вхідного сигналу на певний статичний коефіцієнт W. Ці помножувачі є еквівалентами синапсів у біологічних системах, і вони можуть суттєво змінити вплив даного входу на стан нейрону. Якщо ваговий коефіцієнт більше нуля, то сигнал на відповідному вході буде збуджувати нейрон. Причому, чим більше абсолютне значення вагового коефіцієнту, тим суттєвіший вплив цього входу на стан нейрона. Якщо ж ваговий коефіцієнт менше нуля, то цей вхід навпаки – буде гальмувати нейрон, і чим більше абсолютне значення буде мати ваговий коефіцієнт, тим складніше буде збудити нейрон. Ну а я якщо цей коефіцієнт буде дорівнювати нулю, то це означає, що сигнал на відповідному вході ніяк не буде впливати на стан нейрона.

Будова штучного нейрона

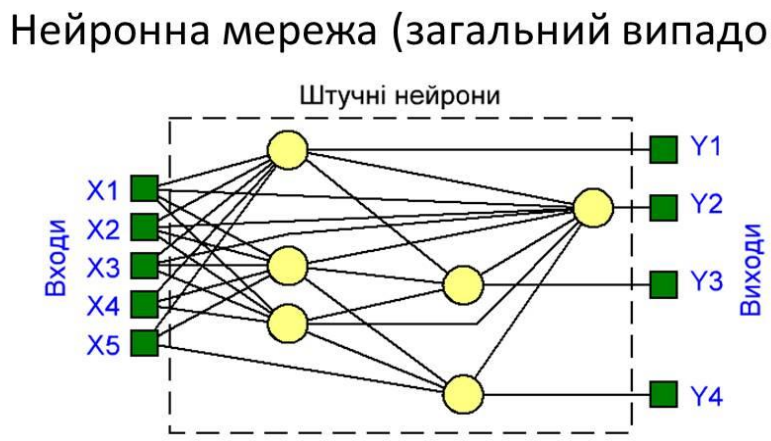


Функція активації, або передавальна функція (Activation Function, Excitation Function, Squashing Function, Transfer Function) штучного нейрона встановлює залежність вихідного сигналу активаційного елемента від сигналу на його вході. Теоретично, сигнал на виході нейрона може приймати будь яке значення, проте частіше за все використовують такі функції, які забезпечують рівень вихідного сигналу або в діапазоні від 0 до 1, або в діапазоні від –1 до 1. Вибір функції активації залежить від конкретного призначення нейронної мережі. Частіше за все використовують сигмоїдні функції активації, проте це не обов'язково. У деяких випадках, наприклад, коли сигнал на виході нейрона повинен мати уніполярні значення, слід використовувати функції на основі гіперболічного тангенса, у деяких випадках – функції на основі псевдовипрямлених значень вхідного сигналу. А от для реалізації деяких простих нейронних мереж інколи достатньо використовувати нейрони із простою пороговою функцією активації, графік якої нагадує сходинку.

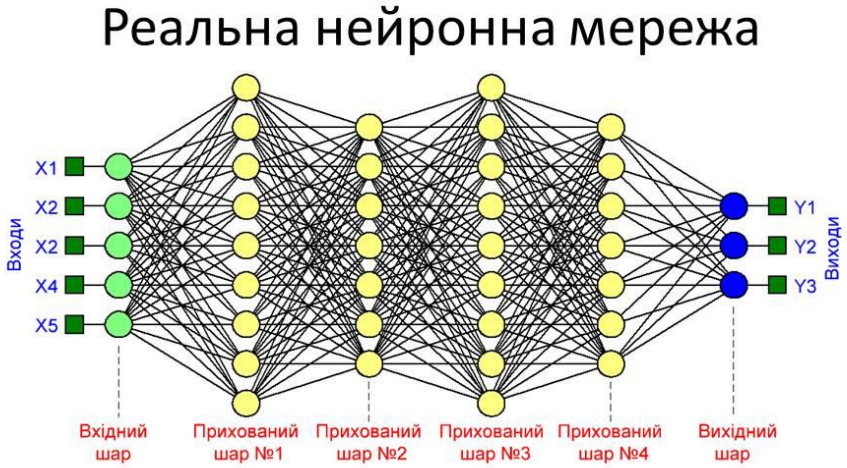
Деякі функції активації

Назва	Графік	Рівняння	Похідна (по x)	Область	Порядок гладкості
Тотожна		$f(x) = x$	$f'(x) = 1$	$(-\infty, \infty)$	C^∞
Двійковий крок		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geqslant 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$	$\{0, 1\}$	C^{-1}
Логістична (Сігмоїда або М'який крок)		$f(x) = \sigma(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1-f(x))$	$(0, 1)$	C^∞
TanH		$f(x) = \tanh(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$	$f'(x) = 1 - f(x)^2$	$(-1, 1)$	C^∞
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2+1}$	$\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$	C^∞
Softsign		$f(x) = \frac{x}{1+\lvert x\rvert}$	$f'(x) = \frac{1}{(1+\lvert x\rvert)^2}$	$(-1, 1)$	C^1
Випрямлена лінійна (Rectified linear unit, ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geqslant 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geqslant 0 \end{cases}$	$[0, \infty)$	C^0

Штучні нейрони з’єднуються між собою, утворюючи мережу. У загальному випадку, штучна нейронна мережа може мати будь-яку кількість входів та будь-яку кількість виходів. Крім того, самі нейрони всередині мережі можуть з’єднуватися між собою як завгодно, утворюючи безкінечно велику кількість можливих комбінацій. Зрозуміло, що подібні системи – системи із довільною конфігурацією – дуже важко як аналізувати, так і реалізовувати. Тому на практиці люди, що працюють у цій галузі, зазвичай, штучно обмежують себе і використовують лише певну кількість стандартних нейронних мереж.

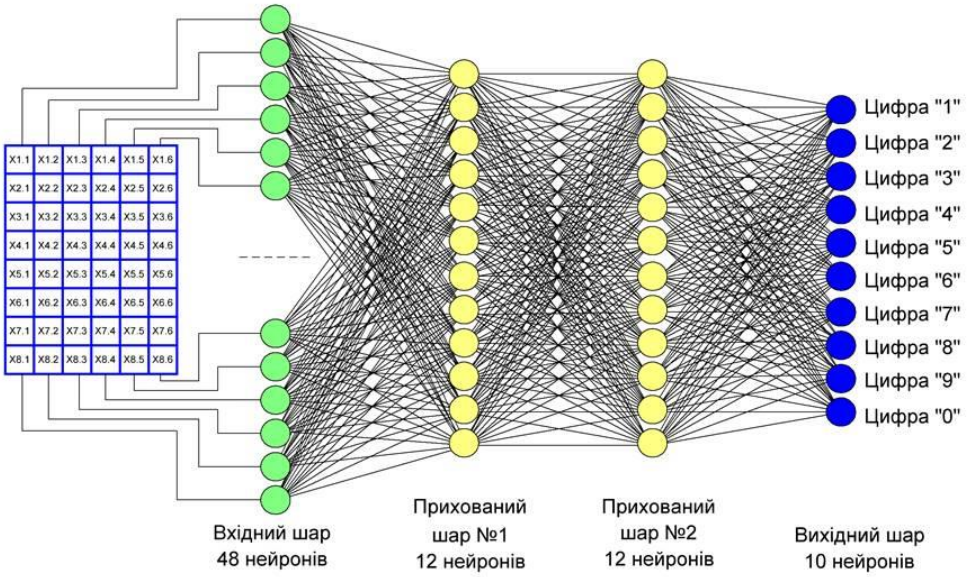


У цьому випадку, усі нейрони штучної нейронної мережі розділяються на окремі частини, які називають шарами. Розрізняють вхідний шар, який з’єднує мережу із джерелами вхідних сигналів, та вихідний шар, який з’єднує систему із приймачами результатів обчислень.
Нейрони вхідного шару, зазвичай, не виконують якогось певного логічного навантаження і у доволі великій кількості теорій нейронних мереж виконують узгоджувальну функцію, що полягає лише у передачі вхідного сигналу до подальших нейронів без якихось змін. У реальних системах вони також лише приймають та зберігають сигнали із зовнішніх джерел без будь-якої первинної обробки, хоча інколи вхідні нейрони можуть виконувати функцію узгодження та нормалізації вхідних сигналів.
Нейрони вихідного шару формують вихідні сигнали і остаточно ідентифікують комбінації вхідних сигналів. Їхні виходи, зазвичай, підключаються до виконавчих елементів, так само як і в біологічних системах вони від’єднуються до м’язів або до залоз зовнішньої чи внутрішньої секреції.

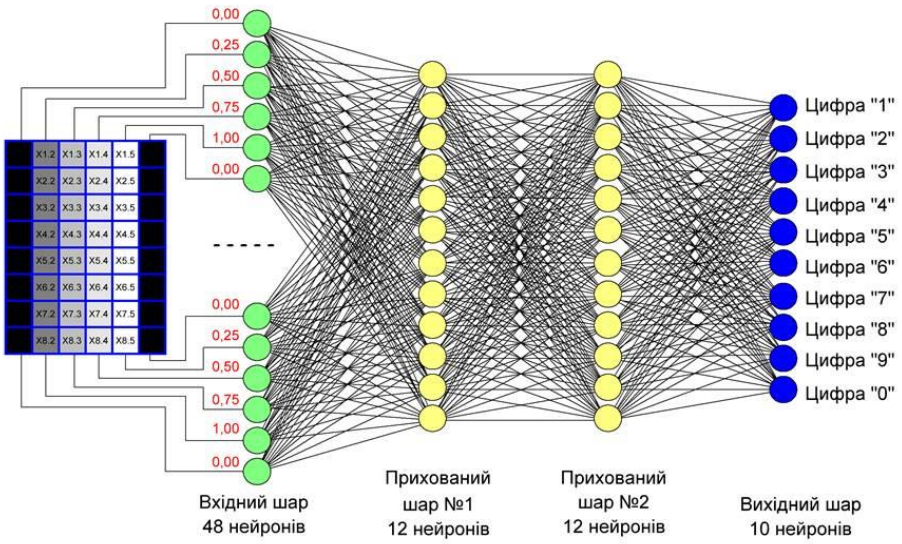


Саме цікаве відбувається у прихованих шарах. Ці шари не мають зв’язку із зовнішнім світом, тому вони і називаються прихованими. У загальному випадку, прихованих шарів може бути будь-яка кількість, у тому числі і нульова – у цьому випадку нейронна мережа складається лише із одного активного вихідного шару. На практиці одного шару нейронів, зазвичай,

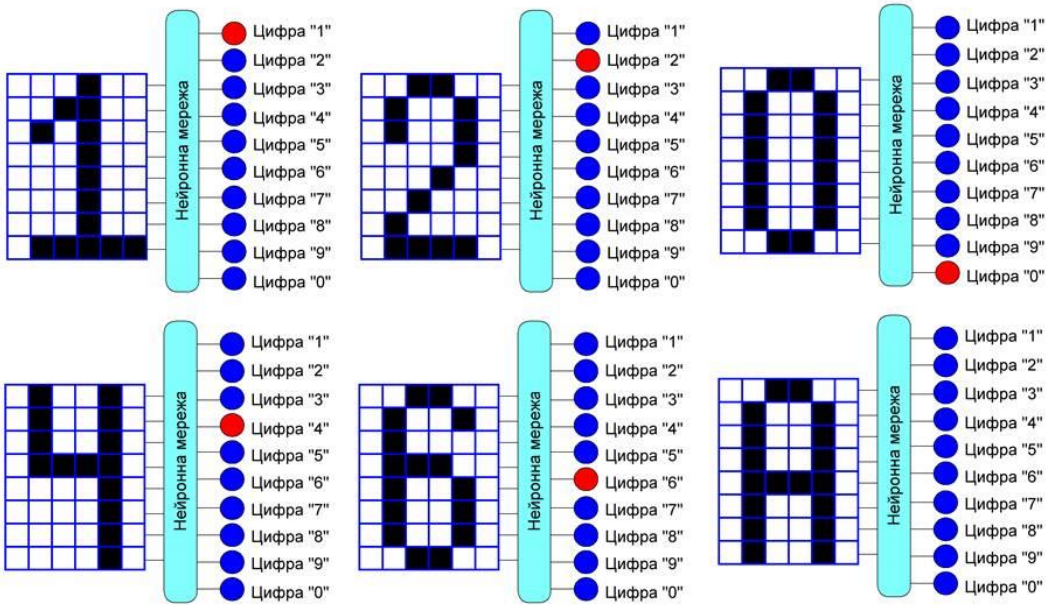
недостатньо, для вирішення більшості задач, тому нейронна мережа все ж таки має приховані шари, кількість яких, зазвичай, не перевищує двох.
Кількість прихованих шарів та кількість нейронів у кожному із прихованих шарів залежить від конкретної задачі. Наразі не існує якихось рекомендацій щодо вибору цих параметрів, які часто називають гіперпараметрами – тобто параметрами, які повинна визначати людина. Зазвичай нейронні мережі містять не більше двох прихованих шарів, але це правило не є абсолютним.
У більшості нейронних мереж усі нейрони у шарі зв'язані зі всіма нейронами у попередньому на наступному шарах – таку систему простіше моделювати та обчислювати. Такий підхід дозволяє також реалізувати нейронні мережі за допомогою доволі простих алгоритмів. Якщо ж потреби у тому чи іншому зв'язку немає, то його завжди можна відключити, встановивши відповідний ваговий коефіцієнт рівним нулю.
Як бачите, штучна нейронна мережа сильно нагадує її біологічного аналога, проте працює вона трішки по іншому. Давайте розглянемо це на прикладі. Нехай у нас буде штучна нейронна мережа, призначена для розпізнавання цифр. Вхідними даними для розпізнавання є фрагмент тексту, що має розміри 8 х 6 пікселів. Отже у нас є загалом 8 х 6 = 48 вхідних сигналів, а це означає, що вхідний шар нашої нейронної мережі повинен мати 48 нейронів. Кожен з яких підключений до фотоелемента, що аналізує яскравість певного пікселя.



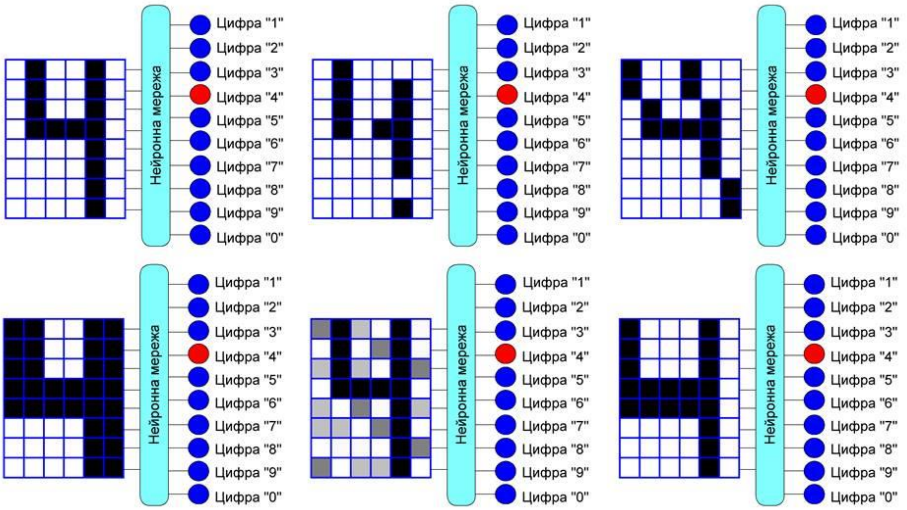
Не будемо зараз вдаватися у технічні подробиці реалізації світлочутливої матриці – нам достатньо вважати, що кожен із фотоелементів формує сигнал, пропорційний яскравості причому, так, що коли цей піксель не засвічений, то вихідний сигнал відповідного фотоелемента дорівнює нулю. А якщо засвічений на 100%, то одиниці. Відповідно, часткове засвічення фотоелементів, що буде відповідати градієнтам сірого, призведе до встановлення на відповідному виході сигналу, що може приймати будь яке значення у діапазоні від 0 до 1.



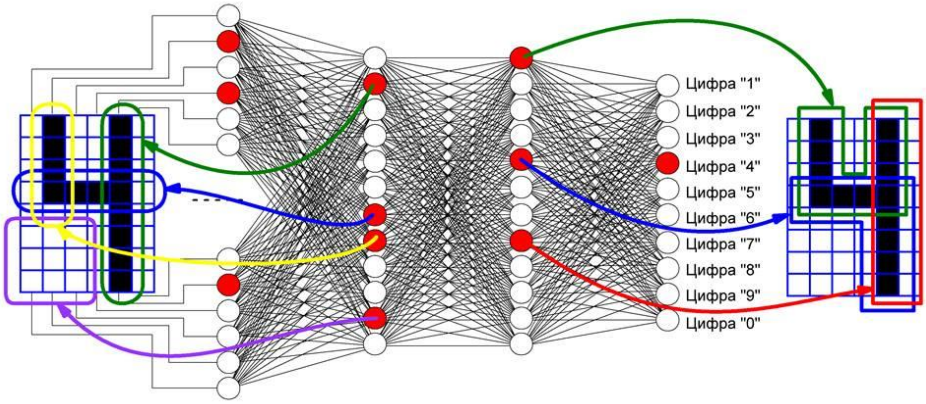
Оскільки наша система призначена для розпізнавання цифр, то вона повинна мати 10 виходів – по одному на кожну цифру. Таким чином, після того, як нейронна мережа буде налаштована, або як кажуть фахівці – навчена, вона зможе розпізнавати образи цифр. І якщо образ засвічених пікселів буде схоже на якусь цифру, то на відповідному виході нейронної мережі буде встановлений високий рівень сигналу. Ну, а якщо пікселі будуть засвічені таким чином, що образ не буде схожий на жодну цифру, наприклад, коли замість зображення цифри сформувати зображення літери, то усі виходи нейронної мережі будуть мати низький рівень.



«А навіщо так складно?» – спитаєте ви – «Невже не можна було розпізнати цифри за допомогою якихось детермінованих алгоритмів?». Я певен, що існують відповідні алгоритми, які дозволять швидко та просто вирішити подібну задачу. Та річ у тому, що ці алгоритми добре справляться лише тоді, коли зображення, що необхідно розпізнати, має відносно високу якість, тобто високу контрастність, чіткі контури а необхідні пропорції. Але ж на практиці таке буває далеко не завжди. І людині інколи доводиться працювати із зображеннями, що далекі від ідеалу. Вони можуть бути пошкодженими, вони можуть мати інші пропорції та навіть не мати деяких фрагментів. Детермінований алгоритм ці зображення навряд чи розпізнає, а от нейронна мережа, як і людина, цілком може справитися із цією задачею.



Але що ж відбувається всередині нейронної мережі? Насправді, цього ніхто достеменно не знає, оскільки нейронні мережі працюють так само загадково, як і людський мозок. Фізично це відбувається приблизно так. Вхідні сигнали з виходів вхідних нейронів подаються одразу на всі нейрони першого прихованого шару. Ступінь зв'язку між певним пікселем та нейроном першого шару залежить від значення відповідного вагового коефіцієнта у конкретному нейроні, тому один і той же сигнал для одного нейрона може бути збуджувальним, а може бути і гальмівним. За певній комбінації вхідних сигналів певні нейрони у першому вхідному шарі збуджуються. На що конкретно вони збуджуються достеменно невідомо і навряд чи це можна однозначно встановити. Більше того, одна і та ж нейронна мережа, але навчена іншим способом, буде збуджуватися вже по іншому.

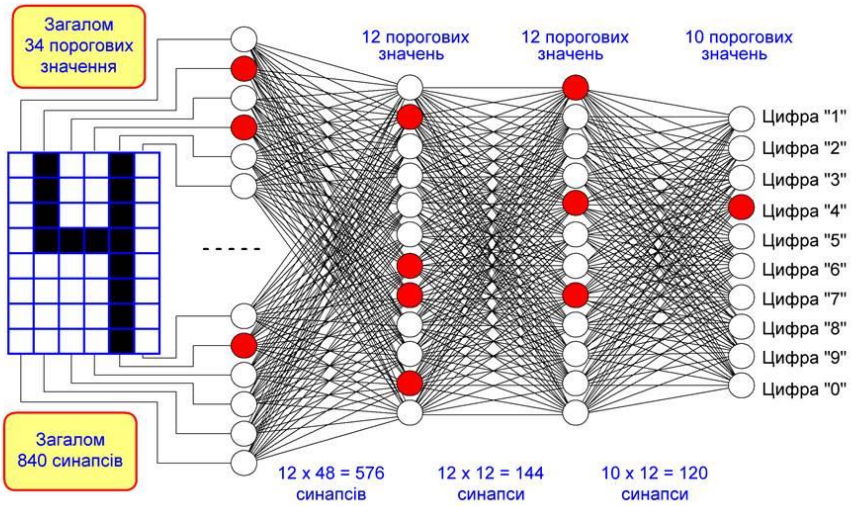


Якщо брати аналогію із роботою мозку людини, то нейрони у першому шарі можуть збуджуватися, наприклад, при виявленні певних графічних примітивів – прямих ліній, кіл, полігонів певної форми та інших. На наступному рівні ступінь абстракції збільшується, тому нейрони другого прихованого шару вже можуть збуджуватися при виявленні більш складних форм. І вже нейрони вихідного шару збуджуються при виявлені фігур, що мають найвищій рівень абстракції – безпосередньо цифр, що є певною комбінацією абстракцій нижчого рівня. Приблизно саме так працює людський мозок – чим далі знаходяться нейрони від джерел сигналів, тим при більшому рівні абстракції вони збуджуються. Проте, не слід забувати, що у випадку штучного інтелекту – це лише припущення – що насправді відбувається всередині навченої нейронної мережі, так само, як і що відбувається всередині головного мозку навіть самої примітивної тварини – поки що ніхто достеменно не знає.

І наостанок, давайте ще раз уважно подивимося на нашу мережу. Її вхідний шар містить 48 нейронів. Ці нейрони лише передають сигнал від вхідних фотоелементів до нейронної мережі і не приймають участі у процесі прийняття рішення. Перший прихований шар містить

12 нейронів, кожен з яких зв’язаний із кожним вхідним нейроном. Отже наша нейронна мережа містить 576 зв’язків, тобто синапсів, між вхідним і першим прихованим шарами. Другий прихований шар містить 12 нейронів, кожен з яких зв’язаний із кожним нейроном із першого прихованого шару. Отже перший і другий приховані шари зв’язані між собою 144 синапсами. І нарешті, вихідний шар має 10 нейронів, кожен з яких зв’язаний із кожним нейроном другого прихованого шару. Отже між вихідним та другим прихованим шаром розташовано ще 120 зв’язків. Таким чином, ця нейронна мережа містить 840 синапсів, які потрібно налаштувати.

Не слід також забувати і про функції активації, які використовуються в нейронах прихованого та вихідного шарів. Ці функції зазвичай також мають певні порогові значення, які, можливо, також необхідно налаштувати. Отже, загалом для налаштування цієї нейронної мережі необхідно правильно встановити 874 взаємозалежних параметра: 840 коефіцієнтів передачі синапсів та 34 порогових значення нейронів.



Хто хоч раз, пробував налаштовувати апаратуру, що має взаємозалежні параметри, знає, що налаштування навіть кількох взаємозалежних параметрів може зайняти доволі багато часу. А що говорити про налаштування майже тисячі взаємозалежних параметрів. Звичайна людина з цим фізично не впорається навіть за кілька років. А це ще далеко не сама складна нейронна мережа. Тому ви розумієте, що побудувати нейронну мережу – то ще не велика робота. Більш складним завданням є її налаштувати так, щоб вона правильно працювала. А це можливо лише при використанні спеціалізованих програмних інструментів, побудованих по відповідним алгоритмам. Але це питання ми розглянемо в одній із наступних лекцій.

Висновки

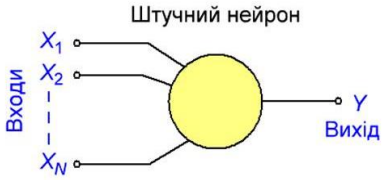
- Штучні нейронні мережі стали першим об’єктом штучного інтелекту
- Штучні нейронні мережі існують більше 60 років і мають найбільш ґрунтовну теоретичну базу
- В історії розвитку штучного інтелекту були періоди бурхливого розвитку та застою
- Штучні нейронні мережі працюють аналогічно нервовим системам живих істот
- Штучні нейронні мережі потрібно **навчати**

Лекція 3. Функції активації штучних нейронів

Із попередніх матеріалів ви вже знаєте, що нейронні мережі за своєю будовою дуже сильно нагадують нервові системи живих істот. Основою нейронних мереж, так само як і основою людського мозку, є нейрони, які можна зв'язати між собою самим чудернацьким способом. Проте різноманіття нейронних мереж поки що значно відстає від різноманіття живої природи. Людина поки що не в змозі опанувати навіть малої частини усіх можливих варіантів побудови штучних інтелектуальних систем і тому вимушена обмежуватися лише кількома типами нейронів та нейронних мереж, які можна технічно реалізувати на практиці. Проте час не стоїть на місці, із кожним днем людина накопичує все більше знань про роботу систем штучного інтелекту, і хто знає, може саме ви станете тими людьми, які виведуть його на зовсім інший якісний рівень. Та поки що продовжимо розбиратися із тим, що уже створено, оскільки саме ці знання повинні стати тією основою, на яку ви будете спиратися у своїй подальшій роботі.

Типи сигналів у нейронних мережах

Ми вже розглядали з вами принципи побудови та роботи штучного нейрона, і тому ви вже знаєте, що штучний нейрон може мати необмежену кількість входів і лише один вихід. Сигнали, якими оперує штучний нейрон, у загальному випадку, можуть приймати будь яке значення, проте розробники нейронних мереж намагаються будувати свої системи таким чином, щоб і вхідні, і вихідні сигнали були обмежені у деякому діапазоні.

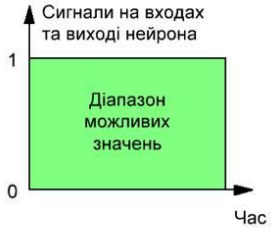


Найбільш поширеним діапазоном сигналів, що використовується у нейронних мережах, є діапазон від 0 до 1. Цей діапазон має кілька переваг. По-перше, він повністю співпадає із двійковими сигналами та ідеально підходить для обчислень за булевою алгеброю. У цьому випадку, вхідні та вихідні сигнали є детермінованими і можуть приймати лише два значення: або нуль, або одиниця. Одиницю можна інтерпретувати як наявність якоїсь події або ситуації а нуль, відповідно, як її відсутність. У випадку ж обчислень за булевою алгеброю, наявність сигналу, що дорівнює одиниці, свідчить про істинність результатів обчислень, а сигналу, що дорівнює нулю – про хибність.

Використання лише двох значень вхідних та вихідних сигналів, тобто або нуль, або одиниця, значно спрощує побудову нейронних мереж, причому як апаратну, так і програмну. Проте можливості подібної системи мало чим відрізняються від можливостей традиційних комп'ютерів, тому нейронні системи, що працюють лише з дискретними сигналами, можуть і не мати особливих переваг перед традиційними програмами на основі детермінованих алгоритмів.

Максимально розкрити усю потужність нейронних мереж можна коли вхідні та вихідні сигнали є неперервними, тобто коли вони можуть приймати будь-яке значення, хоч і, можливо, і у певному обмеженому діапазоні. У даному випадку, діапазон від нуля до одиниці також є доволі зручним, оскільки він дозволяє оперувати нормованими сигналами, тобто сигналами, визначеними у відносній формі. Відповідно до такого підходу, 1 означає, що певний параметр наприклад, рівень пального у баку автомобіля або рівень гучності радіоприймача, має 100% значення, 0,5 – відповідно, 50%, ну а 0 – відповідно, повній відсутності пального, або повному вимкненню звуку. Тобто, у даному випадку, зовсім неважливо скільки літрів пального може поміститися бак автомобіля – 10 чи 500, головне, що він на заповнений на 10%, 30% або 80%.

Діапазон 0...1



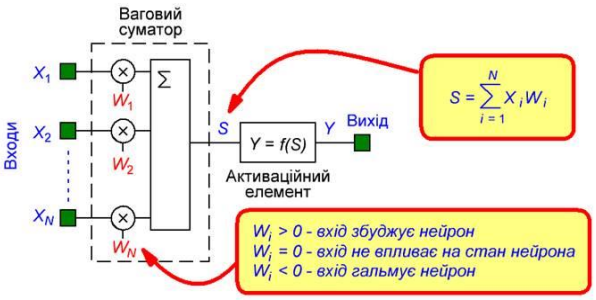
Співпадає із двійковими сигналами та булевою алгеброю
- 0 – Немає, хиба
- 1 – Є, істина

Дозволяє перейти до відносного виду сигналів
- 0,0 – 0%
- 0,5 – 50%
- 1,0 – 100%

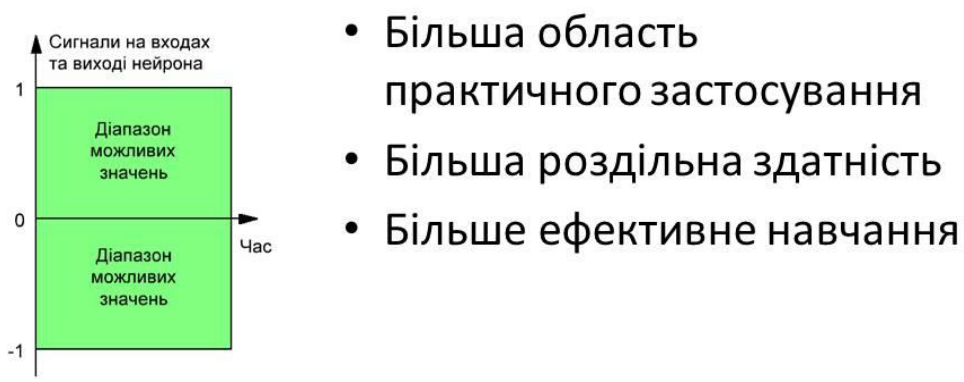
Можна інтерпретувати як імовірність наявності певної події

Крім того, діапазон вхідних та вихідних сигналів від 0 до 1 дозволяє трактувати його як імовірність виявлення тієї чи іншої події. Тут також можна оперувати як безпосереднім значенням вихідного сигналу, так і його значенням, визначеним у відсотках, що дуже часто співпадає із тим, як ми оцінюємо імовірність у реальному житті.
Проте для вирішення деяких задач такий діапазон може не підійти. Але щоб зрозуміти це потрібно згадати як працює штучний нейрон. Ви вже знаєте, що цей вузол складається із двох основних функціональних блоків: вагового суматора та активаційного елемента. Ваговий суматор обчислює алгебраїчну суму вхідних сигналів, причому кожен вхідний сигнал перед тим як увійти у цю суму спершу множиться на відповідний ваговий коефіцієнт, абсолютне значення та знак якого істотно впливають на поведінку нейрона.

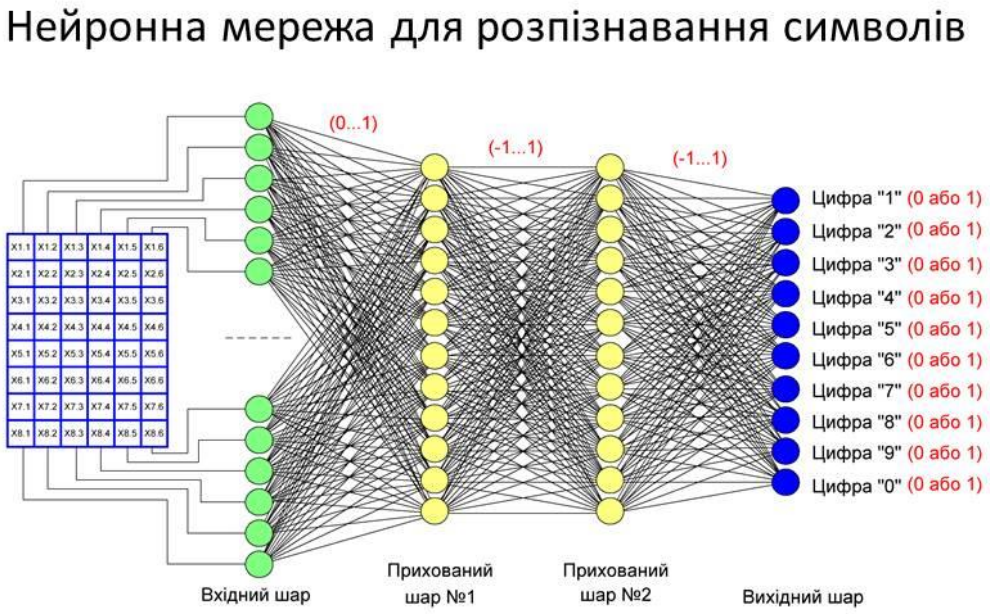
Будова штучного нейрона



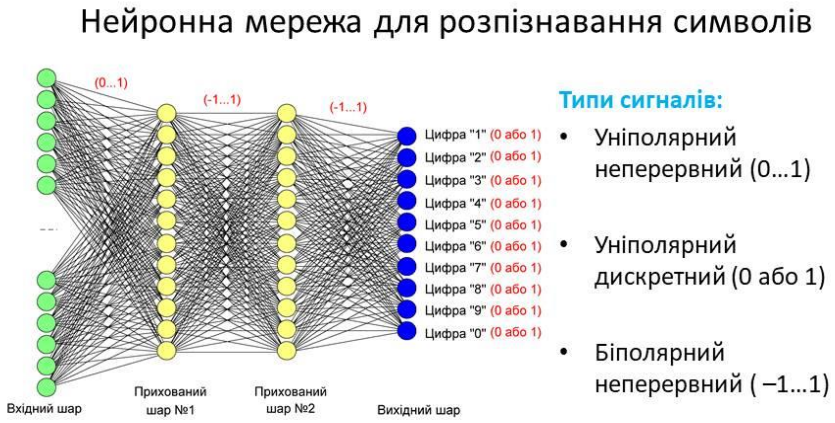
Якщо ваговий коефіцієнт дорівнює нулю, то відповідний вхід нейрона ніяк не буде впливати на стан нейрона. Дійсно, яке би число ми не множили на нуль результат буде однаковим – нуль. Цим іноді користуються для того, щоб відключити у нейронах певні зв'язки. Якщо ваговий коефіцієнт більше нуля, то позитивний сигнал на цьому вході буде збуджувати нейрон, а негативний – гальмувати, тобто перешкоджати збудженню. І чим більші абсолютні значення сигналу та його вагового коефіцієнта, тим більший вплив цього входу на загальний стан нейрону. Якщо ж ваговий коефіцієнт менше нуля, но ситуація змінюється навпаки – позитивний сигнал на вході нейрона буде його гальмувати, а негативний – збуджувати. І, знову ж таки, чим більші абсолютні значення вагового коефіцієнту та вхідного сигналу, тим більший їх вплив на стан нейрона.
Таким чином, перехід до біполярних сигналів, тобто до сигналів, що можуть змінювати свою полярність, значно розширює межі застосування нейронних мереж. Особливо це стосується великих багатошарових систем, у яких використовуються складні алгоритми навчання, наприклад, навчання по градієнтному спуску. У цих випадках перехід до біполярних сигналів дозволяє більш точніше навчити нейронну мережу і тоді вона буде краще виконувати ті функції заради яких її створили.



Зверніть увагу, що зовсім необов'язково, щоб усі сигнали у нейронній мережі були біполярними. Дуже часто, особливо у багатошарових нейронних мережах, буває така ситуація, що зовнішні сигнали, тобто сигнали на вході та виході мережі, є уніполярними, тобто сигналами, що приймають лише позитивні або лише негативні значення, а внутрішні сигнали, тобто сигнали, що розповсюджуються між нейронами, є біполярними, тобто сигналами, що можуть приймати і позивні, і негативні значення. Наприклад, в мережі, призначеній для розпізнавання символів, яку ви зараз бачите на екрані, вихідні сигнали нейронів вхідного шару можуть приймати будь-яке значення у діапазоні від 0 до 1, вихідні сигнали вихідних нейронів можуть приймати тільки сигнали або 0 або 1, а вихідні сигнали нейронів прихованих шарів можуть приймати буть-яке значення у діапазоні від –1 до 1.



Отже у цій системі присутні три види сигналів: уніполярні неперервні, що можуть приймати будь яке значення у діапазоні від 0 до 1, уніполярні дискретні, що можуть приймати лише два значення – або 0 або 1, і біполярні неперервні, що можуть приймати будь яке значення у діапазоні від –1 до 1.



Зверніть увагу, що вид використаних сигналів: уніполярні або біполярні, неперервні або дискретні, має велике значення лише для аналогової техніки, оскільки техніка, що працює із біполярними сигналами, майже завжди, складніша за техніку, що розрахована на обробку сигналів, що не можуть змінити свою полярність. Наприклад, в техніці, що працює з біполярними сигналами, необхідно використовувати більш складні та дорогі мікросхеми та використовувати два джерела живлення – одне для позитивної, а друге – для негативної напруги.

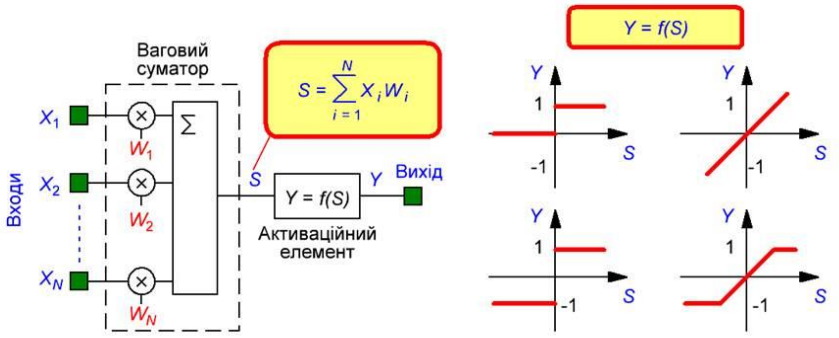
Проте повністю аналогові нейронні мережі, які б працювали максимально наближено до людського мозку, поки ще не створюють – вони виходять дуже складними та дорогими. Тому зараз майже усі нейронні мережі, так чи інакше, будують на основі цифрових мікропроцесорів. А у цьому випадку, типи сигналів, які використовуються всередині нейронної мережі, майже не впливають на складність та вартість системи, оскільки цифрові процесори працюють не безпосередньо із сигналами, а з їх образами – числами, які зберігаються у регістрах та мають формат, який можна інтерпретувати як завгодно. Тобто, одне і те саме значення регістра можна інтерпретувати і як число, що може змінювати свій знак, і як число, яке може приймати лише позитивні значення.

Типи активаційних функцій

Отже сигнали, що присутні всередині мережі можуть бути безперервними або дискретними, а також уніполярними або біполярними, і у більшості випадків це просто числа – певний стан якихось регістрів всередині цифрового мікропроцесора, який може бути інтерпретований різними способами. Із вхідними сигналами усе зрозуміло – вони приходять із зовнішнього світу і ми на них особливо не можемо впливати. А от вихідні сигнали

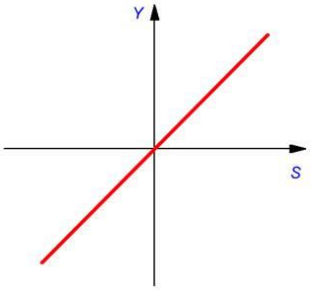
нейронів повинні формуватися безпосередньо нейронами, і відповідає за це вузол, який називається активаційний елемент. Активаційний елемент аналізує сигнал на своєму вході та формує вихідний сигнал відповідно до певного правила, яке називається активаційною функцією. Розглянемо найбільш поширені активаційні функції які використовуються у нейронних мережах та їх вплив на загальну поведінку усієї системи.

Будова штучного нейрона



Найбільш простою є тотожна функція активації. При використанні цієї функції сигнал на виході активаційного елемента завжди дорівнює сигналу на його вході. Фактично, при використанні цієї функції активаційний елемент взагалі не потрібен – у цьому випадку, ваговий суматор можна підключити безпосередньо до виходу нейрона і нічого не зміниться. Проте, незважаючи на свою простоту і, можливо, безглуздість, подібна активаційна функція все ж таки може використовуватись у нейронах вхідних шарів. Ви певно пам'ятаєте, що нейрони вхідного шару дуже часто виконують лише формальну функцію. Проте при розробці програмного забезпечення, особливо при створенні великих та складних нейронних мереж, інколи буває простіше використовувати стандартні програмні модулі – наприклад, універсальний модуль програмного нейрона, який дозволяє налаштовувати кількість входів та активаційну функцію. Тому подібна функція цілком має право на існування, хоч її і використовують і не дуже часто.

Тотожна функція



$$Y = S$$

Особливості:

- Найбільш проста
- Лінійна
- Неперервний вихідний сигнал
- Необмежений діапазон вихідних значень

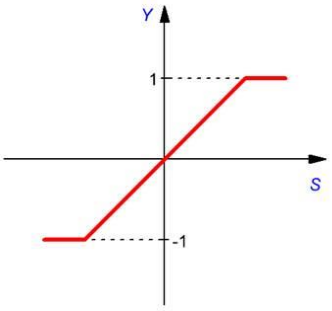
Використання

- Нейрони вхідного шару

Більш зручною для практичного використання є тотожна функція із насиченням. Головною відмінністю цієї функції є те, що її вихідне значення обмежується у певному діапазоні. У загальному випадку – який ви зараз бачите на екрані – вихідне значення цієї функції обмежується у діапазоні від –1 до 1, хоча це правило не є абсолютним – обмеження можу бути на якому завгодно рівні. Така функція також добре підходить для нейронів вхідного шару. У даному випадку вона, не дасть вхідним сигналам досягати великих значень, а це

означає, що, починаючи із певного порога, абсолютне значення сигналу вже не буде впливати на результат роботи мережі. У реальному житті відбувається щось аналогічне. Якщо ми ідентифікуємо певний предмет як «гарячий», то у багатьох випадках це означає, що його температура просто більша ніж 50 °C. І неважливо на скільки вона більша – і предмет із температурою 51 °C , і предмет із температурою 151 °C ми однаково будемо сприймати як гарячий.

Тотожна функція із насиченням



$$Y = \begin{cases} -1, \text{ якщо } S < -1 \\ S, \text{ якщо } -1 \le S \le 1 \\ 1, \text{ якщо } S > 1 \end{cases}$$

Особливості:

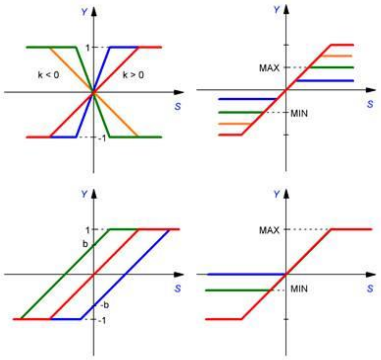
- Проста
- Неперервний вихідний сигнал
- Обмежений діапазон вихідних значень

Використання

- Нейрони вхідного шару

Зверніть увагу, що усі розглянуті перед цим функції були окремими випадками добре усім відомої лінійної функції, яка описується рівнянням y = kx + b, де у якості аргументу x використовувався вхідний сигнал активаційного елемента S. Усі можливі різновиди лінійної функції також можна використовувати у якості активаційної функції нейрона. А от яку саме функцію обрати у тому чи іншому випадку, вже залежить від конкретного призначення нейронної мережі.

Лінійна функція

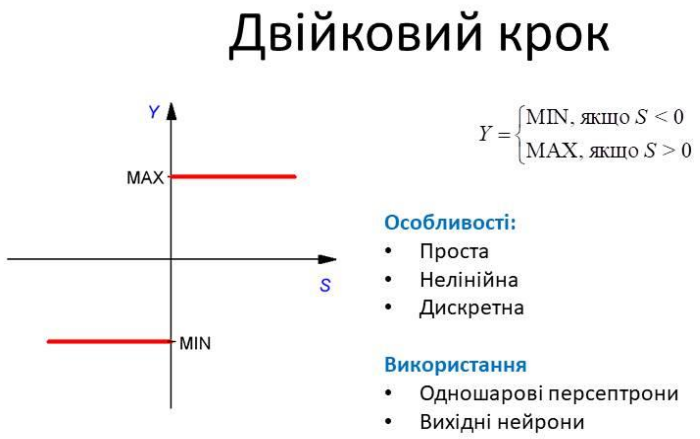


$$Y = \begin{cases} \text{MIN}, \text{ якщо } kS + b < \text{MIN} \\ kS + b, \text{ якщо MIN} \le kS + b \le \text{MAX} \\ \text{MAX}, \text{ якщо } kS + b > \text{MAX} \end{cases}$$

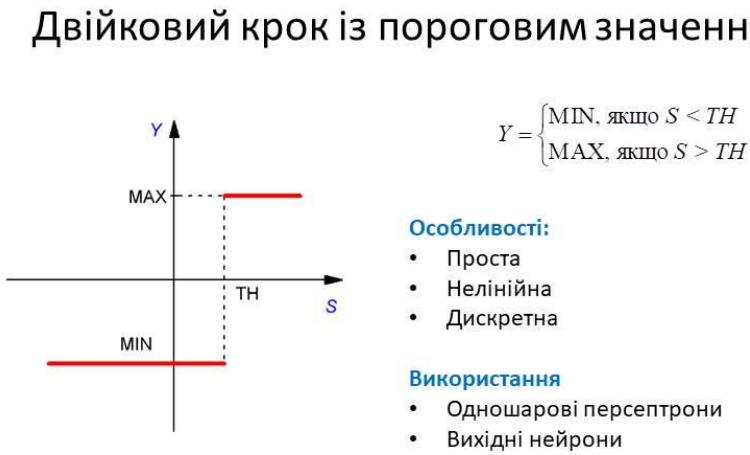
MIN – нижній поріг обмеження (мінімальне значення функції)
MAX – верхній поріг обмеження (максимальне значення функції)
k – кут нахилу лінійної ділянки
b – зміщення по вертикалі

Зверніть увагу, що сигнали з виходу вагового суматора можна посилити або послабити. Для цього достатньо лише змінити коефіцієнт пропорційності k. Чим більше його значення, тим більшим буде кут нахилу активаційної функції відносно вісі S. Пороги обмеження функції активації, у загальному випадку, також можуть відрізнятися від одиниці і бути як симетричними, так і несиметричними. Наприклад, для уніполярних сигналів мінімальний поріг обмеження повинен дорівнювати нулю. Особливу увагу зверніть на параметр b, який зміщує функцію активації по вертикальній вісі. Вибором цього параметра можна змістити функцію активації в область, де вона буде працювати найбільш ефективно. Докладно це явище ми розглянемо, коли будемо знайомитися із нейронами зміщення, а поки вам просто необхідно зрозуміти, що кількість варіантів використання лінійних функцій фактично безмежна, і наразі не існує якихось сталих правил чи рекомендацій щодо їх використання. Але, окрім розглянутих перед цим двох варіантів використання лінійних функцій – тотожної

функції та тотожної функції із насиченням, є ще два цікавих окремих її випадки, як заслуговують на особливу увагу.
Перший варіант утворюється, якщо параметр k збільшити до нескінченності. У цьому випадку, результат функції в області нуля дуже швидко – майже миттєво – переходить від негативного нескінченного значення до позитивного нескінченного значення. Оскільки вихідне значення функції активації штучно обмежується двома порогами – мінімальним та максимальним, то виходить, що в області нуля результат цієї функції стрибкоподібно переходить від мінімального значення до максимального. Параметр b на зовнішній вигляд графіка особливо не впливає, оскільки якщо до якогось числа додати нескінченно велике значення, то результат усе одно буде нескінченним.

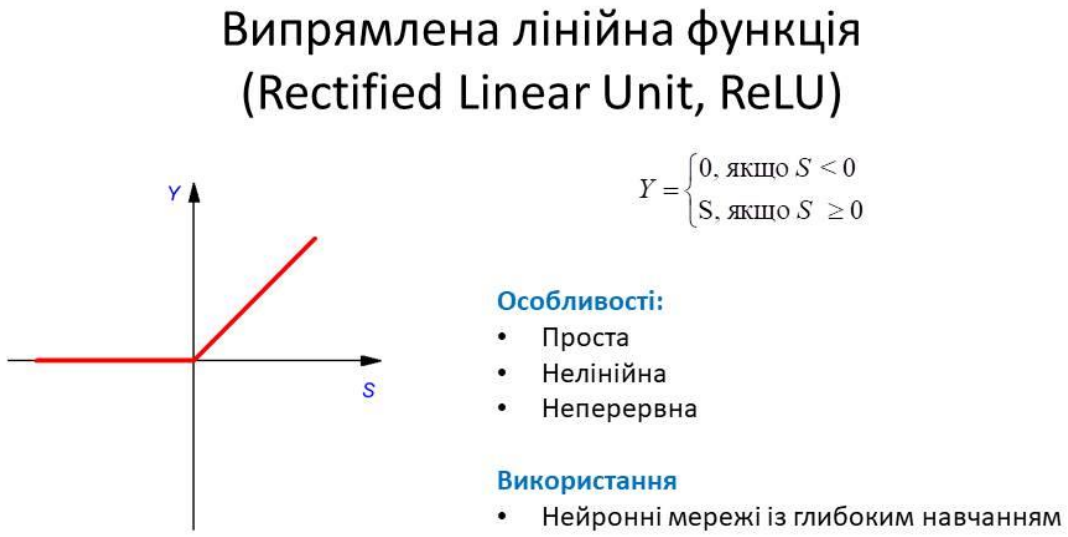


У результаті у нас виходить функція, яка називається двійковий крок або порогова функція. Така функція використовується майже із самого початку розвитку нейронних мереж. Якщо мінімальне значення цієї функції дорівнює нулю а максимальне – одиниці, то вона співпадає із класичної функцією Гевісайда. Проте у нейронних мережах функцію Гевісайда використовують у трішки модифікованому вигляді. Річ у тому, перехід від мінімального до максимального значення у області нуля не завжди є зручним – особливо коли нейронна мережа працює із уніполярними сигналами. Тому її трішки зміщують по горизонтальній вісі за допомогою деякого порогового значення TH, який дозволяє максимально ефективно підлаштовувати нейрон під конкретну задачу.



Якщо порогова функція використовувалась у нейронних мережах із самого початку, то наступний окремий випадок лінійної функції із насиченням, який ви зараз бачите на екрані, почали використовувати в нейронних мережах не так давно. Ця функція отримала назву випрямленої лінійної функції або випрямленого лінійного модуля (Rectified Linear Unit,

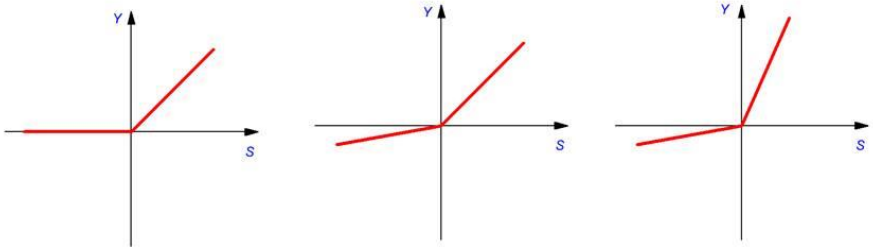
ReLU). Її особливістю є те, що коли сигнал на вході активаційного елемента є негативним, то вихідний сигнал нейрона дорівнює нулю, а якщо позитивним – то вхідному сигналу. Подібні передавальні характеристики мають напвіперіодні випрямлячі на основі діодів, що використовуються у різних електронних вузлах, зокрема, у системах електроживлення, тому ця функція і отримала відповідну назву. Практика використання цієї функції показала, що вона найкращим чином підходить для нейронів, що використовуються у складних багатошарових нейронних мережах, що навчаються за складними алгоритмами, у тому числі і за алгоритмами градієнтного спуску. Відсутність обмеження максимального значення дозволяє краще враховувати помилку роботи мережі під час навчання, що сприяє кращому налаштуванні мережі при розпізнаванні близьких комбінацій вхідних сигналів.



Одним із недоліків випрямленої лінійної функції є наявність сталого вихідного сигналу, коли сигнал на вході активаційного елемента приймає негативне значення. Причому мало того, що вихідний сигнал у цій області не змінюється, так він ще й дорівнює нулю. У деяких випадках така поведінка активаційної функції призводить до погіршення навчальної здатності нейронної мережі, тому у деяких застосунках цю функцію модифікують таким чином, щоб в області негативних значень вхідного сигналу сигнал на виході нейрона все ж таки був ненульовим. Простіше за все це можна зробити, помноживши вихідний сигнал в негативній області на деякий коефіцієнт k, який, зазвичай, набагато менший одиниці. У деяких нейронних мережах вхідний сигнал множать на статичні коефіцієнти як в області негативних, так і в області позитивних значень вхідного сигналу, що є еквівалентом його підсилення або послаблення. Якщо відповідний коефіцієнт пропорційності більший одиниці, то сигнал неначе підсилюється, а якщо менший за одиницю – то, відповідно, послаблюється.
Одним із головних недоліків лінійних функції активації є, власне кажучи, їх лінійність. Докладні математичні дослідження нейронних мереж, які були виконані ще на самому початку їх розвитку, показали, що використання лінійних функцій активації, наприклад тотожної функції, значно зменшує роздільну знатність нейронних мереж. Зокрема, свого часу було математично доведено, що любу багатошарову нейронну мережу у якій використовуються нейрони із тотожними функціями активації, можна перетворити на одношарову, тобто використання лінійних функції активації знищує усі переваги багатошарових нейронних мереж.

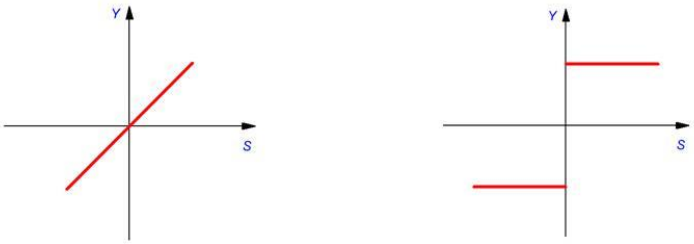
Варіанти функцій активації на основі випрямленої лінійної функції

$$Y = \begin{cases} 0, \text{якщо } S < 0 \\ S, \text{якщо } S \geq 0 \end{cases} \qquad Y = \begin{cases} kS, \text{якщо } S < 0, (k << 1) \\ S, \text{якщо } S \geq 0 \end{cases} \qquad Y = \begin{cases} k_1 S, \text{якщо } S < 0 \\ k_2 S, \text{якщо } S \geq 0 \end{cases}$$



Отже, чим більш нелінійною є функція активації, тим, теоретично, краще нейронна мережа буде розрізняти ті, чи інші комбінації вхідних сигналів, але при цьому нейронну мережу буде складніше навчати, особливо коли потрібно навчати великі багатошарові мережі. А от використання лінійних функції навпаки – суттєво спрощує навчанні мережі, проте їх роздільна здатність, тобто спроможність розрізняти схожі комбінації вхідних сигналів, істотно погіршується. Тому тотожна функція яка, теоретично, дозволить найкраще навчити нейронну мережу, у первісному вигляд майже не використовується, а порогова функція, яка, теоретично, зможе забезпечити найбільшу роздільну знатність, використовується лише у системах із простими алгоритмами навчання.

Лінійна vs. нелінійна



Краща спроможність навчатися
Гірша роздільна здатність

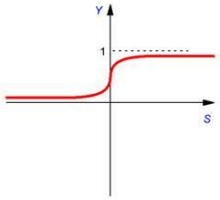
Гірша спроможність навчатися
Краща роздільна здатність

Тому у багатьох нейронних мережах активно використовують більш складні нелінійні функції, які є певними компромісними рішеннями між роздільною здатністю та спроможністю навчатися. Найбільш популярними активаційними функціями є функції із сімейства сигмоїдних, графік яких нагадує латинську літеру S. В нейронних мережах, що працюють з уніполярними сигналами, частіше за все використовують логістичну функцію, яку ще інколи називають «м'який крок», а в мережах, що повинні працювати із біполярними сигналами – функцію на основі гіперболічного тангенсу. Особливістю цих функцій є те, що вони доволі швидко зростають в активні області, тобто в області прийняття рішень, проте, на відміну від порогової функції, яка у цій області має розрив, сигмоїдні функцію є неперервними. Крім того, їх значення також змінюється і в областях далеких від активної, тобто в областях де вхідний сигнал знаходиться далеко від зони прийняття рішень. Чому це так важливо? Давайте розглянемо на прикладі.

Сигмоїдні функції

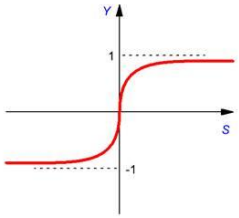
Логістична

$$Y = \frac{1}{1+e^{-S}}$$



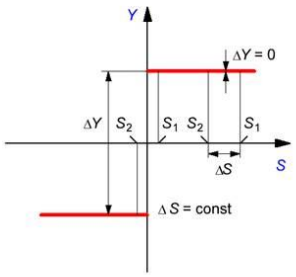
На основі гіперболічного тангенсу

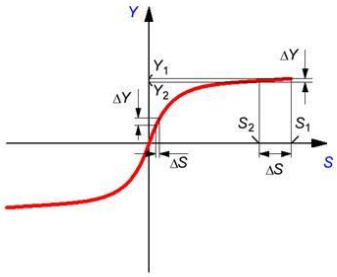
$$Y = \tanh(S) = \frac{e^{S} - e^{-S}}{e^{S} + e^{-S}}$$



Основою навчання нейронних мереж, та й взагалі любого навчання, є визначення наскільки далеко отриманий результат знаходиться від очікуваного. Проте абсолютне значення цієї різниці нам мало що дає. Потрібно ще визначитися у якому напрямку нам необхідно рухатися. Отже, нехай у нас є ваговий суматор, який за певної комбінації вхідних сигналів, помножених на відповідні вагові коефіцієнти, видає деякий сигнал S1, якому відповідає деякий вихідний сигнал Y1. Якщо вихідний сигнал нейрона відрізняється від очікуваного сигналу, то мережу необхідно навчати. Нехай якимось чином, не будемо зараз розбиратися яким саме, ми змінили вагові коефіцієнти, і тепер при тій же самій комбінації вхідних сигналів на вході активаційного елемента присутній вже інший сигнал S2. Якщо активаційною функцією є одна із сигмоїдних функції, то зміна вхідного сигналу S призведе до зміни і вихідного сигналу Y. Нехай ця різниця буде малою, проте навіть при невеликій зміні вихідного сигналу ми зможемо визначити чи зменшилась похибка, чи навпаки – збільшилась. Це дасть нам змогу у процесі навчання визначити, чи рухаємось ми у правильному напрямку, чи ні, чи взагалі не рухаємось. Таким чином, для сигмоїдних функцій можна сказати, що люба зміна їх аргументу призводить до зміни результату.

Монотонність функцій





Для того, щоб ефективно навчити нейронну мережу необхідно, щоб активаційна функція змінювалась при любій зміні аргументу

А тепер спробуємо те ж саме зробити для нейрона із пороговою функцією активації, де зміна значення функції відбувається лише в одній точці, причому одразу на фіксовану величину. Це означає, що під час навчання ми можемо лише визначити факт переходу через поріг. А от у випадку, коли вхідний сигнал змінюється в області, що не містить порогової точки, сигнал на виході активаційного елемента не буде змінюватися зовсім. Отже ми ніяк не можемо визначити, чи призвели наші маніпуляції із ваговими коефіцієнтами до зменшення похибки, чи навпаки – призвели до збільшення, чи може взагалі ніяк не вплинули на роботу мережі.

Обчислювальна складність функцій

Не потребують використання складних алгоритмів (прості)

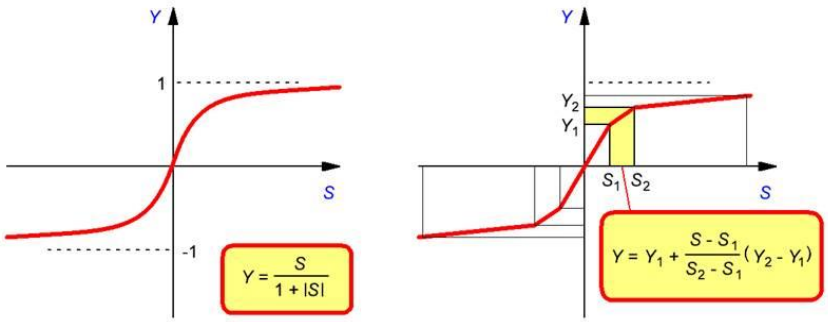
- арифметичні операції (додавання, віднімання, множення, ділення)
- логічні операції (диз'юнкція, кон'юнкція, інверсія, виключна диз'юнкція, тощо)
- взяття модуля
- обмеження за значенням
- кусково-лінійна апроксимація

Потребують використання складних алгоритмів чисельних методів (складні)

- тригонометричні
- експоненціальні
- логарифмічні
- зведення у ступінь
- визначення похідних та інтегралів

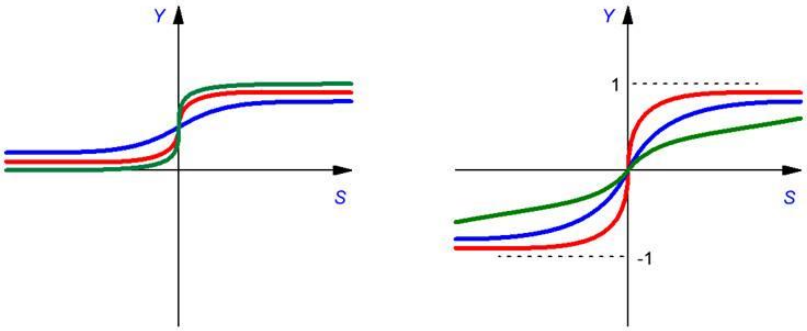
Отже для того, щоб складну нейронну мережу можна було навчити, необхідно щоб функції активації нейронів монотонно зростали і в області збудження нейрону, і за її межами. Таких функцій є доволі багато. Але при їх використанні слід пам'ятати, що обчислення майже всіх математичних функцій, у тому числі експоненціальних, логарифмічних, тригонометричних та інших, потребує використання чисельних методів, що значно збільшує навантаження на процесор нейронної мережу. Тому на практиці знову шукають компроміс: або збільшують точність шляхом використання дорогих та потужних процесорів, зданих у реальному часі обчислювати необхідну кількість математичних операцій, або використовують більш прості функції активації, хоч і, можливо, за рахунок зменшення точності роботи мережі.

Варіанти використання аналогів сигмоїдних функцій



Наприклад, замість активаційної функції на основі гіперболічного тангенсу, можна використати подібну до неї функцію на основі модуля, яка буде обчислюватися набагато скоріше і не буде потребувати дорогого арифметичного сопроцесора. Якщо ж без складних сигмоїдних функцій ніяк не обійтися, то можна скористатися методом кусочно-лінійної апроксимації на замінити безперервні тригонометричні або експоненціальні функції їх кусково-лінійними еквівалентами – тобто розбити увесь робочий діапазон на кілька ділянок, і на кожній ділянці вважати, що ця функція є лінійною. Лінійна функція, яка містить лише арифметичні операції, обчислюється набагато швидше ніж експоненціальні або логарифмічні функції, особливо якщо процесор оснащений модулями апаратного множення, головне, щоб кількість ділянок не виявилась занадто великою і пошук необхідного робочого інтервалу не займав би багато часу.

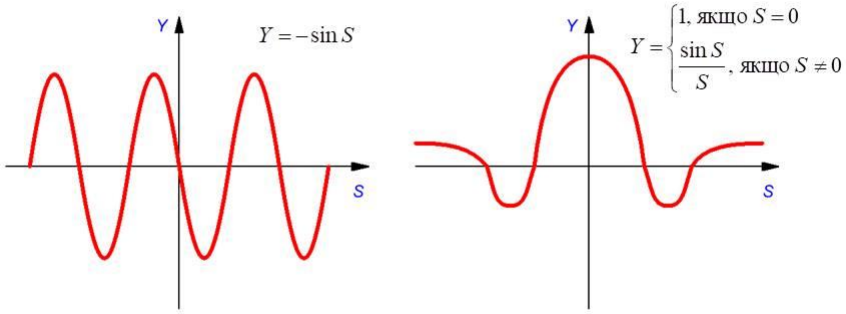
Варіанти сигмоїдних функцій



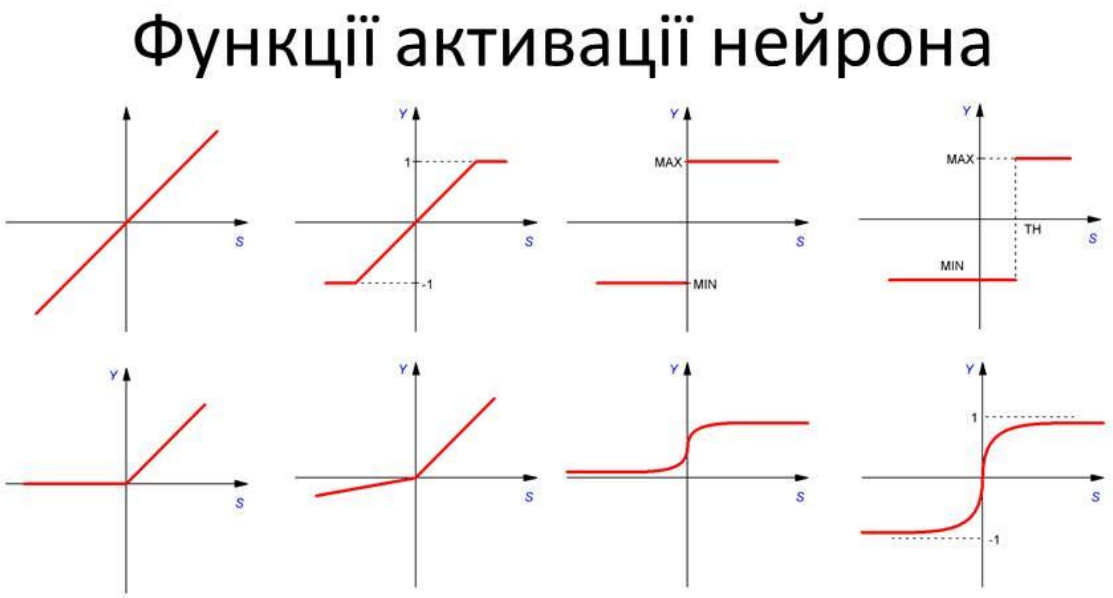
Зверніть увагу, що сигмоїдні функції, так само як і лінійні, можуть бути модифіковані і містити додаткові параметри, які дозволяють змінювати їх крутизну на певних ділянках, робити їх несиметричними та зміщувати відносно початку координат. Усе це робиться для того, що даний нейрон найкраще виконував свою функцію, тобто найточніше ідентифікував необхідну комбінацію вхідних сигналів.

Слід також зауважити, що вибір розглянутих функцій, тобто постійно зростаючих або постійно спадаючих із ростом значення вхідного сигналу не є абсолютним правилом. Є ряд проектів де я якості активаційних функцій використовуються і більш екзотичні для нейронних мереж математичні функції, у тому числі і періодичні. Проте у базовий курс дисципліни по вивченню штучного інтелекту вони не входять.

Варіанти функцій активації



Отже ви бачите, що вибір функції активації для штучного нейрона не є тривіальною задачею. З одного боку, слід забезпечити низьку вартість та простоту реалізації системи, щоб процесор у реальному часі встиг виконати усю необхідну кількість математичних операцій. А з іншого – необхідно забезпечити максимальну точність і роздільну здатність роботи нейронної мережі, а також її ефективне навчання. Тому наразі немає, і, напевно, ніколи не буде, єдиного підходу до вибору функції активізації. У кожному випадку слід обирати ту функцію, яка забезпечить вам необхідний компромісний результат. А це, у більшості випадків, доведеться робити, поки що, лише шляхом проб і помилок.



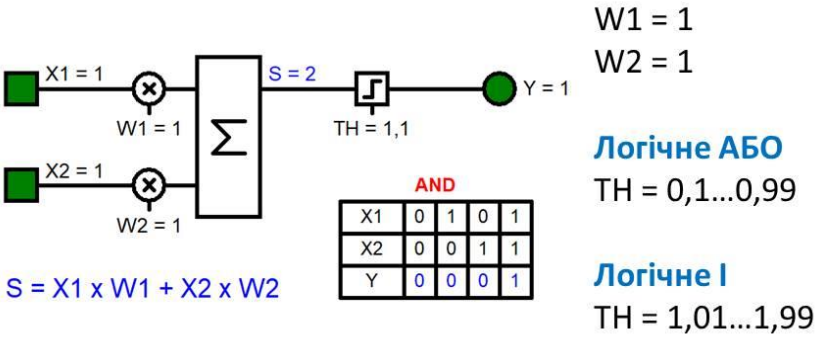
Зміщення вагової суми

На сьогоднішній день, нейронними мережами займається у світі доволі багато людей, кожен із яких має свою думку та погляд на ті чи інші речі. Тому деякі підходи до проектування нейронних мереж у деяких авторів можуть відрізнятися один від одного, а інколи, навіть і суперечити один одному. Тому повинно ще пройти багато часу для того щоб з'явилися усталені традиції та правила у цій галузі, хоча вже зараз можна сказати, що деякі із них вже існують.

У цій частині мова піде про питання, вирішення якого у різних авторів значно відрізняється один від одного, що може викликати певну плутанину. Далі мова піде про зміщення вагової суми. Фактично це питання вирішується дуже просто – шляхом зміщення сигналу на вході активаційного елемента на певну величину, зате формально по цьому питанню різні люди вже навигадували стільки, що людина, яка тільки починає розбиратися у нейронних мережах, може дуже швидко заплутатися.

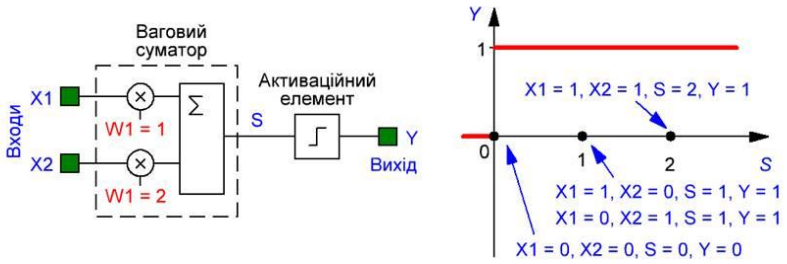
Давайте спершу поговоримо про проблему. Із нею ви вже повинні були попередньо ознайомитися на одному із практичних занять, коли налаштовували найпростіший двохвходовий персептрон для роботи у якості логічного суматора та помножувача. Давайте згадаємо цю роботу. У вас був двохвходовий перспетрон, на кожному із входів якого сигнал міг приймати значення або 0, або 1. Тобто ця нейронна мережа працювала із уніполярними дискретними вхідними сигналами. Вихідний сигнал персептрона також був уніполярним дискретним і міг приймати значення лише 0, або 1. Одне із завдань полягало у тому, що вам потрібно було так налаштувати два вагові коефіцієнти, щоб цей персептрон виконував функцію логічного І, тобто так, щоб сигнал на виході активаційного елемента був рівним одиниці лише тоді, коли обидва сигнали на входах дорівнювали одиницям.

Двохвходовий одношаровий персептрон



Активаційний елемент цього нейрона мав порогову функцію активації. У первісному вигляді, ця функція змінює свій стан при нульовому значення аргументу. А це означає, що при її використанні, точніше при використанні її первісного вигляду – коли вона змінює свій стан при нульовому значенні, реалізувати нейрон, що виконував би функцію І принципово неможливо. І дійсно, вхідні сигнали, можуть приймати значення лише 0 або лише 1. І ці сигнали потім надходять до вагового суматора. Якщо вагові коефіцієнти обох сигналів більше нуля, то нейрон перейде у збуджений стан при появі сигналу. що дорівнює логічній одиниці на любому із входів, тобто замість функції логічного І він почне працювати за правилом логічного АБО. І виправити цю ситуацію шляхом підбору вагових коефіцієнтів принципово неможливо, бо якщо ваговий коефіцієнт буде дорівнювати нулю, то цей вхід буде взагалі відключений, а якщо ваговий коефіцієнт буде меншим за нуль, то цей вхід замість збудження буде лише гальмувати нейрон.

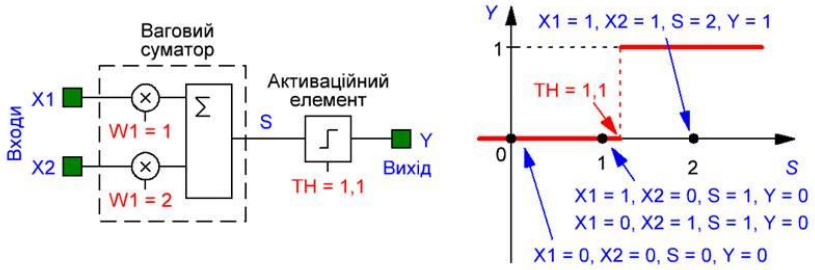
Використання активаційної функції без зміщення



При використанні активаційної функції без зміщення реалізувати функцію логічного І неможливо

У практичній роботі ця проблема вирішувалась доволі просто – активаційний елемент нейрона мав додатковий параметр – поріг активації. Для того щоб нейрон почав виконувати функцію логічного І було достатньо встановити поріг активації більшим ніж максимальне значення вагових коефіцієнтів. Наприклад, якщо вагові коефіцієнти обох входів встановити рівними 1, то для перетворення нейрона на логічний помножувач достатньо було встановити поріг активації у діапазоні від 1,1 до 2. У цьому випадку, поява одиниці лише на одному із входів не могла перевести нейрон у збуджений стан, оскільки зважена сума вхідних сигналів у цьому випадку дорівнювала одиниці, що менше порога його збудження. Отже нейрон збуджувався тільки при появі одиниці на обох входах – у цьому випадку зважена сума на вході активаційного елемента дорівнювала 2, що було більше порога збудження.

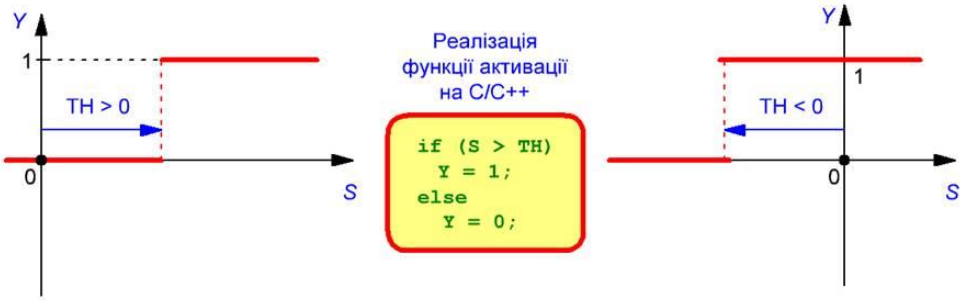
Використання активаційної функції зі зміщенням



При використанні активаційної функції зі зміщенням реалізувати функцію логічного І можливо

А тепер давайте подивимося на ситуацію із іншого боку. Оригінальна функція активації має порогове значення, що дорівнює нулю. Сама функція дуже проста, на язиках програмування високого рівня реалізується, фактично, однією строчкою коду. Тому процесору немає ніякого значення, буде він порівнювати значення вхідного сигналу із константою – у даному випадку нулем – або із якоюсь змінною, у якій зберігається поріг активації. Отже, у цьому випадку пішли по шляху найменшого спротиву – модифікували функцію активації, додавши до неї додатковий параметр – поріг активації. Але зверніть увагу, що поріг активації, фактично, зміщує функцію активації вліво чи вправо на величину цього параметра.

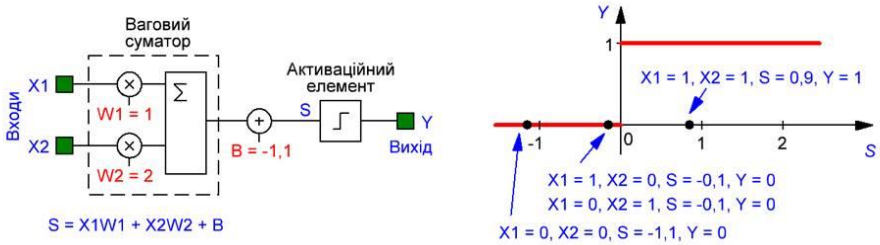
Використання активаційної функції зі зміщенням



Поріг активації TH зміщує активаційну функцію вліво (якщо TH < 0) або вправо (якщо TH > 0)

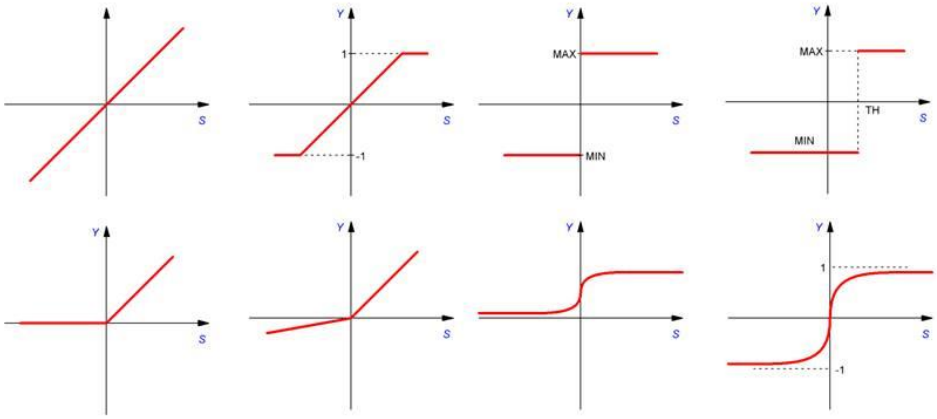
А можна було зробити по іншому. Можна було залишити функцію активації без змін, але змістити сам вхідний сигнал на ту ж саму величину. І дійсно, якщо на виході вагового суматора встановити ще один додатковий суматор, який буде додавати до вагової суми певний статичний коефіцієнт, то результат виявиться тим самим. У передньому прикладі для того, щоб нейрон виконував функцію логічного І при вагових коефіцієнтах обох сигналів рівних 1, необхідно змістити вагову суму на величину від –2 до –1,1. У цьому випадку, при появі на любому із входів нейрона сигналу, що дорівнює 1, максимальний сигнал на вході активаційного елемента не буде перевищувати –0,1, що буде недостатньо для збудження нейрона. А от якщо на обох входах нейрона буде 1, тоді зміщена вагова сума перевищить 0 і нейрон перейде у збуджений стан.

Зміщення вагової суми



Зміщення вагової суми (на B = –1,1) дозволяє реалізувати функцію логічного І для любої кількості входів і з любою функцією активації

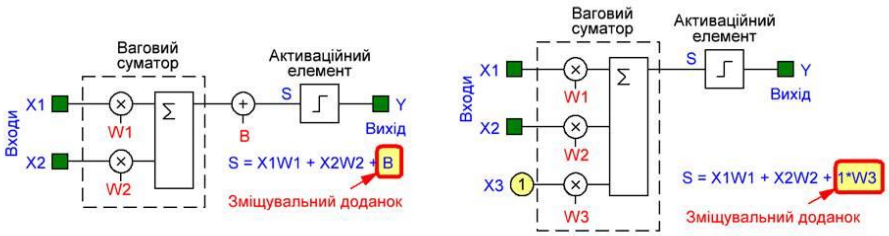
«А у чому різниця?» – спитаєте ви, – «Навіщо виконувати додаткову операцію, якщо можна просто змінити поріг активації?». Дійсно, при використанні порогової функції простіше змінити поріг активації, ніж вовтузитись із зміщенням вхідного сигналу. Проте при використанні інших функції активації, особливо сигмоїдних, змістити їх активну область буде набагато складніше, ніж додати до вхідного сигналу додаткове число. Отже зміщення сигналу на вході активаційного елементу потрібно, бо далеко не завжди вагова сума при необхідній комбінації вхідних сигналів буде знаходитись в активній області активаційної функції. Це просто і це зрозуміло – для цього необхідно після обчислення вагової суми додати певний статичний коефіцієнт. Це розуміють усі люди, що займаються нейронними мережами. А от далі починається суцільна плутанина.



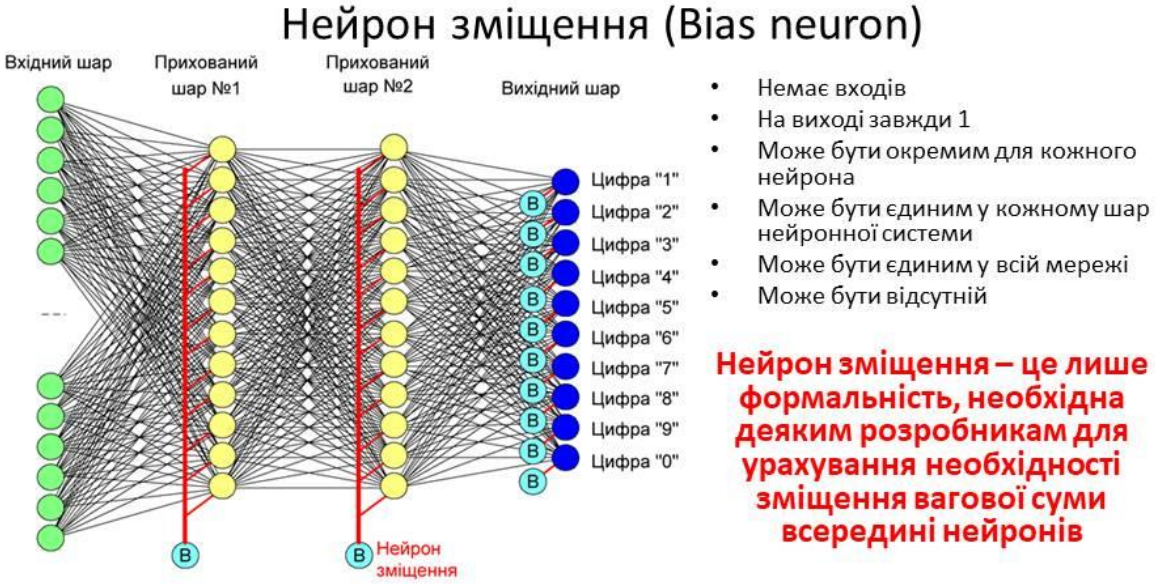
При використанні багатьох функцій активації простіше змістити вагову суму ніж функцію активації

Деякі автори не дуже замислюються над цим питанням і вводять зміщувальний доданок одразу у характеристику нейрона і пишуть, що вагова сума повинна обчислюватись вже з урахуванням додаткового доданку. У цьому випадку, зміщувальний доданок є окремим параметром штучного нейрона, таким же самим як вагові коефіцієнти і функція активації. А от деякі автори для зміщення вагової суми виділяють окремий вхід вагового суматора, на який завжди подається сигнал, що дорівнює одиниці. У такому випадку за величину зміщення вагової суми відповідає окремий ваговий коефіцієнт, величина якого може задаватися вручну, обчислюватися по окремим алгоритмам або обчислюватися на загальних засадах у процесі навчання системи.

Варіанти зміщення вагової суми у штучному нейроні



Обоє з підходів практично рівнозначні, проте у другому випадку з'являється питання: «а куди тепер підключити новий вхід нейрона?», бо залишати його не підключеним якось негарно і нелогічно. І тут з'являється новий вид абстракції, який отримав назву нейрона зміщення (Bias Neuron). Цей нейрон має певні властивості. Перша полягає у тому, що у нього немає входів, а є лише один вихід. А друга властивість полягає у тому, що його вихідний сигнал завжди дорівнює одиниці. Тобто виходить така собі константа всередині нейронної мережі.



Що ж, можливо, такий підхід дозволяє простіше реалізовувати програмне забезпечення для нейронних мереж. Проте деякі автори пішли далі і почали створювати якій незрозумілі додаткові обмеження на нейрони зміщення. Наприклад, у кожному шарі нейронів може бути тільки нейрон зміщення, або у всій нейронній мережі не можу бути більше одного нейрона подібного типу. Саме прикре, що ці роздуми і обмеження вносять лише плутанину. Насправді нейрон зміщення – це звичайна абстракція, спрямована на те, щоб якось формалізувати необхідність зміщення вагової суми всередині нейрона в активну область активаційної функції. І без нейрона зміщення можна обійтись. Проте пам'ятайте, що без формального нейрона зміщення можна обійтись, а от без зміщення вагової суми обійтись навряд чи вдасться. Але це відчуття вже прийде безпосередньо із практикою – коли ви самі будете створювати та налаштовувати нейронні мережі.

Висновки

Таким чином, сьогодні ви взнали багато чого нового. Наприклад, якими можуть бути сигнали всередині нейронних мереж, і якими можуть бути активаційні функції. Навіщо потрібно зміщувати вагову суму і чи можна обійтись без штучного нейрона. І тепер, коли стало зрозуміло якими бувають штучні нейрони, можна переходити до наступного кроку – розбиратися із тим якими можуть бути самі нейронні мережі. Проте цю інформацію я планую довести до вашої свідомості трішки пізніше – наприклад, на наступній лекції.

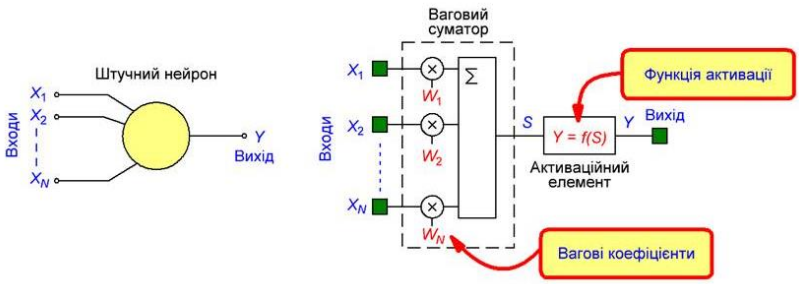
Лекція 4. Нейронні мережі із прямим поширенням інформації

Із попередніх матеріалів ви вже знаєте, що нейронні мережі є доволі цікавими та корисними системами, які можна використовувати у великій кількості застосунків, більшість із яких відноситься до категорії «Hi-Tech», тобто до тієї категорії, яку ми із завмиранням серця вимовляємо як «високі технології». Але насправді, якщо добре розібратися, то виявиться, що нічого надскладного в нейронних системах немає. І дійсно, практичні реалізації нейронних мереж є доволі простими, можна навіть сказати – примітивними, програмами, які зможе створити за кілька годин навіть програміст-початківець. Тоді що ж в нейронних мережах є такого складного, що стримувало розвиток штучного інтелекту на протязі часу, що охоплює вже більше ніж півсторіччя?

Технології нейронних мереж є частиною «високих» технологій

Давайте згадаємо те, що ми вже знаємо. Нейронна мережа складається із штучних нейронів, які працюють приблизно так само, як і нейрони нашого головного мозку. Кожен штучний нейрон має кілька входів, у загальному випадку – необмежену кількість, та один вихід. Усі штучні нейрони, окрім нейронів зміщення, мають однакову будову і складаються із двох основних функціональних блоків: вагового суматора та активаційного елемента. Ваговий суматор обчислює суму усіх вхідних сигналів, помножених на відповідні вагові коефіцієнти, а активаційний елемент аналізує результат роботи суматора, та формує сигнал відповідного до певного правила, яке називається активаційною функцією. Головними параметрами штучного нейрона є вагові коефіцієнти та функція активації. Вагові коефіцієнти це звичайні числа, які зберігаються десь у пам'яті комп'ютера, що обчислює нейронну мережу. Зміною вагових коефіцієнтів можна посилити, послабити та навіть інвертувати сигнали, що приходять на входи нейрона. Підбір вагових коефіцієнтів відбувається під час спеціального процесу, який називається «навчання нейронних мереж» –процесом, із яким ми обов'язково познайомимся, але не сьогодні, а в одній із наступних лекцій.

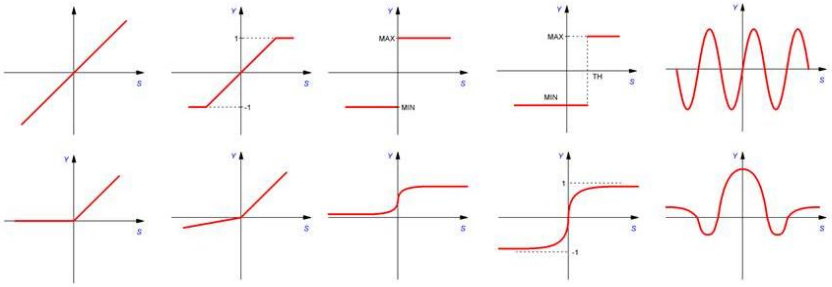
Штучний нейрон



Активаційні функції визначають загальну реакцію нейрона на вхідні сигнали і формують поведінку нейрона під час збудження. Наразі в нейронних системах використовується доволі велика кількість активаційних функцій, кожна із яких має свої переваги та недоліки. На

попередній лекції ми докладно розглядали особливості різних активаційних функції, тому ви знаєте, що існують порогові, лінійні та сигмоїдні функції, переваги та недоліки яких взаємно компенсують один одного. А це означає, що для одних застосунків краще використовувати лінійні функції, для інших – сигмоїдні, а деяких випадках, можливо, навіть періодичні.

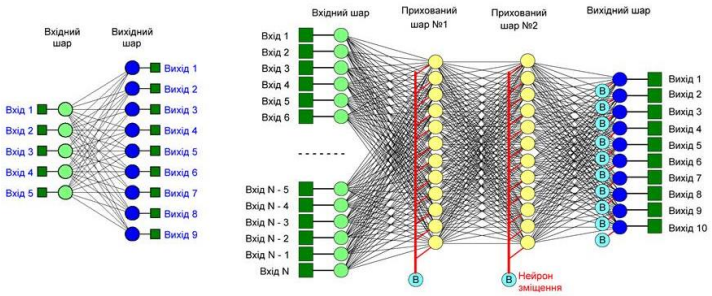
Функції активації нейрона



Ми також вже знаємо, що сам по собі штучний нейрон має невелику користь. Проте, якщо об'єднати нейрони у нейронну мережу, тоді користь від них зросте настільки, що можна говорити вже не тільки, про можливість реалізації якихось інтелектуальних функцій – функцій які не можна реалізувати за допомогою детермінованих алгоритмів – а навіть про можливість появи штучної свідомості – властивості, притаманної, наскільки відомо, лише людям, що мають розум.

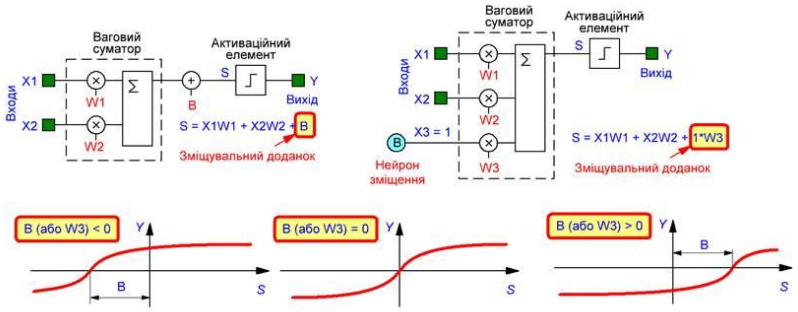
Ви вже знаєте, що нейрони між собою можуть з'єднуватися яким завгодно способом, що утворює нескінченну кількість варіантів побудови нейронних мереж. Та, оскільки людина фізично неспроможна проаналізувати нескінченну кількість варіантів, то їй довелося штучно себе обмежити і зменшити їх кількість до певного числа. Отже зараз типові нейронні мережі складаються із нейронів, що поділяються на шари. У типовій нейронній мережі, як правило, існує один вхідний шар, нейрони якого підключаються до джерел вхідних сигналів, один вихідний шар, нейрони якого підключаються до подальших виконавчих елементів, та певна кількість прихованих шарів, нейрони яких не мають з'єднань із будь-якими елементами поза межами системи.

Варіанти побудови нейронних мереж



Також у системі можуть бути присутні спеціальні нейрони зміщення – це окремий тип нейронів, які не мають входів, а мають тільки один вихід, причому на виході штучного нейрона сигнал завжди дорівнює одиниці. У попередній лекції ми вже розглянули, що нейрони зміщення є необов'язковими, і фізично або програмно вони не реалізуються – це лише абстракція, необхідна для урахування можливості зміщення зваженої суми вхідних сигналів нейрона в активну область активаційної функції, тобто в область максимальної чутливості до необхідної комбінації вхідних сигналів.

Нейрон зміщення

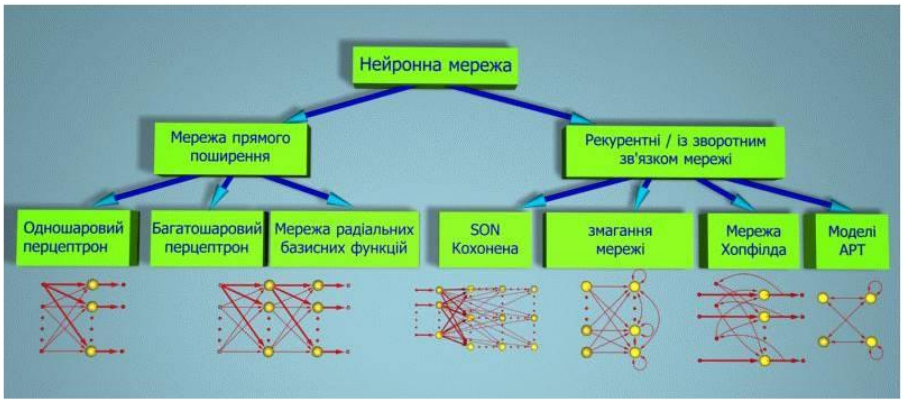


Отже, ви вже доволі багато знаєте про нейронні мережі. Настав час розібратися із тим, як з'єднуються нейрони між собою, тобто яким чином можна утворити нейронну мережу. А точніше, сьогодні ми почнемо розбиратися із тим, які нейронні мережі вже були створені людьми, і які їх особливості вже були виявлені під час практичних випробувань.

Типи нейронних мереж

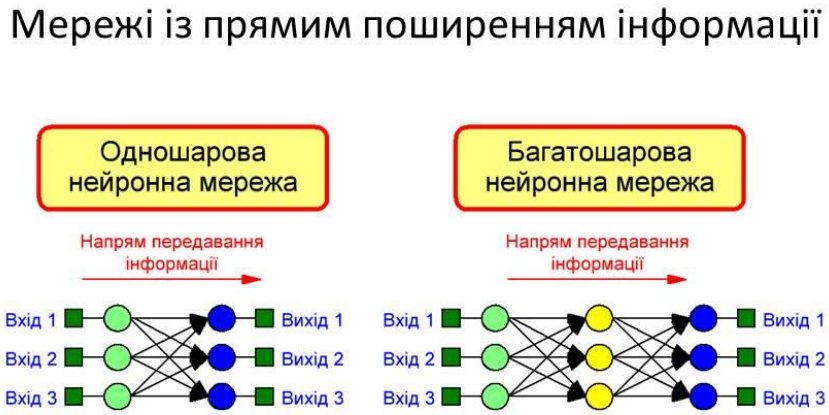
Наразі розрізняють доволі багато варіантів класифікації нейронних мереж. Наприклад, існують гомогенні нейронні мережі – мережі, у яких всі нейрони мають однакові активаційні функції, і гетерогенні – у яких функції активації нейронів різних шарів можуть відрізнятися одна від одної. Також існують одношарові та багатошарові нейронні мережі, причому наразі немає однакової думки про те як правильно підраховувати кількість шарів, у тому числі, чи є вхідний шар повноцінним шаром, чи це є лише черговою абстракцією, причиною появи якої є обмеженість теоретичної бази.
Та якщо ви знайомі із основами наукових досліджень, то знаєте, що класифікація технічних засобів не є вагомим науковим досягненням, оскільки сам процес класифікації чогось, що створено людиною, не містить нових знань, а варіантів групування може існувати нескінченна велика кількість. Тому для первинного ознайомлення із типами нейронних мереж у якості основи візьмемо класифікацію, які ви зараз бачите на екрані, хоча б через те, що вона мені, особисто, просто подобається.

Класифікація нейронних мереж

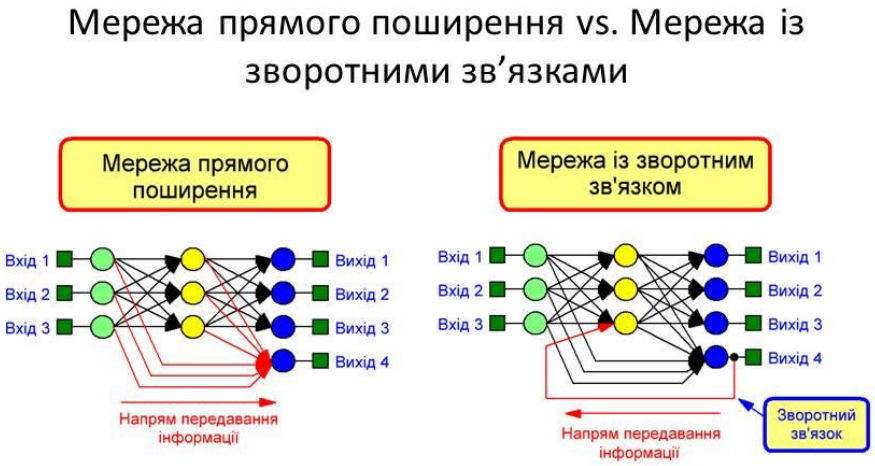


Відповідно до цієї класифікації, усі нейронні мережі поділяються на дві великі категорії: мережі із прямим поширенням інформації та мережі із зворотними зв'язками. Що це таке? Давайте розбиратися.

У мережах із прямим поширенням інформації інформація передається лише в одному напрямку від входів нейронної мережі до її виходів. Це правило справедливе для всіх елементів мережі. Отже, у мережах прямого поширення усі входи нейронів зв’язуються лише із виходами нейронів попередніх шарів, ну а входи нейронів вхідного шару зв’язуються лише із джерелами вхідних сигналів і не можуть мати будь-якого зв’язку із будь-яким іншим елементом мережі. Це правило є головною ознакою мереж прямого поширення і повинно виковуватись для всіх її елементів без винятку.



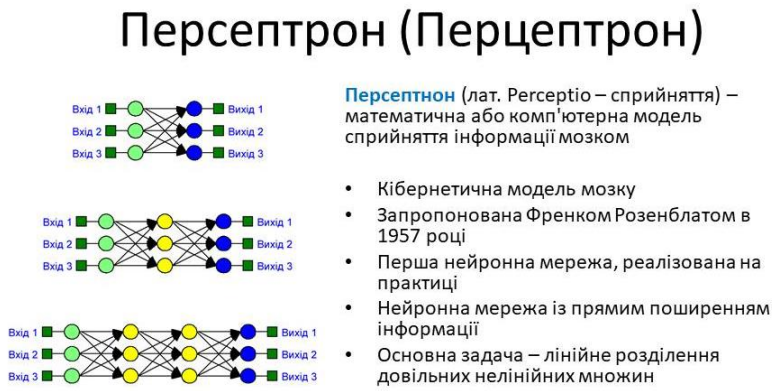
Навіть коли мережа має нетипову конфігурацію, наприклад, коли деякі нейрони зв’язуються не лише із нейронами попереднього шару, а й з нейронами інших попередніх шарів, усе одно, якщо інформація розповсюджується тільки в одному напрямку – у напрямку від виходів нейронів попередніх шарів до входів нейронів, розташованих у наступних шарах – то така мережа класифікується як мережа прямого поширення.



А от якщо в мережі є хоча б один зв’язок, який передає інформацію у зворотному напрямку – від виходів елементів, розташованих у вихідних шарах, до входів нейронів, розміщених у шарах із меншим порядковим номером – то таку мережу вже необхідно розглядати як мережу із зворотними зв’язками. І неважливо скільки цих зворотних зв’язків існує – один чи декілька, і неважливо скільки рівнів охоплює цей зворотний зв’язок – лише один єдиний нейрон, чи декілька нейронів у різних шарах – усе одно така система є системою із зворотними зв’язками, яку необхідно розглядати вже трішки по іншим правилам, ніж мережу із прямим поширення інформації. Отже ви вже, напевно, зрозуміли, що наявність зворотних зв’язків значно змінює поведінку нейронної мережі, настільки – що вони переходять до абсолютного іншого класу нейронних мереж.

Персептрони Розенблата

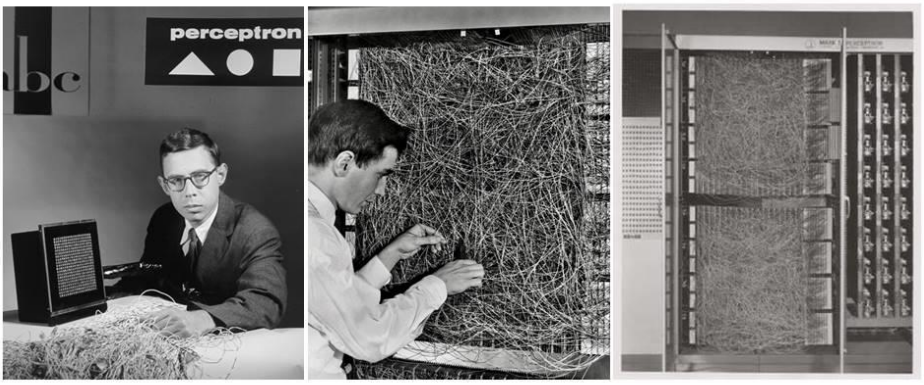
Персептронами називають нейронні мережі прямого поширення, що мають один або кілька шарів. Само слово «персептрон» походить від латинського слова «perceptio», що означає «сприйняття». По-українськи слово «персептрон» звучало би як «сприймальник» або «відчувайлик». Але фахівці, що працюють в галузі нейронних мереж, чомусь не стали його перекладати і просто використали транскрипцію цього слова. Зверніть увагу, що існують два варіанти транскрипції: «персептрон», яку ми використовували і будемо використовувати надалі, і «перцептрон». Різниці між цими двома слова немає ніякої – наразі не існує усталеної термінології, зафіксованої у Державних стандартах України (ДСТУ), тому якщо ви десь побачите слово «перцептрон», то знайте, що це те ж саме, що і «персептрон».
Як видно із походження цього терміну – слово «персептрон» означає «щось сприймати» або навіть «відчувати» так само як і живі істоти. Тобто, технічний прилад, оснащений персептроном, здатний сприймати дійсність, приблизно так, як її сприймає людський мозок. Це дуже сміливе твердження, але не забуваємо, що його запропонували ще в 1957 році, коли досвіду про роботу нейронних мереж ще не існувало.



Формально персептрон є лише математичною або комп'ютерною моделлю. З математичної точки зору персептрон призначений для лінійного розділення довільних нелінійних множин, що в перекладі на просту мову означає, що персептрон здатний ідентифікувати певну комбінацію сигналів на його входах.
Перший персептрон, а фактично – перший нейрокомп’ютер, було створено Френком Розенблатом у 1960 році. Апаратною основою цієї системи став перший американський повноцінний комп’ютер «Automatic Sequence Controlled Calculator», створений гарвардським математиком Говардом Ейкеном та чотирма інженерами компанії IBM на основі ідей англійця Чарльза Беббіджа. Ця обчислювальна машина мала приблизно 765 000 деталей, більшу частину з яких складали електромеханічні реле та перемикачі, досягала у довжину майже 17 м, у висоту – понад 2,5 м і важила приблизно 4,5 тон. Для з’єднання усіх елементів цієї обчислювальної машини було витрачено майже 800 км з’єднувального проводу. Основні обчислювальні модулі синхронізувалися механічно за допомогою 15-метрового вала, що обертався електричним двигуном, потужністю 4 кВт.

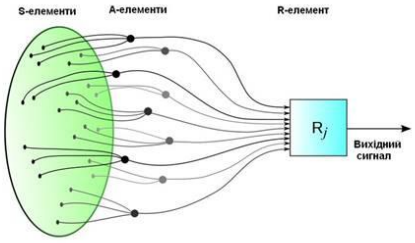
Для створення персептрона цей комп'ютер підключили до чотирьохсот сульфід-кадмієвих фотоелементів, які утворили світлочутливу матрицю із роздільною здатністю 20 на 20 елементів. Зв'язки всередині нейронної мережі налаштовувались за допомогою патч-панелі, а сила зв'язків – за допомогою спеціальних потенціометрів. Не зважаючи на свою простоту, така система могла розпізнавати деякі літери, які підносили до «ока» комп'ютера на спеціальних картках.

Френк Розенблат з першим нейрокомп'ютером «Марк-» (1960 р.)



Відповідно до оригінальних робіт Розенблата, елементарний персептрон повинен складатися із елементів трьох типів: S-елементів, у якості яких виступають датчики або сенсори, A-елементів або асоціативних елементів, які збуджуються при виявлені певних комбінацій вхідних сигналів, та одного елемента R-елемента – вихідного вирішувального елемента, який приймає остаточне рішення по ідентифікації комбінації вхідних сигналів. Як бачите, оригінальна класифікація Розенблата дещо відрізняється від сучасної класифікації нейронних мереж, що вносить певну плутанину у технічній літературі. Крім того, принцип роботи персептрона Розенблата також дещо відрізнявся від принципів роботи тих нейронних мереж, що ми вже розглядали. Тому давайте більш детально із цим розберемося.

Елементарний персептрон Розенблата



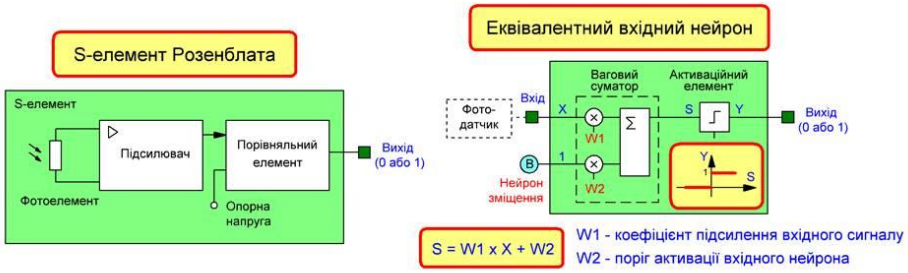
S-елементи (Sensors) – шар сенсорів або рецепторів

A-елементи (Associative Elements) – шар асоціативних елементів, які збуджуються при виявленні певних «асоціацій»

R-елемент (Reacting elements) – елемент, що приймає рішення (реагує)

Почнемо із вхідного шару. У персептроні Розенблата вхідний шар, а точніше –еквівалент вхідного шару, утворюється S-елементами. У даному випадку S-елементи є фізичними датчиками, рецепторами або сенсорами, що можуть знаходитися або у стані спокою, або у збудженому стані. У реальній системі вони відповідають, наприклад, світлочутливим клітинам сітківки ока або фоторезисторам матриці камери. Кожен S-елемент може перебувати в одному з двох станів – спокою або збудження. У першому випадку вихідний сигнал S-елемента дорівнює нулю, у другому – одиниці. Отже вихідні сигнали S-елементів по своїй природі повинні бути дискретними.

Персептрон Розенблата vs. Сучасна нейронна мережа



У сучасних системах конкретну фізичну реалізацію датчиків вхідних сигналів, що оброблюються нейронними мережами, зазвичай, не враховують. Аналіз роботи нейронних мереж починають із вхідних нейронів, а точніше із нейронів, на які подаються довільні вхідні сигнали. Отже сучасними еквівалентами S-елементів персептронів Розенблата є одновходові нейрони, із пороговою функцією активації – саме вони забезпечать дискретний вхідний сигнал.

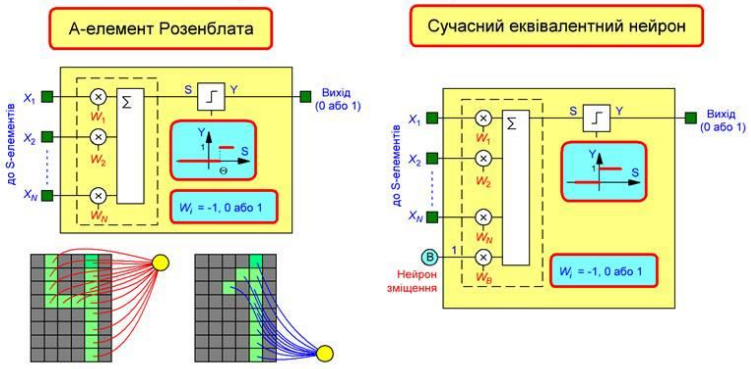
Зверніть увагу на особливості вагових коефіцієнтів такого нейрона. Ваговий коефіцієнт W1 визначає коефіцієнт підсилення або послаблення вхідного сигналу X. Якщо абсолютне значення цього коефіцієнту більше за одиницю, то вхідний сигнал буде підсилюватися, а якщо менший – то, відповідно, послаблюватися. Крім того, якщо значення цього коефіцієнту буде негативним, то вхідний сигнал буде інвертуватися, що дозволить системі працювати, у тому числі, і з уніполярними вхідними сигналами. А от ваговий коефіцієнт W2 фактично визначає поріг активації нейрона, тобто при якому рівні вхідного сигналу він перейде у збуджений стан. Отже, якщо докладно проаналізувати можливості запропонованого вхідного нейрона, то виявиться, що він може працювати із любими сигналами, тобто сигналами, що можуть мати будь-яку полярність, та будь-який рівень.

Вагові коефіцієнти нейронів, що є еквівалентами S-елементів персептронів Розенблата, повинні бути фіксованими і не змінюватися у процесі навчання нейронної мережі. Я вважаю, що це є цілком зрозумілим, оскільки вони залежать від характеристик первинних сигналів і їх зміна у процесі навчання може призвести до того, система повністю перестане працювати.

Асоціативні елементи, або «А-елементи», персептрона Розенблата мають кілька входів, які підключаються до S-елементів. Відповідно до оригінального задуму автора, кожен асоціативний елемент повинен збуджуватися при появі певної комбінації вхідних сигналів, тобто при появі певного образа. Наприклад, у системі, призначеній для розпізнавання цифр, можуть бути асоціативні елементи, які збуджуються при появі комбінацій сигналів, які виникають при появі зображень певних цифр, наприклад, «4» або «1». Проте це не обов’язково – у системі цілком можуть існувати асоціативні елементи, які збуджуються при появі навіть окремих елементів літер, чи цифр, наприклад, ліній, кіл, прямокутників, дуг, трикутників, розташованих як у певних частинах світлочутливої матриці, так і на всій її площині.

Кожен асоціативний елемент має ваговий суматор та пороговий елемент. Асоціативні елементи не мають фіксованої кількості входів, отже кожен асоціативний елемент може з’єднуватися лише із певною кількістю вхідних S-елементів. Враховуючи, що у першому комп’ютері зв’язки між нейронами були фізичними і виконувались зо допомогою проводів, то таке рішення виглядає цілком зрозумілим, оскільки для з’єднання А-елемента зі всіма S-елементами, необхідно було підключити 400 проводів. Тому цілком природно, що деякі асоціативні елементи можуть не мати зв’язку із вхідними елементами, наприклад, елементами, розташованими у зоні, яка не використовується у дані асоціації.

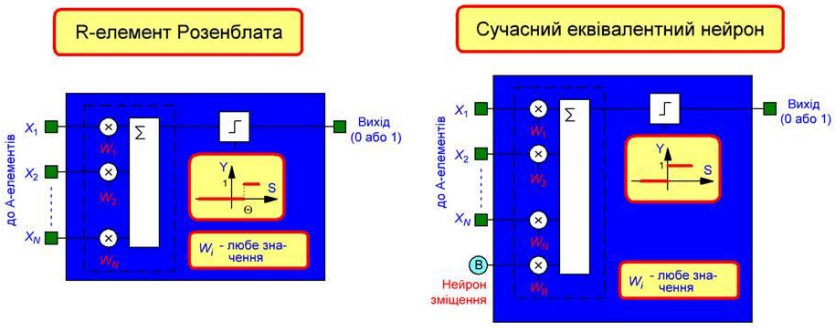
Персептрон Розенблата vs. Сучасна нейронна мережа



В персептронах Розенблата ступінь зв'язку між асоціативними та вхідними елементами є фіксованою і не може змінюватися в процесі навчання нейронної мережі. Вагові коефіцієнти усіх А-елементів є фіксованими і можуть приймати значення лише –1, 0 або 1. Якщо ваговий коефіцієнт певного входу асоціативного елемента має значення, що дорівнює 1, то засвічення фотоелемента, зв'язаного із цим входом, допомагає збуджувати цей елемент, якщо –1 – то перешкоджає збудженню, тобто гальмує, ну а якщо цей коефіцієнт дорівнює нулю, то цей вхід ніяк не впливає на стан цього А-елемента. Поріг збудження Θ асоціативного елемента підбирається експериментально у процесі навчання системи.

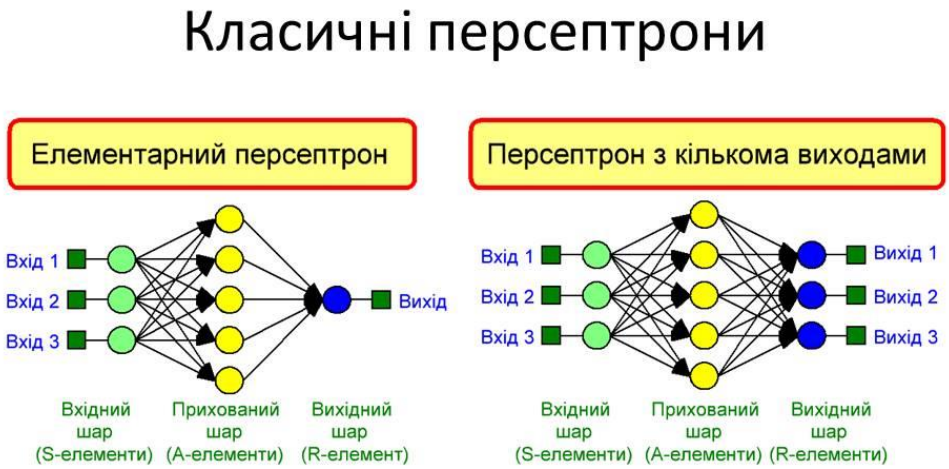
Отже, кожен асоціативний елемент персептрона Розенблата має певну кількість входів, фіксовані вагові коефіцієнти і порогову активаційну функцію із змінним порогом збудження. В оригінальній версії системи, А-елемент переходить у збуджений стан, якщо кількість засвічених вхідних фотоелементів, які збуджують цей нейрон, перевищує кількість елементів, які його гальмують. Сьогодні асоціативні елементи персептрона Розенблата можна замінити типовими штучними нейронами, що також мають порогову функцію активації і фіксовані вагові коефіцієнти. Єдиною відмінністю сучасної реалізації асоціативних елементів є наявність нейрона зміщення, який дозволяє змінити поріг активації шляхом підбору відповідного додаткового вагового коефіцієнту.

Персептрон Розенблата vs. Сучасна нейронна мережа

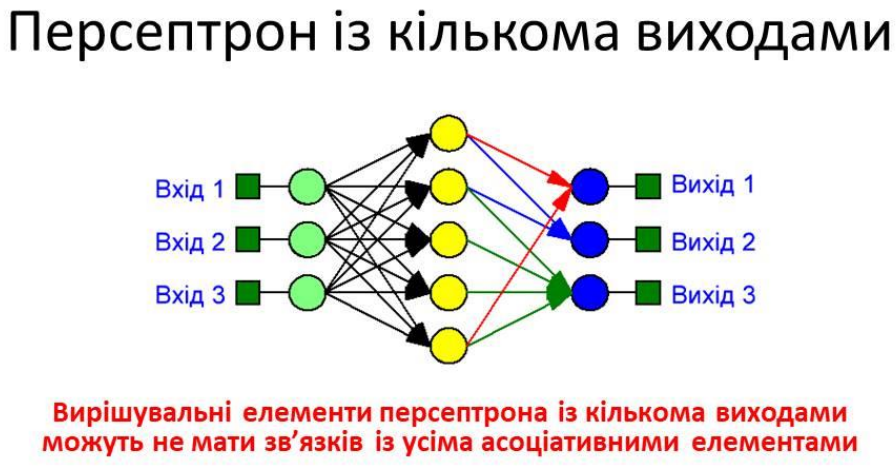


В елементарному персептроні, що був запропонований Розенблатом, елемент, що приймає рішення, або R-елемент, лише один. За своєю структурою він нічим не відрізняється від асоціативних елементів. Єдиною відмінністю вирішувального елемента від асоціативного є те, що вагові коефіцієнти вирішувального елемента не є фіксованими і можуть приймати будь-яке значення. Вихідний сигнал вирішувального елемента, так само як і асоціативного, є дискретним і може приймати значення 0 або 1. Ну і зрозуміло, що через наявність лише одного вирішувального елемента елементарний персептрон спроможний розрізнити лише одну комбінацію вхідних сигналів.

Отже класичний персептрон Розенблата є нейронною мережею, що має один вхідний шар, один прихований шар, та один вихідний шар. Елементарний персептрон має лише один вирішувальний елемент і може розпізнати лише одну комбінацію вхідних сигналів. Зрозуміло, що користі від нейронної мережі, що має лише один єдиний вихід, небагато, тому кількість вирішувальних елементів у вихідному шарі, зазвичай, збільшують. І тут виникає певна особливість, на яку необхідно звернути увагу.

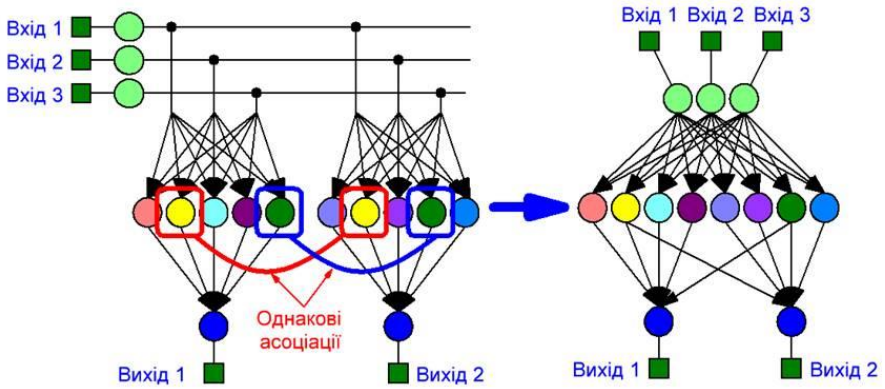


Елементарний персептрон, тобто мінімальна комірка, яка може прийняти рішення, має лише один вирішувальний елемент, який обов'язково зв'язаний із усіма асоціативними елементами. Це зрозуміло, бо інакше навіщо нам у елементарному персептроні асоціативні елементи, що генерують асоціації, які не беруться до уваги при прийнятті рішення. Але якщо ми будемо збільшувати кількість виходів, то це правило може бути порушеним, оскільки у багатоканальному персептроні вирішувальні елементи можуть не мати зв'язку із деякими асоціативними елементами. Чому так? Давайте розглянемо приклад.



Нехай у нас є багатоканальний персептрон, призначений для розпізнавання цифр. Він фактично складається із певної кількості елементарних персептронів, що працюють незалежно один від одного. Проте може виникнути така ситуація, коли в асоціативному шарі певні елементи при появі різних цифр будуть збуджуватися абсолютно однаково. Наприклад, при появі цифр «5» та «6», які дуже схожі, можуть виникати деякі абсолютно однакові асоціації, пов'язані із засвіченням абсолютно однакових ділянок світлочутливої матриці. Це означає, що при об'єднані двох елементарних персептронів кількість елементів у проміжному асоціативному шарі можна зменшити.

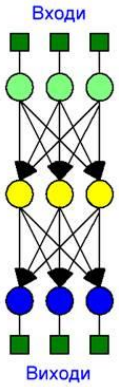
Персептрон із кількома виходами



Отже ви вже, напевно, зрозуміли, що з часів Розенблата тлумачення терміну «персептрон» трішки змінилося, і зараз для того, щоб нейронну мережу назвали персептроном, необхідно виконання певних умов, хоча жодна з них не є обов'язковою.

По-перше, персептрони Розенблата відносяться до мереж із прямим розповсюдженням інформації, тобто ніяких зворотних зв'язків у цій мережі не повинно бути. Але це тільки теоретично. За весь час існування нейронних мереж були спроби побудувати персептрони і іншого типу. Отже не дивуйтесь, якщо ви знайдете десь опис персептрона із перехресними або зворотними зв'язками. Це також може бути, бо автори цих мереж колись також називали їх персептронами.

Сучасні ознаки персептрона Розенблата

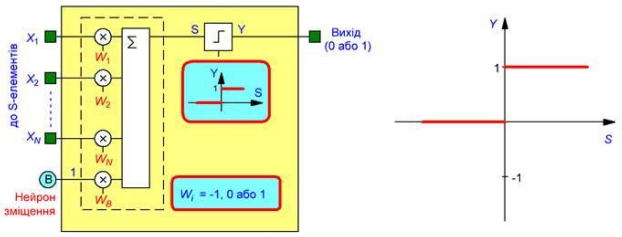


- Мережа із прямим поширенням інформації
- Вихідні сигнали усіх нейронів є дискретними (або 0 або 1)
- Елементи вхідного шару (S-елементи) не приймають участі в розпізнаванні сигналу
- Передача сигналу між нейронами відбувається миттєво, або на протязі фіксованого інтервалу часу

Другою ознакою персептрона Розенблата є дискретний характер вихідних сигналів усіх нейронів. Тобто, усі вихідні сигнали елементів вхідного, асоціативного та вирішального шарів повинні приймати значення тільки 0 або 1. У більшості сучасних персептронів це правило, зазвичай, виконується проте і воно вже стало не обов'язковим.

В персептронах Розенблата використовувались порогові функції активації. І хоч це правило також не є догмою, якщо є така можливість, то його намагаються дотримуватись. Також не є обов'язковим використанням фіксованих вагових коефіцієнтів в нейронах асоціативного шару. Нагадую, що в перших версіях персептронів вагові коефіцієнти в елементах асоціативного шару могли приймати лише значення –1, 0 або 1. В сучасних персептронах ці вагові коефіцієнти можуть приймати теоретично будь яке значення.

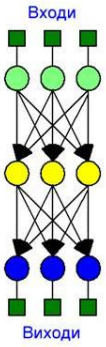
Сучасні версії персептрона



Використання порогової функції активації та фіксованих вагових коефіцієнтів в нейронах асоціативного шару вже не є обов'язковою умовою для визначення нейронної мережі як персептрона

У якості додаткових ознак персептрона можна зазначити, що елементи вхідного шару не повинні приймати участі у розпізнаванні вхідних сигналів, а також те, що сигнали між нейронами повинні передаватися миттєво, або на протязі певного фіксованого часу. Ці ознаки є принциповими для нейронних мереж, що мають фізичну реалізацію, наприклад, для нейрокомп'ютерів. А якщо нейронна мережа реалізується програмно, то, наприклад, миттєве розповсюдження сигналів забезпечується майже автоматично.

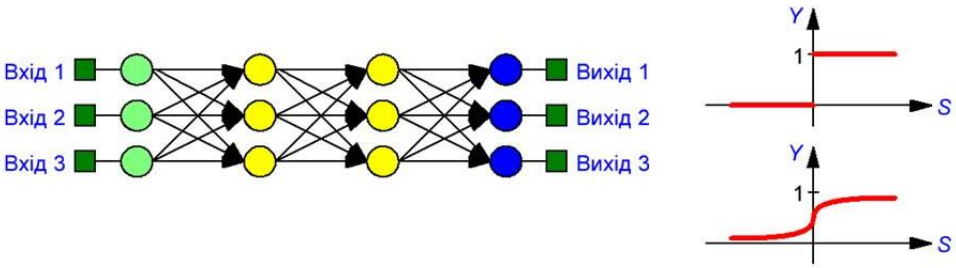
Сучасні ознаки персептрона Розенблата



- Мережа із прямим поширенням інформації
- Вихідні сигнали усіх нейронів є дискретними (або 0 або 1)
- Елементи вхідного шару (S-елементи) не приймають участі в розпізнаванні сигналу
- Передача сигналу між нейронами відбувається миттєво, або на протязі фіксованого інтервалу часу

Отже, якщо врахувати усі сучасні уточнення та доповнення оригінального терміну, «персептрон», запропонованого Розенблатом, то залишиться лише наступне. Персептроном можна назвати нейронну мережу із прямим розповсюдженням інформації, усі нейрони якої спроможні дуже швидко змінювати свій стан стан. Отже від оригінального визначення залишились лише дві ключові ознаки, які і дозволяють називати ту чи іншу нейронну мережу персептроном.

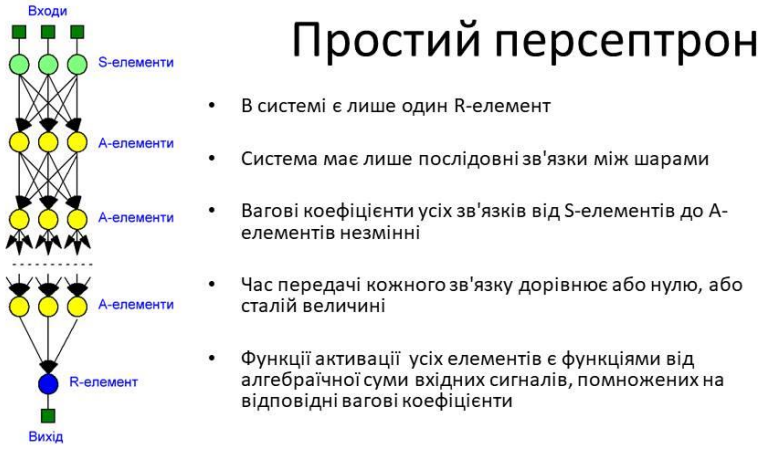
Сучасні ознаки персептрона



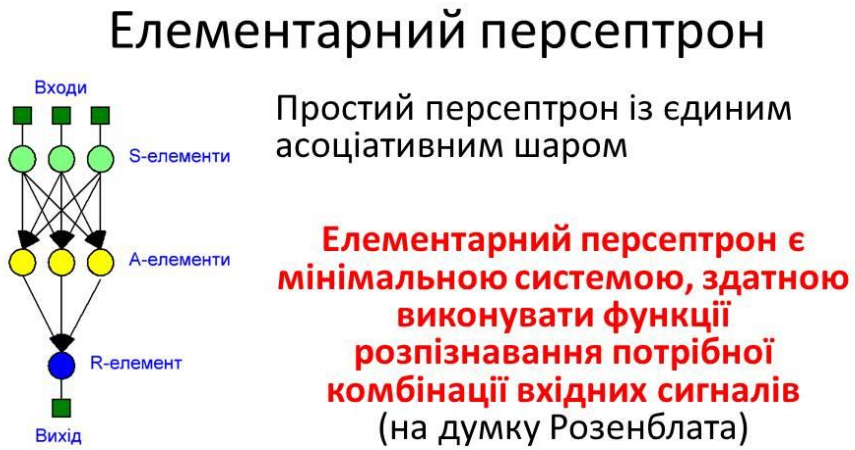
Персептроном можна назвати нейронну мережу із прямим розповсюдженням інформації, усі нейрони якої спроможні швидко змінювати свій стан

Проте існують кілька означень які не змінилися із часів Розенблата. Наприклад, простим персептроном називають нейронну мережу, що має п'ять наступних ознак. По-перше, в

системі є лише один вирішувальний елемент, який природно зв’язаний із усіма елементами, розташованими в попередньому асоціативному шарі. По-друге, мережа має лише послідовні зв’язки між шарами, тобто, виходи елементів вхідного шару з’єднані тільки із входами елементами першого асоціативного шару, виходи елементів першого асоціативного шару з’єднані лише із входами елементів наступного асоціативного шару і, нарешті, усі елементи останнього асоціативного шару з’єднані тільки із одним єдиним елементом, розташованим у шарі прийняття рішень. По-третє, вагові коефіцієнти усіх нейронів асоціативних шарів є незмінними в процесі навчання нейронної мережі, хоча вони можуть змінюватися у процесі її створення. Четвертою ознакою є сталість розповсюдження сигналів всередині нейронної мережі, і, нарешті, останньою – п’ятою – ознакою є те, що функції активації є функціями від алгебраїчної суми вхідних сигналів, помножених на відповідні вагові коефіцієнти.

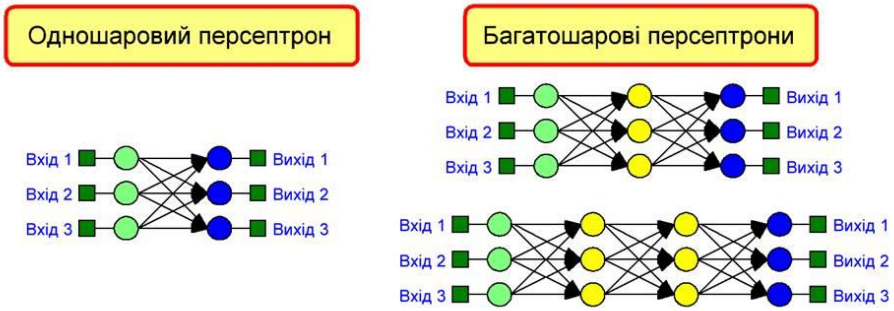


Іншим означенням, пов’язаним із терміном «персептрон», є термін «елементарний персептрон». Елементарний персептрон – це простий персептрон, що має лише один асоціативний шар. Відповідно до точки зору Розенблата, елементарний персептрон це мінімальна система, здатна виконувати функції розпізнавання вхідних сигналів.



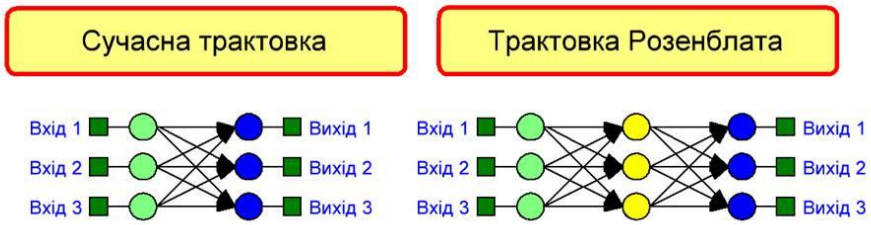
Проте сучасні фахівці із цією думкою не завжди погоджуються, і наразі подібні системи вони класифікують як одношарові та багатошарові персептрони. В одношарових персептронах нейрони асоціативного та вирішувального шарів об’єднані. Тобто така система фактично має лише один – вихідний – шар, у якому приймаються рішення. Така комбінація стає можливою, якщо у простому персептроні замість порогових функцій активації використовувати в асоціативних елементах лінійні – у цьому випадку від асоціативних шарів можна повністю відмовитись і приймати рішення лише в одному вирішувальному шарі.

Сучасна класифікація персептронів



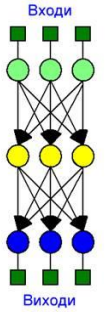
Слід зауважити, що сам Розенблат вважав, що наявність асоціативного шару є обов'язковою для розпізнавання комбінацій вхідних сигналів, тому він під одношаровим персептроном мав на увазі персептрон із одним прихованим шаром – мережу, яка зараз позиціонується як багатошарова, а точніше – двошарова.

Одношаровий персептрон



Таким чином, незважаючи на певну плутанину в термінології можна визначити, що зараз під персептронами зараз розуміють доволі прості нейронні мережі із прямим розповсюдженням інформації, які не мають будь-яких зворотних зв'язків ні всередині, ні ззовні. Крім того, персептрони, зазвичай, мають дискретні вихідні сигнали усіх нейронів, які можуть приймати значення або 0, або 1. Напевно, ці ознаки є головними при визначенні того, чи є ця система персептроном, чи ні. Тому їх доведеться запам'ятати, оскільки надалі вони у будь-який час можуть знадобитися.

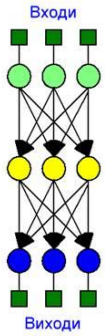
Сучасні ознаки персептрона



- Мережа із прямим поширенням інформації
- Вихідні сигнали усіх нейронів є дискретними (або 0 або 1)
- Елементи вхідного шару (S-елементи) не приймають участі в розпізнаванні сигналу
- Передача сигналу між нейронами відбувається миттєво, або на протязі фіксованого інтервалу часу

Незважаючи на свою простоту, персептрони мають певне практичне значення. Ви вже знаєте, що їх можна використовувати для розпізнавання образів, зокрема літер або цифр. Проте область їх застосування цим не обмежується. Наразі персептрони активно використовують для апроксимації функцій, для вирішення задач прогнозування, для керування агентами, та у багатьох інших застосунках.

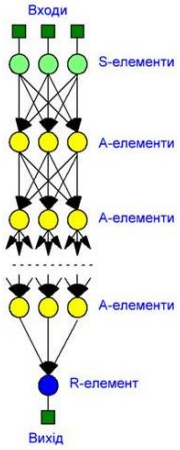
Використання персептронів



- Розпізнавання образів
- Класифікація об’єктів
- Апроксимація функцій
- Прогнозування поведінки об’єктів і систем
- Керування агентами

Персептрони Румельхарта

Персептрони, запропоновані Розенблатом, для нейронних мереж стали своєрідним локомотивом. Тому немає нічого дивного у появі великої кількості їх модифікацій. Однією із них стали багатошарові персептрони Румельхарта, які зовні нічим не відрізняються від оригінальних багатошарових мереж, запропонованих Розенблатом. Проте, незважаючи на зовнішню схожість, персептрони Румельхарта мають кілька суттєвих відмінностей, які розширюють сферу їх практичного застосування.



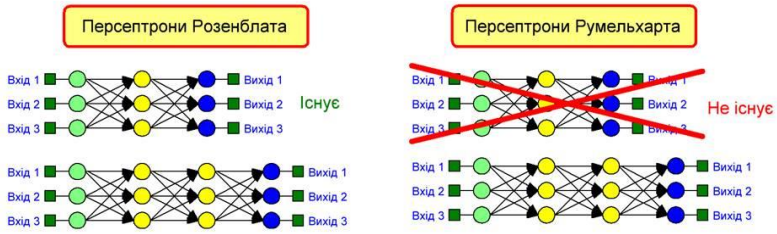
Персептрон Румельхарта

Відмінності від персептрона Розенблата

- Використання сигмоїдної функції активації
- Більше одного асоціативного шару
- Використання неперервних вхідних та вихідних сигналів
- Довільна архітектура зв'язків
- Використання інших методів навчання

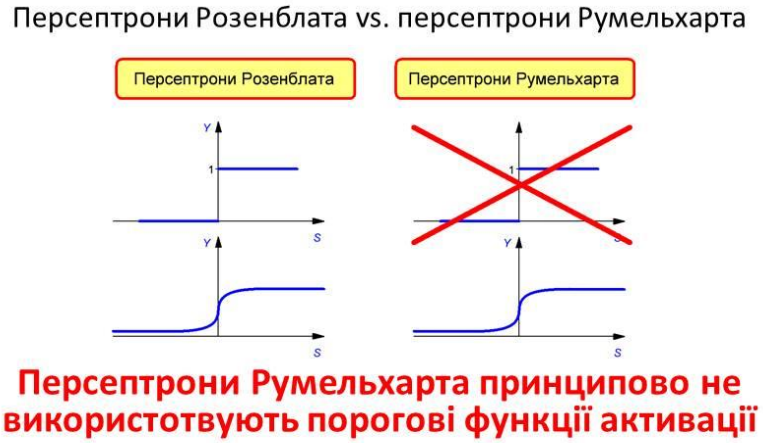
По-перше, усі персептрони Румельхарта мають більше одного асоціативного шару, тобто вони принципово є багатошаровими. Якщо персептрони Розенблата можуть мати тільки один асоціативний шар, то у персептронах Румельхарта їх обов’язково повинно бути декілька. Проте ця ознака не є принциповою, оскільки багатошарові персептрони, тобто персептрони, що мають більше одного асоціативного шару, використовували у своїх дослідженнях і Розенблат, і Румельхарт, і ще велика кількість інших дослідників штучного інтелекту.

Персептрони Розенблата vs. персептрони Румельхарта

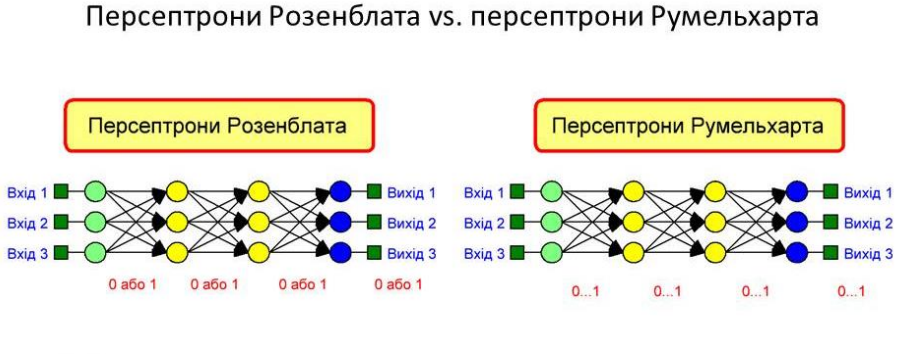


Персептрони Румельхарта принципово мають більше одного асоціативного шару

Більш істотною відмінністю персептронів Розенблата від персептронів Румельхарта є типи функцій активації. Розенблат у свої системах, зазвичай, використовував порогові функції активації, що призводило до дискретизації сигналів, із якими працює нейронна мережа. Румельхарт у своїх системах порогові функції активації не використовував принципово. Усі функції активації усіх нейронів у персептронах Румельхарта є сигмоїдними. А вони, як ви пам'ятаєте, можуть забезпечити неперервні сигнали, у той час як порогові функції – тільки дискретні.

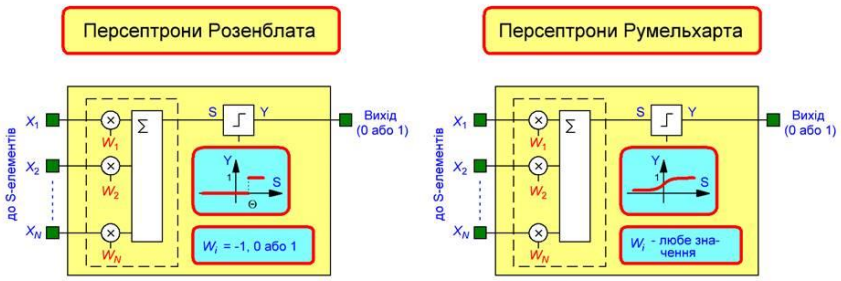


Використання сигмоїдних функцій активації дозволило зняти обмеження на рівні сигналів, що використовуються в мережі. Персептрони Розенблата в оригінальній версії працюють лише із дискретними сигналами, що можуть приймати значення або 0, або 1. А от персептрони Румельхарта працюють тільки із неперервними сигналами, що можуть приймати будь-яке значення, хоч і в обмеженому діапазоні. В оригінальних роботах Румельхарт використовував той же самий діапазон від 0 до 1, тобто також працював із нормованими сигналами. Проте, ще раз зверну увагу, що сигнали в персептронах Румельхарта є безперервними і можуть приймати будь-яке значення у діапазоні від 0 до 1.



Перехід до неперервних сигналів дозволив зняти обмеження на діапазон значень вагових коефіцієнтів у нейронах асоціативних шарів. В персептронах Розенблата вагові коефіцієнти цих елементів могли приймати лише фіксовані значення: –1, 0 або 1. Крім того, ці коефіцієнти були сталими тобто не змінювались в процесі навчання мережі. У персептронах Румельхарта вагові коефіцієнти елементів асоціативних шарів можуть приймати будь-яке значення, при цьому вони змінюються у процесі навчання мережі. Тобто у нейронних мережах Румельхарта, фактично, зникає різниця між елементами асоціативного та вирішувального шарів. У мережах Румельхарта ці нейрони є однаковими, у той час як у мережах Розенблата нейрони асоціативного шару, незважаючи на структурну еквівалентність із нейронами шару, у якому приймається остаточне рішення, мають низку відмінностей у значені вагових коефіцієнтів.

Персептрони Розенблата vs. персептрони Румельхарта



У персептронах Румельхарта вагові коефіцієнти елементів асоціативних шарів можуть приймати будь яке значення і змінюватися у процесі навчання мережі

Румельхарт також зняв принципове обмеження на ступінь зв'язків всередині мережі. В персептронах Розенблата зв'язки можуть бути тільки між сусідніми шарами, тобто виходи елементів вхідного шару з'єднуються тільки із входами елементами першого асоціативного шару, виходи елементів першого асоціативного шару з'єднуються лише із входами елементів наступного асоціативного шару, ну і так далі – до шару вирішувальних елементів. В персептронах Румельхарта нейрони можуть зв'язуватися між собою яким завгодно способом, за умови підтримання одного напрямку розповсюдження інформації, тобто в персептронах Румельхарта допустимі любі типи зв'язків, окрім зворотних.

Персептрони Розенблата vs. персептрони Румельхарта

Персептрони Розенблата

- Навчання методом корекції помилки

Персептрони Румельхарта

- Навчання методом зворотного розповсюдження помилки

Персептрон Румельхарта буде мати функціональні перевагами, порівняно із персептроном Розенблатта, лише в тому випадку, якщо буде не просто реагувати на певні комбінації вхідних сигналів, а якщо буде спроможний самостійно створювати відповідні реакції

Але головною особливістю персептронів Румельхарта є метод навчання нейронної мережі. У персептронах Розенблата використовується метод корекції помилки, у той час як у персептронах Румельхарта – метод зворотного розповсюдження помилки. Ці методи ми будемо докладно вивчати дуже скоро, а зараз потрібно лише запам'ятати, що ці нейронні мережі дуже схожі зовні, проте дуже різні всередині.

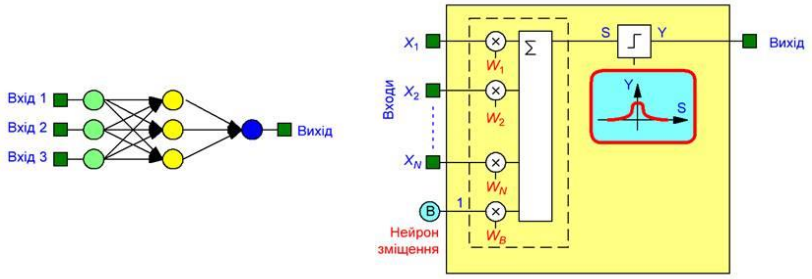
Персептрони Румельхарта можуть мати певні функціональні перевагами, порівняно із персептронами Розенблатта. Ці переваги проявляються не тільки у кращому розпізнавання комбінації вхідних сигналів – персептрони Розенблата це також вміють добре робити. Головною особливістю персептронів Румельхарта є теоретична здатність до самостійного навчання, у той час як персептрони Розенблата потрібно штучно навчати. Проте тема самостійного навчання нейронних мереж, навіть незважаючи на «пристойний вік» подібних систем, ще й досі потребує докладного вивчення.

Нейронні мережі із радіально-базисними функціями

Серед нейронних мереж із прямим поширення інформації окремої уваги заслуговують нейронні мережі із радіально-базисними функціями. З першого погляду ці мережі також мало чим відрізняються від розглянутих перед цим персептронів Розенблата та Румельхарта. Вони так само мають один вхідний шар, один прихований шар і один вихідний шар, нейрони якого приймають остаточне рішення по ідентифікації вхідних сигналів.

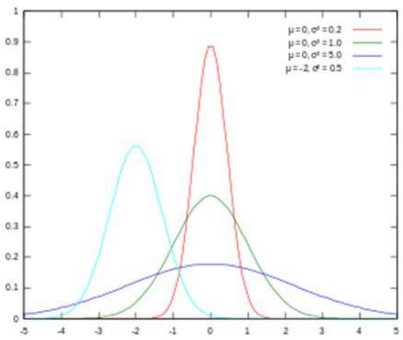
Головною особливістю мереж цього типу є функції активації, які використовуються в нейронах прихованого та вихідного шарів. У цих мережах замість монотонно-зростаючих сигмоїдних функцій або порогових функцій, що також збільшуються лише в одному напрямку, використовується спеціальний клас симетричних функцій, які називаються радіально-базисними. Головною особливістю цих функцій є те, що вони мають чіткий максимум при певному значенні вхідного аргументу, у якості якого, так само як і в розглянутих вище персептронах Розенблата та Румельхарта використовується сума вхідних сигналів, помножених на відповідні вагові коефіцієнти.

Нейронна мережа із радіально-базисними функціями



Наразі відома доволі велика кількість радіально-базисних функцій – функцій, що мають певний максимум, і швидко зменшуються майже до нуля при відхиленні від нього. Найбільш поширеною серед них є функція Гауса, графік якої нагадує дзвін. Вона має три головні параметри – амплітуду A, що визначає її максимальне значення, позицію максимуму на горизонтальній вісі, за яку відповідає параметр b, та параметр c, який визначає як швидко значення цієї функції буде зменшуватись при відхиленні аргументу від значення, що відповідає максимуму, тобто відповідає за ширину «дзвона». В нейронних мережах, що, зазвичай, працюють із нормованими сигналами, максимальне значення активаційної функції, як правило, дорівнює одиниці. Позиція максимуму особливої ролі не має, оскільки комбінацію вхідних сигналів завжди можна змістити в потрібну область за допомогою вагового коефіцієнта, пов'язаного із нейроном зміщення. Таким чином, під час навчання нейронної мережі головною задачею є пошук потрібного значення коефіцієнта c.

Функція Гауса



$$y(x) = Ae^{-\frac{(x-b)^2}{2c^2}}$$

A – амплітуда

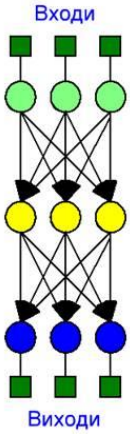
b – позиція центру

c – ширина «активної частини»

На відміну від персептронів Розенблата та Румельхарта, які мають велику кількість різновидів та широку область застосування, нейронні мережі на основі радіально-базисних функцій мають набагато меншу кількість варіантів побудови. Зазвичай, вони використовуються лише у базовому варіанті – коли мережа має лише три шари: вхідний, прихований, та вихідний. Крім того, у нейронних мережах на основі радіально-базисних функцій використовується доволі специфічний алгоритм навчання, який не використовується більше ніде.

Нейронні мережі із радіально-базисними функціями краще за все використовувати для апроксимації функцій. Використання радіально-базисних функцій активації нейронів прихованого на вихідних шарів дозволяє навчити мережу краще і головне – набагато швидше, ніж при використанні традиційних персептронів. Крім того, подібні нейронні мережі можна використовувати із в інших застосунках, у тому числі і для вирішення задач прогнозування, керування та класифікації. Слід зазначити, що нейронні мережі на основі радіально-базисних функцій є відносно новим класом систем. Вони вперше були запропоновані Брумхедом та Девідом Лоу у 1988 році, тому наразі їх можливості та обмеження ще до кінця не визначені.

Нейронна мережа із радіально-базисними функціями



Особливості

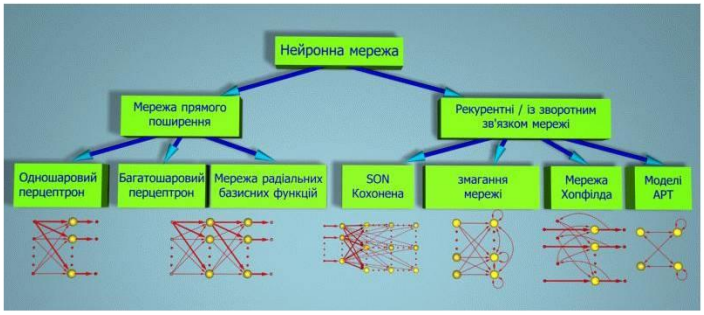
- Менша кількість варіантів
- Використання специфічного алгоритму навчання

Використання

- Апроксимація функцій
- Прогнозування часових рядів
- Класифікація об’єктів
- Керування об’єктами та системами

Отже сьогодні ми познайомилися лише із одним класом нейронних мереж – із мережами із прямим розповсюдженням інформації. Слід зауважити, що цей клас є, напевно, найкраще за всі інші класи вивченим та дослідженим, тому, незважаючи на невелику кількість представників, можлива кількість варіантів побудови нейронних мереж із прямим поширенням інформації майже безмежна. Так само, як і область практичного застосування мереж цього типу, напевно, є найширшою. Крім того, не слід забувати, що саме мережі із прямим розповсюдження інформації свого часу дали не тільки поштовх для розвитку нейронних мереж, а й дозволили почати роботу над створенням штучних інтелектуальних систем, які, можливо, колись за рівнем інтелекту зможуть перевищити навіть людину.

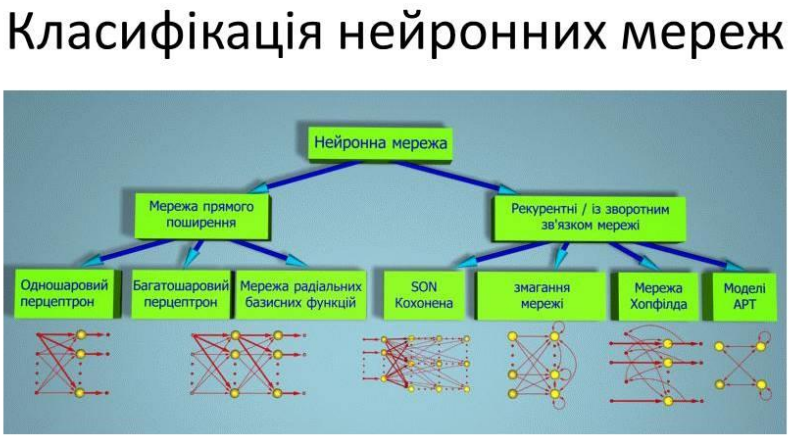
Класифікація нейронних мереж



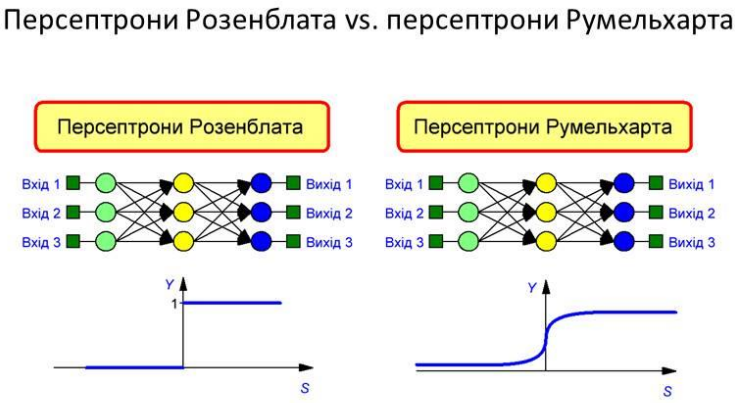
Ну а з нейронними мережами із зворотними зв’язками ми познайомимося, напевно, вже на наступній лекції. Ну а на сьогодні я вам дякую за увагу і маю надію, що ви зробити ще один крок у створенні як власного майбутнього, так і майбутнього всього людства.

Лекція 5. Зворотний зв'язок в нейронних мережах

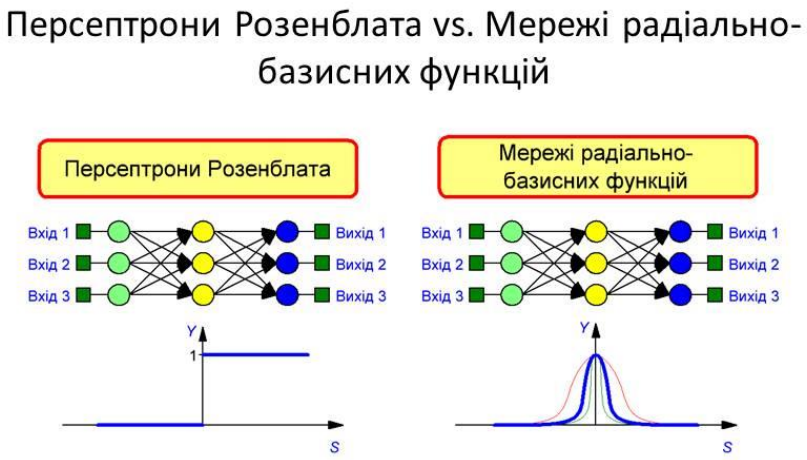
На попередній лекції ми з вами познайомитися із класифікацією нейронних мереж, відповідно до якої усі нейронні мережі поділяються на дві великі групи: мережі із прямим поширенням інформації та мережі із зворотними зв'язками. Ми докладно розібрали принципи побудови мереж першої групи і з'ясували, більшість із них називаються персептронами. Ми дуже докладно вивчили персептрони Розенблата, і це не випадково, оскільки ці вони історично стали першими нейронними мережами, які людина змогла практично реалізувати, і тому ці системи дуже добре вивченні і досліджені.



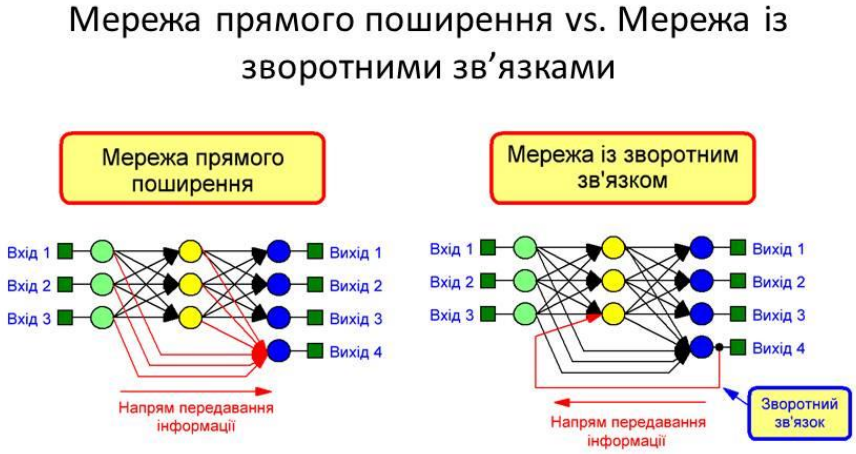
Більше того, багато особливостей персептронів Розенблата притаманні і іншим нейронним мережам. Наприклад, ви пам'ятаєте, що персептрони Румерхальта дуже схожі на персептрони Розенблата, але в них замість дискретних сигналів, що можуть приймати значення лише 0 або 1, використовуються неперервні сигнали, що можуть приймати будь яке значення, хоч і в тому ж самому діапазоні: від 0 до 1. Ви також пам'ятаєте, що досягається це, у першу, чергу, шляхом заміни порогових функцій активації їх неперервними аналогами, наприклад, сигмоїдними функціями.



Не є винятком і мережі на основі радіально-базисних функцій, у яких вихідні сигнали нейронів, на відміну від персептронів Розенблата та Румерхальта, стають максимальними лише при певному значенні вагової суми і швидко зменшуються, якщо вона відрізняється від цієї величини, причому як в більшу, так і в меншу сторони.



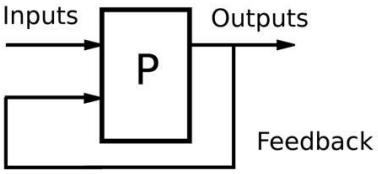
Та головною ознакою усіх розглянутих перед цим мереж є те, що інформація в них поширювалася тільки в одному напрямку – від входу до виходу, і винятків від цього правила, зазвичай, не було. Настав час познайомитися із мережами, що мають зворотні зв’язки між нейронами, тобто зв’язки, по яким інформація поширюється у зворотному напрямку – від виходу нейрона, до його входу. Але перед тим як перейти до розгляду нейронних мереж із зворотними зв’язками, слід спершу згадати, що таке зворотний зв’язок, яким він буває, де використовується, і головне – навіщо його потрібно застосовувати у нейронних мережах.



Зворотний зв’язок

Під терміном «зворотний зв'язок» (англ. feedback) мається на увазі забезпечення можливості впливу результату функціонування будь-якої системи на характер її подальшого функціонування. Термін «зворотний зв'язок» використовують у соціальних, біологічних, технічних, економічних та інших системах, а також у кібернетиці та теорії автоматичного регулювання. У техніці, у тому числі і в нейронних мережах, зворотний зв’язок формується шляхом подачі сигналу з виходу якогось вузла, блока, чи навіть цілої системи, на вхід або того ж самого, або іншого вузла чи блока – головне, щоб цей вузол чи блок був частиною тієї ж самої системи, а сигнал з кола зворотного зв’язку міг впливати на результат її роботи.

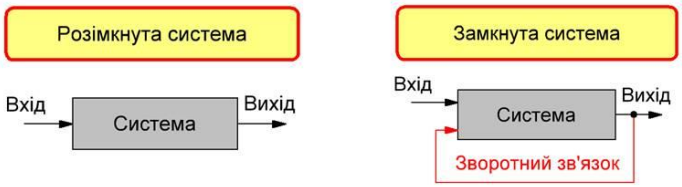
Зворотний зв’язок (Feedback)



Зворотний зв'язок» – забезпечення можливості впливу результату функціонування будь-якої системи на характер її подальшого функціонування

В технічній літературі, зокрема в теорії автоматизованого керування, системи, які не мають зворотного зв’язку, називаються розімкнутими системами, системи, що мають зворотний зв’язок, відповідно, замкнутими системами. У розімкнутих системах, тобто у системах, що не мають зворотних зв’язків, вихідні сигнали залежать лише від комбінації вхідних сигналів і, відповідно, від характеристик самої системи. А от у системах із зворотними зв’язками, тобто у замкнутих системах, вихідні сигнали ще додаткового залежать від поточного стану самої системи.

Розімкнута vs. Замкнута система

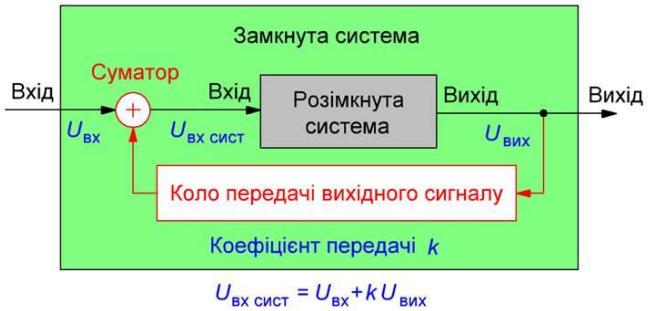


Залежність вихідних сигналів:
- Вхідні сигнали
- Параметри системи

Залежність вихідних сигналів:
- Вхідні сигнали
- Параметри системи
- **Стан системи**

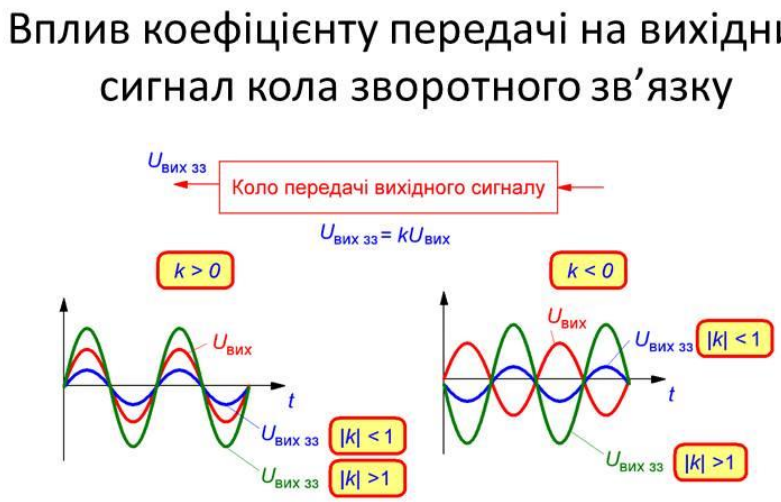
Любу розімкнуту систему можна перетворити на замкнуту можна шляхом додавання до неї додаткових вузлів, що будуть забезпечувати зворотний зв’язок, а саме – кола, що передає сигнал із виходу на вхід з певним коефіцієнтом передачі, та суматора, який розташовується на вході і об’єднує вхідний сигнал із сигналом зворотного зв’язку. Функція суматора є зрозумілою – він виконує просту арифметичну операцію додавання. А от із колом передачі вихідного сигналу усе непросто.

Найпростіший спосіб перетворення розімкнутої системи у замкнуту



$$U_{вх\ сист} = U_{вх} + k\,U_{вих}$$

У найпростішому випадку, цей вузол має один головний параметр, який називається коефіцієнт передачі. У загальному випадку, коефіцієнт передачі може бути статичним, тобто незмінним і незалежним ні від чого, а може залежати від якихось інших параметрів, наприклад, від частоти. Якщо коефіцієнт передачі більший за одиницю, то сигнал, що проходить через коло зворотного зв'язку, підсилюється і його амплітуда стає більшою за амплітуду сигналу, присутньому на виході системи і, відповідно, на вході кола зворотного зв'язку. Але частіше за все, коефіцієнт передачі цього вузла менший за одиницю. А це означає, що сигнал, що проходить через коло зворотного зв'язку, послаблюється і його амплітуда стає меншою за амплітуду первинного сигналу. А от варіант, коли цей коефіцієнт дорівнює нулю ми не розглядаємо, оскільки, у даному випадку, ніякого зворотного зв'язку немає, і система стає розімкненою. Отже, абсолютне значення коефіцієнту передачі кола зворотного зв'язку може бути або більшим, або меншим за одиницю, що відповідає, відповідно, послабленню або посиленню вихідного сигналу.

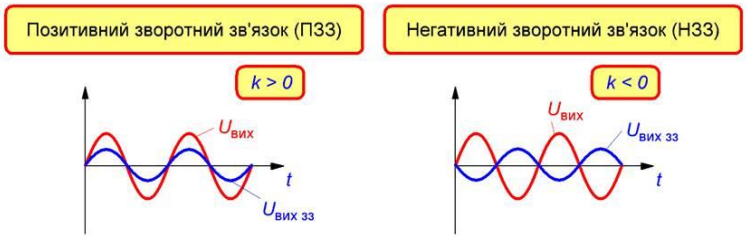


Але тут можуть бути і інші варіанти. Зокрема, коефіцієнт передачі може бути ще й позитивним або негативним, тобто меншим або більшим за нуль. Що це означає? Це означає, що коло зворотного зв'язку може інвертувати сигнал, який подається на його вхід. У цьому випадку, якщо на вході кола зворотного зв'язку із негативним коефіцієнтом передачі присутній позитивний сигнал, то на його виході сигнал буде негативним, і навпаки – при подачі негативного сигналу сигнал на виході стане позитивним. І знову ж таки, якщо абсолютне значення, тобто модуль, коефіцієнта передачі є більшим за одиницю, то сигнал буде підсилюватися, а якщо меншим за одиницю – то послаблюватися. Тобто, абсолютне значення, або модуль, коефіцієнта передачі впливає на амплітуду сигналу, що проходить через коло зворотного зв'язку, а його знак – на фазу.

Абсолютне значення коефіцієнту передачі кола зворотного зв'язку нас наразі мало цікавить, оскільки воно показує лише «силу» зворотного зв'язку, тобто наскільки сильно вхід системи зв'язаний з її виходом. А от знак коефіцієнту передачі нас цікавить набагато більше, оскільки він змінює характер поведінки усієї системи.

Зрозуміло, що тут можуть бути лише два варіанти: варіант, коли коефіцієнт передачі більше нуля, і варіант, коли менше нуля. Знову ж таки нагадаю, що варіант, коли коефіцієнт передачі дорівнює нулю, ми взагалі не розглядаємо, оскільки у цьому випадку ніякого зворотного зв'язку, і система стає типовою розімкненою системою. Отже, якщо коефіцієнт передачі кола зворотного зв'язку більше нуля, то такий зворотний зв'язок називають позитивним, а якщо менше нуля, то, відповідно, негативним.

Варіанти зворотного зв’язку



У технічних системах позитивний та негативний зворотний зв’язки використовують приблизно однаково. Але практичне застосування різних типів зворотного зв’язку різне. Негативний зворотний зв’язок використовують, у більшості випадків, для стабілізації певних процесів. Зокрема негативний зворотний зв’язок майже завжди використовують у системах автоматизованого керування, головною метою яких є підтримання якихось параметрів у заданих межах. А от позивний зворотний зв’язок, навпаки, використовують у випадках, коли систему необхідно вивести із певного стану – частіше за все стану невизначеності, тому позитивний зворотний зв’язок використовують, наприклад, в автогенераторах. Його використовують також у пристроях, де повинні протікати ланцюгові реакції, наприклад, у лазерах або ядерних реакторах, а також у системах прийняття рішень.

Застосування зворотного зв’язку в техніці

Позитивний зворотний зв’язок

- Автогенератори
- Системи із ланцюговими реакціями
- Гістерезисні системи
- Системи прийняття рішень

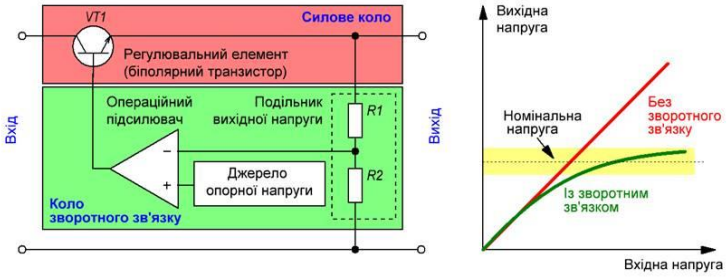
Негативний зворотний зв’язок

- Стабілізатори
- Підсилювачі
- Автоматичні регулятори

Одним із яскравих прикладів використання негативного зворотного зв’язку в електроніці є компенсаційні стабілізатори напруги. У цих вузлах силова частина, тобто коло передачі енергії, організована регулювальним елементом, у якості якого використовують біполярні або польові транзистори. Коло негативного зворотного зв’язку, головним призначенням якого є підтримання необхідного рівня напруги на виході стабілізатора, має три головні вузли: подільник вихідної напруги, операційний підсилювач та джерело опорної напруги. Коли напруга на виході стабілізатора зменшується, то це призводить до зменшення різниці напруг на входах операційного підсилювача, оскільки напруга на його інвертуючому вході є пропорційною вихідної напрузі, а от напруга на неінвертуючому вході підтримується постійною джерелом опорної напруги. Зменшення різниці напруг на входах операційного підсилювача призводить до збільшення напруги на його виході. Це призводить до зменшення внутрішнього опору регулювального елементу, а це, у свою чергу, призводить до збільшення вихідної напруги. Якщо ж напруга на виході стабілізатора стає більшою за номінальну напругу, то все відбувається навпаки – напруга на інвертуючому вході операційного підсилювача збільшується, напруга на виході операційного підсилювача зменшується, внутрішній опір регулювального елемента збільшується, що, у кінцевому рахунку,

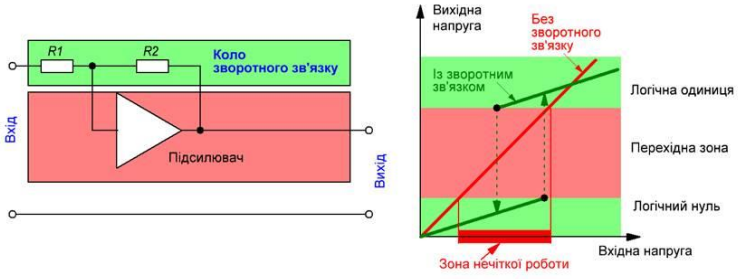
призводить до зменшення вихідної напруги. Таким чином, подібна система дозволяє автоматично підтримувати вихідну напругу в заданих межах при любих її відхиленнях. Причому неважливо, що стало причиною відхилення напруги на виході стабілізатора: коливання вхідної напруги, коливання температури, коливання струму навантаження або якихось інших параметрів.

Принцип роботи стабілізатора напруги
(система з негативним зворотним зв'язком



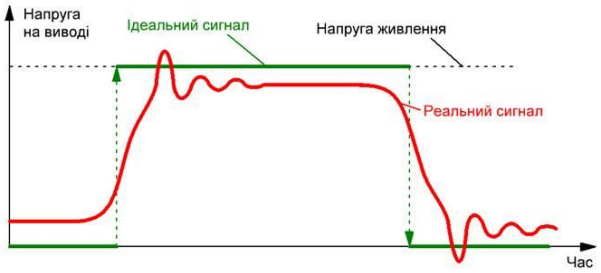
Яскравим прикладом застосування позитивного зворотного зв'язку є тригери Шмідта, які широко використовуються в цифровій техніці, зокрема, у вхідних колах портів вводу/виводу мікроконтролерів. Незважаючи на те, що мікроконтролери, мікропроцесори та інші цифрові елементи працюють із дискретними сигналами, насправді, усі вони базуються на аналогових вузлах. Отже, в основі усіх дискретних сигналів лежать сигнали, які відносяться до категорії неперервних. Таким чином, в реальних цифрових системах завжди виникає задача перетворення аналогового сигналу, тобто сигналу, який може приймати будь-яке значення, у дискретний сигнал, тобто сигнал, що має можливість приймати лише певні фіксовані значення.

Принцип роботи тригера Шмідта
(система з позитивним зворотним зв'язком



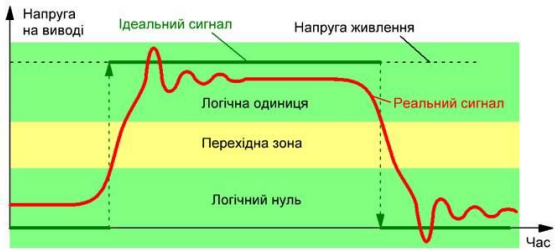
В цифровій електронній техніці використовується двійкова система числення, яку простіше за все реалізувати на практиці У цьому випадку, за логічну одиницю приймають факт наявності напруги на певному виводі, а за логічний нуль – відповідно, факт її відсутності. Проте у реальних електронних колах, через неідеальність елементів та особливостей схемотехніки, ніколи не буває ситуації, коли напруга на якомусь виводі повністю відсутня – отже, коли електронна схема працює, то напруга на кожному з її виводів завжди буде більша за нуль. А ще на жодному з виводів також дуже складно встановити напругу більшу за максимальну напругу яка присутня у системі, тобто напругу живлення. Отже, напруга в любій точці цифрової системи, майже завжди займає якесь проміжне значення у діапазоні від нуля до напруги живлення.

Реальні аналогові сигнали в дискретних цифрових системах



Але нам потрібно якось інтерпретувати цю напругу, бо подальше програмне забезпечення повинно працювати лише із дискретними сигналами. У даному випадку вводять два діапазони напруг. Якщо реальна напруга знаходиться в одному діапазоні, то вважають, що сигнал має рівень логічного нуля, а якщо в іншому – то рівень логічної одиниці. Але навіть для самої найкраще спроектованої системи існує певний діапазон напруг, де вона буде працювати нечітко. І якщо напруга буде знаходитись у цьому діапазоні, то сигнал може бути інтерпретований як завгодно, і як логічний нуль, і як логічна одиниця. Причому різні екземпляри однієї і тієї ж моделі мікропроцесора або мікроконтролера через неминучі технологічні відхилення одну і ту ж напругу будуть інтерпретувати по-різному.

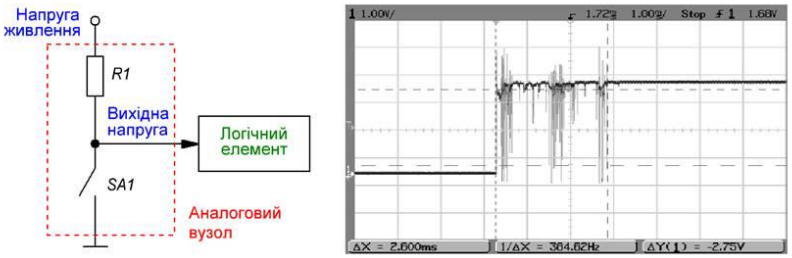
Реальні аналогові сигнали в дискретних цифрових системах



Більш складна ситуація виникає, коли цифровий сигнал починає змінюватися. Через наявність в реальній електронній схемі паразитних ємностей та індуктивностей, напруга в електричних колах ніяк не може миттєво змінити своє значення. Отже зміна сигналу з нуля в одиницю і навпаки відбувається не миттєво, як в ідеальному випадку, а займає певний час, на протязі якого сигнал обов'язково попаде у проміжну зону де він може визначатися і як нуль, і як одиниця. Причому, через високий коефіцієнт підсилення цифрових елементів, коли вхідний сигнал знаходиться у зоні невизначеності, для зміни дискретного рівня сигналу достатньо найменшого шуму.

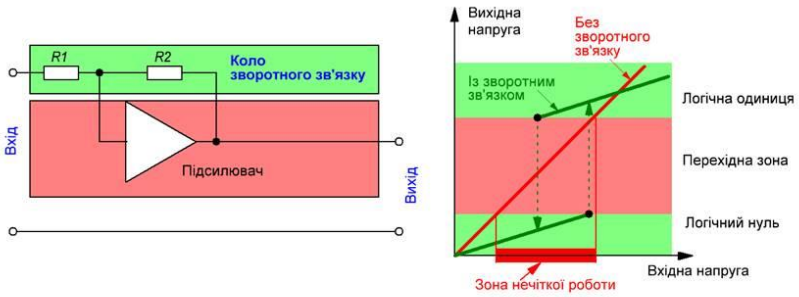
Ця проблема є аналогічною проблемі «брязкоту контактів». Із брязкотом контактів стикаються усі розробники електронної апаратури, у яких в інтерфейсній частині використовуються кнопки, тумблери або інші механічні перемикачі. Проблема полягає у тому, що опір механічних контактів під час замикання або розмикання зміється хаотично, що призводить до того, логічний елемент, підключений до такого приймача, за одне натискання на кнопку може кілька разів змінити рівень логічного сигналу з нуля до одиниці і навпаки, тому, якщо не приймати відповідних заходів, то така інтерпретація вхідних сигналів призведе до неправильної роботи усієї системи. Наприклад, ви єдиному натисканні на якусь кнопку, а мікроконтролер, не обладнаний необхідними елементами для усунення брязкоту контактів, прийняв рішення, що кнопку натиснули кілька разів.

Проблема брязкоту контактів



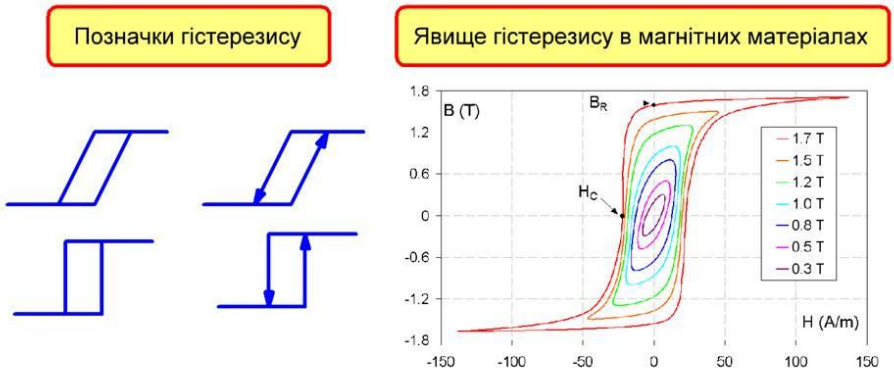
Для запобігання цьому і використовуються вузли із позитивним зворотним зв'язком, зокрема тригери Шмідта. Тригер Шмідта являє собою неінвертуючий підсилювач, охоплений позитивним зворотним зв'язком. Зворотний зв'язок реалізується да допомогою резисторного подільника напруги, який частину вихідного сигналу без будь-якої інверсії подає на його вхід. Якщо рівень вхідного сигналу менший за певний пороговий рівень, то підсилювач формує сигнал, що відповідає рівню логічного нуля. Як тільки вхідний сигнал стає більшим за певне порогове значення, то сигнал на виході підсилювача починає збільшуватися, що призведе і до збільшення сигналу зворотного зв'язку, а це, у свою чергу, призведе до додаткового збільшення вхідного сигналу і, відповідно, і до іще більшого збільшення вихідного. В результаті цього лавиноподібного процесу сигнал на виході підсилювача дуже швидко – майже миттєво – зміниться з рівня логічного нуля до рівня логічної одиниці. При цьому збільшиться і рівень порогового значення, і тепер, для того, щоб перевести тригер Шмідта у попередній стан – коли рівень вихідного сигналу дорівнює нулю – необхідно щоб напруга на вході цього вузла стала меншою за напругу, яка призвела до переводу тригера у стан з високим рівнем вихідного сигналу. Отже при використанні позитивного зворотного зв'язку система принципово не може залишитися у зоні невизначеності або у зоні нечіткої роботи і швидко переходить із одного фіксованого стану до іншого.

Принцип роботи тригера Шмідта (система з позитивним зворотним зв'язком



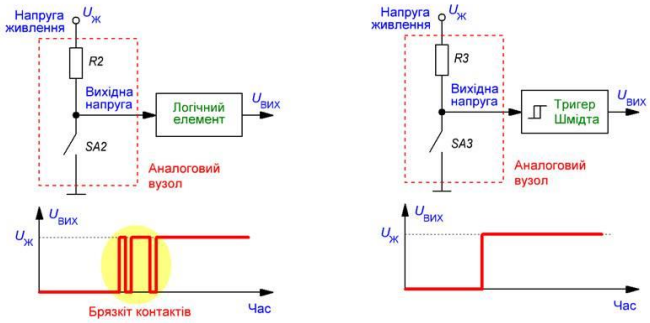
Це явище у техніці називається гістерезисом, який позначається відповідним значком. Гістерезис притаманний не тільки електронним схемам – він може бути присутнім і на фізичному рівні. Наприклад, майже усі феромагнітні матеріали мають гістерезис, проте в феромагнітних матеріалах для його реалізації нічого не потрібно робити, а у технічних системах для реалізації гістерезису необхідно застосовувати позитивний зворотний зв'язок.

Гістерезис



Таким чином, позитивний зворотний зв'язок дозволяє уникнути невизначеності. У випадку брязкоту контактів, тригери Шмідта, тобто системи із позитивним зворотним зв'язком, у комбінації із елементами, що не дозволяють швидко змінюватися напрузі на механічних контактах, дозволять сформувати чіткий логічний сигнал з яким система буде нормально працювати.

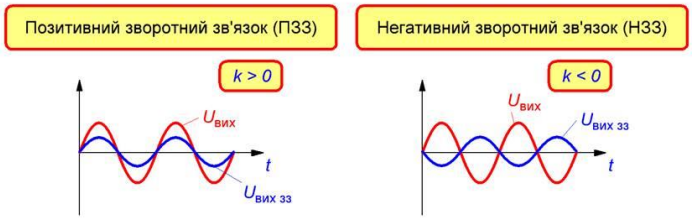
Усунення брязкоту контактів за допомогою тригера Шмідта



Використання зворотного зв'язку в нейронних мережах

Отже, ви напевно вже зрозуміли, зо зворотний зв'язок – це дуже хороша річ, яка активно використовується в технічних пристроях. Тепер залишилось з'ясувати, навіщо потрібно було наводити стільки прикладів використання і позивного, і негативного зворотного зв'язку, причому у системах, які, здається, не мають жодного відношення до нейронних мереж.

Варіанти зворотного зв'язку



Але річ у тому, що саме поняття зворотний зв'язок виходить далеко за межі електроніки чи комп'ютерної техніки. Зворотний зв'язок існує у великій кількості систем, які не

обмежуються тільки технічними засобами. Навіть у навчанні використовується зворотний зв’язок – коли ви присилаєте свої роботи викладачу, то викладач починає коригувати свій курс для того, щоб врахувати ваші індивідуальні особливості і довести вас до певного рівня. У даному випадку цей зв’язок є негативним. Але негативним не у сенсі, що він є неприємним для викладача, а у сенсі стабілізації процесу навчання, оскільки негативний зворотний зв’язок дозволяє отримати бажаний результат при будь-яких відхиленнях початкових даних – у даному випадку – вашого поточного рівня знань та вмінь. А тепер давайте з’ясуємо як зворотний зв’язок дозволить покращити наші нейронні мережі.

Де існує зворотний зв’язок

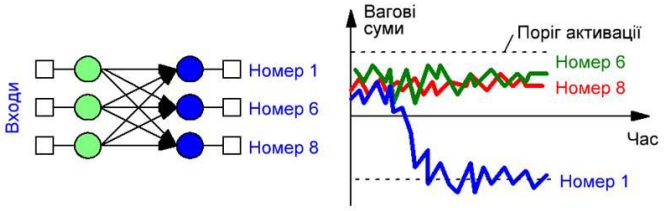
- Біологічні системи
- Хімічні реакції
- Математичні системи
- Динамічні системи
- Кліматичні процеси
- Системи автоматизованого керування
- Навчання
- Програмне забезпечення

Розглянемо конкретний приклад. Ми стоїте на зупинці громадського транспорту і чекаєте, наприклад, на тролейбус. По цій дорозі їздять тролейбуси по маршрутам номер 1, 6 та 8, і вам необхідний тролейбус номер 6. Зображення цифри 1 доволі сильно відрізняється від зображення цифр 6 та 8, тому її можна розпізнати із більшої відстані ніж у інших випадках. А от цифри 6 та 8 доволі схожі, тому їх розпізнати важче, особливо коли умови для розпізнавання не дуже сприятливі.

Розпізнавання номеру тролейбуса

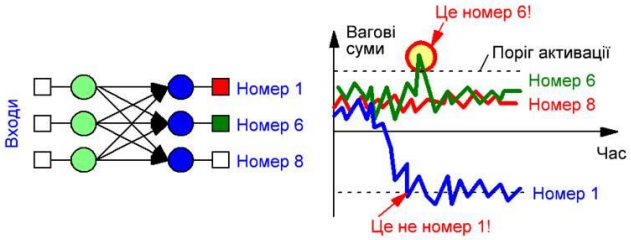
І от ви бачите, що під’їжджає тролейбус і бачите, що його номер не схожий на номер 1 а більше схожий на номер 6 або 8. Зверніть увагу, що від того, приїде «ваш» тролейбус, чи не «ваш», залежить ваша подальша поведінка – чи будете ви далі стояти на зупинці і відійдете в сторону, щоб не заважати іншим людям, що почнете готуватися до посадки. Тобто, наразі ваш мозок працює як система прийняття рішень, яке вам доведеться приймати обов’язково. Отже ви поки що погано бачите, який тролейбус під’їжджає: із номером 6 або номером 8. Це означає, що ваша внутрішня нейронна мережа знаходиться у нестабільному стані, хоча сигнал нейрон, що відповідає за розпізнавання цифри 1, має негативне значення, тобто ви точно бачите не цифру 1. А от сигнали з нейронів, що відповідають за розпізнавання цифр 6 та 8, приблизно однакові і ви все ще не можете прийняти рішення, що далі робити.

Розпізнавання номеру тролейбуса



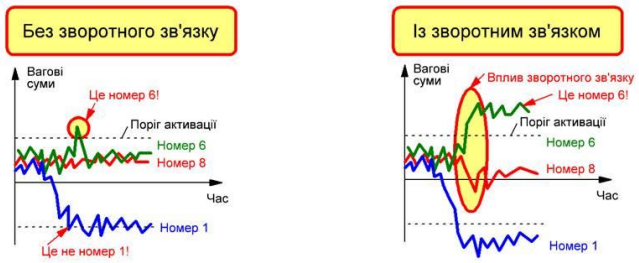
Та ось, на якусь долю секунди ви чітко побачили табличку тролейбусу і розпізнали, що він має номер 6. Отже, вам потрібно готуватися до посадки і, наприклад, дістати гроші за проїзд. Зверніть увагу, що ви чітко побачили номер лише на якусь мить і, можливо, ви навіть до кінця не впевнені, що цей номер є номером 6. Проте імовірність, що їде саме цей тролейбус, найвища. Що буде відбуватися далі? У реальному житті ви, навіть незважаючи на неповну впевненість, перестанете дивитися на табличку тролейбуса і почнете готуватися до посадки. Тобто ви вже прийняли рішення і десь у голові собі його зафіксували.

Розпізнавання номеру тролейбуса



Приблизно так само повинна спрацювати і штучна нейронна мережа. Проте нейронна мережа без зворотних зв’язків, наприклад, персептрон Розенблата, ніколи не сформує таку реакцію. Так, у певні моменти часу вагові суми певних нейронів перевищать порогові значення, проте це буде лише на якусь мить. А вам необхідно, що система знаходилась у стабільному стані на протязі тривалого часу.

Розпізнавання номеру тролейбуса



Для цього у систему вводяться зворотні зв’язки. І тепер у розрахунках вагових сум приймають участь не тільки вхідні сигнали, а ще й вихідні. У цьому випадку, як тільки ви розпізнали номер 6 і прийняли рішення, що це їде ваш тролейбус, тоді ви починаєте наче підсилювати ваше рішення і гальмувати усі інші. І тепер, щоб переконати вас, що це їде не

«ваш» тролейбус із номером 6, а «не ваш» тролейбус із номером 8, або навіть із номером 1, треба, щоб рівень вхідних сигналів, що розпізнають номери 1 та 8, були значно більшими, оскільки із їхніх вагових сум тепер віднімається результат спрацьовування нейрона, що колись розпізнав цифру 6.

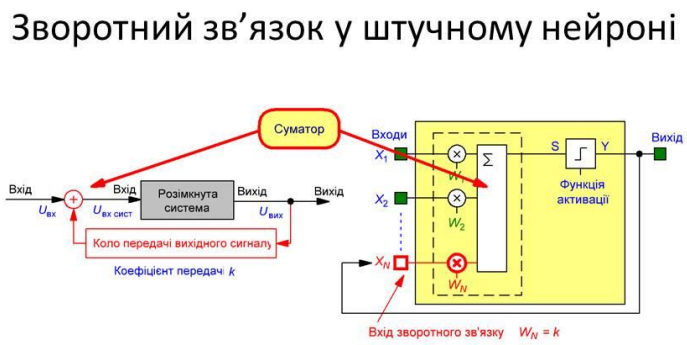
А який це зв'язок позивний чи негативний? Згадаємо приклади застосування позитивного та негативного зв'язків. Негативний зворотний зв'язок використовується для стабілізації певного параметру, а позитивний навпаки – для дестабілізації. Якщо б ми використовували негативний зворотний зв'язок, то він намагався б підтримати систему у певному стабільному стані. А який стан для системи розпізнавання образів є самим стабільним? У даному випадку, для системи розпізнавання образів самим стабільним є стан, коли ніякого рішення не приймається, тобто коли вихідні сигнали нейронів мають певне фіксоване значення. Згадайте стабілізатор напруги, що ми розглядали перед цим – його задача – підтримувати вихідну напругу у заданих межах у будь-якій ситуації. Тому, якщо б ми використовували негативний зворотний зв'язок, то сигнал на виході нейрона, що розпізнає цифру 6, та і усіх інших нейронів, ніколи б не став більше порога активації.

Згадайте самі – коли ви не хочете приймати якесь рішення, ви завжди будете шукати якісь виправдування, тобто причини, які будуть гальмувати активації нейрона, який перевів би вас у активний стан. Тому у випадку очікування, ви так і залишитесь на зупинці і не сядете у жоден тролейбус. Ця проблема відома як проблема Буриданова віслюка, у якого було дві копни сіна, але він помер з голоду, тому що через надлишок розуму не міг вирішити з якої копни сіна йому розпочати трапезу – із лівої, чи з правої.

Отже негативний зворотний зв'язок у випадку прийняття рішень може стати вагомою завадою до будь-якого вирішення. Тому у нейронних мережах, що використовуються для прийняття рішень – однією із головних інтелектуальних функцій, притаманних людині – частіше використовують саме позитивні зворотні зв'язки.

Як реалізувати зворотний зв’язок в нейронній мережі

А як реалізувати зворотний зв’язок у нейронних мережах? Давайте згадаємо, що необхідно для того, щоб розімкнута система, тобто система без зворотного зв’язку, перетворилася на замкнуту, тобто системою із зворотними зв’язками. А для цього потрібно у систему додати два вузли – коло для передачі сигналу зворотного зв’язку, яке має певний коефіцієнт передачі і вхідний суматор, який буде додавати до вхідного сигналу сигнал зворотного зв’язку.

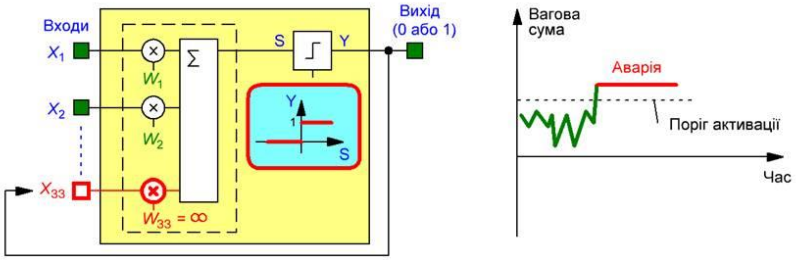


А тепер згадайте будову штучного нейрона. Одним із основних вузлів штучного нейрона є ваговий суматор, який обчислює суму вхідних сигналів, кожен з яких перемножується на свої вагові коефіцієнти. Як бачите, для введення у штучний нейрон сигналу зворотного зв’язку достатньо до нього додати іще один вхід, який нічим не буде відрізнятися від інших входів. Якщо нейронна мережа реалізується програмно, то для додавання до нейрону ще одного входу потрібно змінити лише кілька констант у програмі, що, при правильно спроектованому програмному забезпеченні, робиться менше ніж за хвилину. Тобто можна вважати, що для введення сигналу зворотного зв’язку у штучний нейрон майже нічого робити не потрібно.

Зверніть увагу, що параметри зворотного зв’язку цілком та повністю залежать від значення вагового коефіцієнта, зв’язаного із відповідним входом зворотного зв’язку. Якщо цей коефіцієнт менший нуля, то реалізується негативний зворотний зв’язок. Якщо більший нуля – то позитивний. А якщо цей коефіцієнт дорівнює нулю, то ніякого зворотного зв’язку немає, оскільки яке б число ви не помножили на нуль, результат буде нульовим і цей вхід не буде мати ніякого впливу на величину вагової суми і, відповідно, на стан нейрона. Цією властивістю часто користуються під час налаштування нейронних мереж, бо у деяких випадках нейронну мережу спочатку потрібно навчити без зворотних зв’язків, а вже потім активувати зворотні зв’язки.

Крім того, зверніть увагу, що якщо абсолютне значення, тобто модуль, вагового коефіцієнту більше за одиницю, то сигнал підсилюється. Якщо менше за одиницю – то послаблюється, а якщо дорівнює одиниці – то передається без змін. Цією властивістю інколи користуються під час реалізації особливої поведінки нейронної мережі, яка називається «защіпка».
Функція защіпки часто використовується коли систему потрібно чітко та надійно зафіксувати у певному стані, наприклад, у стані аварії. У більшості випадків, виникнення подібного стану є ознакою того, що автоматизована система керування вже не спроможна самостійно приймати рішення і потребує втручання людини. У даному випадку, один із нейронів спеціально виділяється для того, щоб виявляти цей стан. Цей нейрон має порогову функцію активації, і його вихідне значення є дискретним і може приймати значення або 0, або 1.

Функція «защіпки»



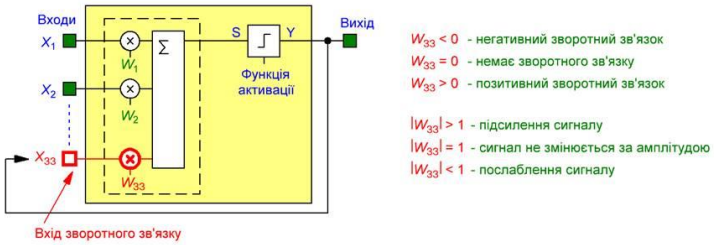
Для реалізації функції защіпки ваговому коефіцієнту входу, пов’язаному із зворотним зв’язком, присвоюється дуже велике позитивне значення, тобто нейрон охоплюється глибоким позитивним зворотним. Усі інші входи аварійного нейрона підключаються до контрольованих сигналів. Коли усі контрольовані сигнали знаходяться у межах норми, то вагова сума менше порога активації, і сигнал на виході аварійного нейрона дорівнює нулю. Зверніть увагу, що, якщо вихідний сигнал нейрона дорівнює нулю, то зворотний зв’язок виявляється начебто відключеним і ніяк не впливає на вагову суму. Та якщо вагова сума стане більшою порогового значення, то нейрон активується і на його виході з’явиться сигнал, що дорівнює рівню логічної одиниці. Як тільки це станеться, ця одиниця активує позитивний зворотний зв’язок і до вагової суми одразу почне додаватися безкінечно велике значення, яке буде утримувати нейрон в активному стані навіть якщо вхідні сигнали повернуться у нормальні межі. Але система вже буде знаходитись у стані аварії і вивести її з цього стану можна буде лише вручну, наприклад, перезавантаженням або ручним розриванням кола зворотного зв’язку.
Функція защіпки активно використовується у різних технічних системах, зокрема в охоронних системах або системах пожежної сигналізації. Ви знаєте, що у випадку пожежі достатньо один раз натиснути на кнопку пожежної сигналізації, то вона активується і буде утримуватись у цьому стані навіть після того, як цю кнопку відпустити. А от щоб вимкнути пожежну сигналізацію вже потрібне втручання людини, яка вручну – якщо пожежу ліквідовано – перезавантажить систему. Така ж сама функція реалізована у більшості комп’ютерних блоків живлення. Якщо потужність, яку споживає обладнання системного блоку, стає більше за максимально допустиме значення, то блок живлення, щоб уникнути виходу із ладу, перейде у стан блокування і вивести його з цього стану можна тільки шляхом повного відключення від мережі.

Приклади систем із функцією защіпки

Пожежна сигналізація | Охоронна сигналізація | Блок живлення комп'ютера

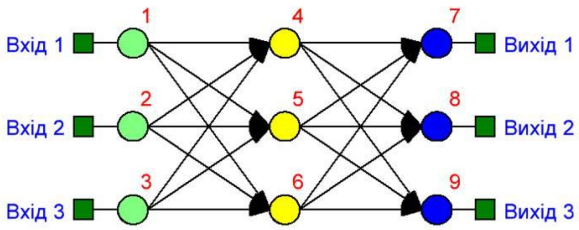
Отже для того, щоб реалізувати зворотний зв'язок у штучному нейроні майже нічого не потрібно робити – достатньо лише виділити у ньому додатковий вхід, який нічим не буде відрізнятися від інших входів. Проте це твердження зовсім не стосується нейронних мереж, що мають зворотні зв'язки. І програмна реалізація нейронних мереж із зворотними зв'язками дуже сильно відрізняється від нейронних мереж із прямим поширення інформації.

Налаштування зворотного зв'язку



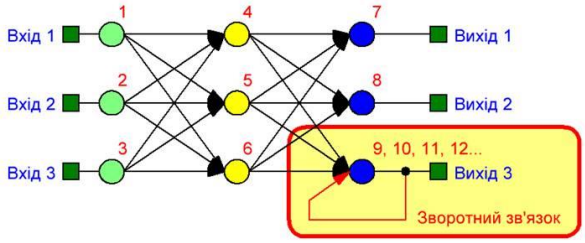
Давайте розглянемо як програмно реалізується нейронна мережа із прямим поширенням інформації. Нехай у нас буде мережа із дев'ятьма нейронами, що містить три нейрони у вхідному шарі, три – в прихованому і три – в вихідному. Якщо у нас немає зворотних зв'язків, то для обчислення подібної нейронної мережі достатньо лише послідовно обчислити стан нейронів у її шарах. Тобто, спочатку обчислити стан вхідних нейронів, причому у будь якому порядку. Після того, як обчислені вихідні сигнали вхідних нейронів, можна переходити до обчислення стану нейронів у прихованому шарі – усі дані для цього у нас є. І знову ж таки, обчислювати стан нейронів у прихованому шарі можна у будь-якому порядку – на кінцевий результат це ніяк не вплине. Ну, а після того, як обчислили стан усіх нейронів у прихованому шарі, можна переходити і до розрахунку станів нейронів у наступному – у даному випадку – вихідному шарі.

Порядок обчислення нейронів у нейронній мережі із прямим поширення інформації



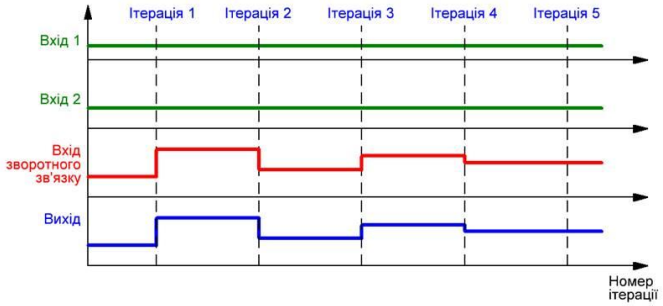
Отже, як бачите, при обчисленні стану нейронної мережі із прямим поширенням інформації немає нічого складного – достатньо лише послідовно обійти усі шари, починаючи від вхідного, і у кожному шарі послідовно обчислити усі нейрони, причому всередині шару нейрони можна обчислити у будь якому порядку.
А тепер додаймо зворотний зв’язок. Наприклад охопимо зворотним зв’язком лише один нейрон у вихідному шарі. Як тепер зміниться наш алгоритм обчислення? Зрозуміло, що спочатку треба обчислити нейрони вхідного шару, потім нейрони прихованого, а потім нейрони вихідного шару. Отже порядок обчислення нейронів, що не охоплені зворотними зв’язками нічим не змінився. Але що відбудеться із останнім нейроном у вихідному шарі, який охоплений зворотним зв’язком?

Порядок обчислення нейронів у нейронній мережі із зворотним зв’язком



Перед обчисленням його стану на входах цього нейрона, у тому числі і на вході, виділеному для подачі зворотного зв’язку, присутні певні рівні сигналів. Після того, як обчислений стан нейрона, тобто, обчислена його вагова сума та результат функції активації, рівень сигналу на виході нейрона зміниться. У нейронні мережі із прямим поширенням інформації зміна вихідного сигналу нейрона не впливає на стан його вхідних сигналів. А от у мережі із зворотними зв’язками зміна вихідного сигналу нейрона, призведе до зміни його вхідних сигналів. У даному випадку сигнали, що приходять із інших нейронів залишаться незмінними, а от сигнал зворотного зв’язку зміниться. Отже стан нейрона буде некоректним і його потрібно обчислити ще раз, але вже з іншою комбінацією вхідних сигналів.

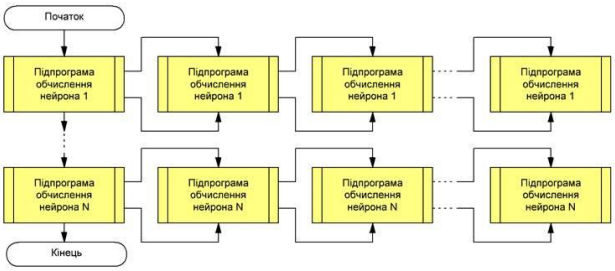
Обчислення нейронної мережі із зворотним зв’язком



Але після другого обчислення вихідний сигнал нейрону також може змінитися, що потребує проведення нової ітерації обчислення, але вже з третьою комбінацією вхідних сигналів. І так потрібно робити допоки вихідний сигнал не перестане змінюватися і не перейде у якийсь стабільний стан.
Такі програми називаються рекурсивними. Рекурсія – це виклик тієї ж самої підпрограми всередині самої підпрограми. Використання рекурсії є доволі ризикованим методом програмування, оскільки програма може перейти у нескінчений цикл, що призведе або до її зависання, або до виникнення переривання, пов’язаного із переповненням стеку викликів підпрограм. Проте без використання рекурсії нейронну мережу із зворотними зв’язками

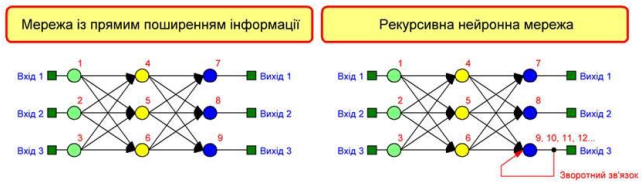
обчислити просто неможливо, тому програмістам, що будуть реалізовувати нейронні мережі із зворотними зв'язками доводиться впроваджувати механізми запобігання краху програми, наприклад, вводити механізмів контролю кількості вкладених викликів однієї ж тої підпрограми.

Алгоритм рекурсивної програми



Тому дуже часто у технічній літературі нейронні мережі із зворотними зв'язками називають рекурсивними нейронними мережами. Цією назвою підкреслюють той факт, що при обчисленні нейронних мереж із зворотними зв'язками використовують рекурсивні алгоритми, які, по-перше, складніші в реалізації, а по-друге, потребують більшої кількості арифметико-логічних операцій, порівняно із мережами, у яких інформація поширюється лише в одному напрямку і зворотні зв'язки не використовуються.

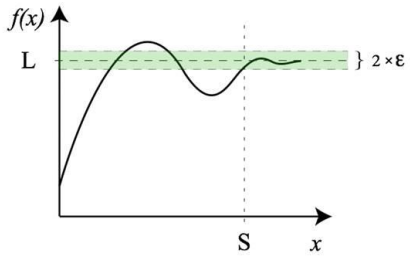
Рекурсивна нейронна мережа



Нейронні мережі із зворотними зв'язками (рекурсивні нейронні мережі) мають більш складну програмну реалізацію та потребують більшої кількості обчислень

При реалізації нейронних мереж із зворотними зв'язками виникає проблема, яка називається проблемою збіжності. Збіжність у математиці означає те, що нескінченна послідовність, сума нескінченного ряду або невласний інтеграл мають границю. У випадку нейронних мереж збіжність означає, що при нескінченній кількості рекурсивних викликів підпрограми результат розрахунку виявиться кінцевим, тобто буде наближатися до певного числа. Але не всім нейронним мережам притаманна така властивість.

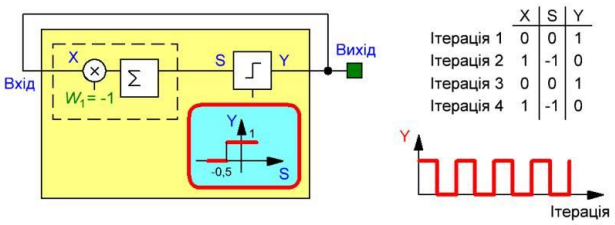
Збіжність функцій



Давайте розглянемо нейронну мережу, що містить лише один нейрон, охоплений зворотним зв'язком. Нехай ваговий коефіцієнт входу зворотного зв'язку дорівнює –1. Також нехай у

якості активаційної функції використовується порогова функція із порогом спрацьовування – 0,5. Приблизно такий нейрон ви створювали на одному із практичних занять, коли намагалися налаштувати його для реалізації базових логічних функцій. Але тоді у вас була мережа із прямим поширенням інформації, а у даному випадку у вас є мережа із зворотним зв'язком.

Нейрон, який неможливо обчислити
(незбіжна нейронна мережа)



	X	S	Y
Ітерація 1	0	0	1
Ітерація 2	1	-1	0
Ітерація 3	0	0	1
Ітерація 4	1	-1	0

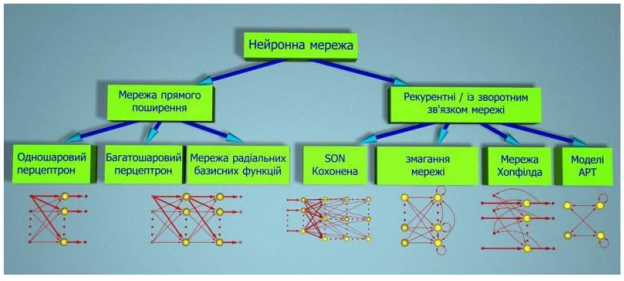
Припустимо, що після ініціалізації нейронної мережі на виході нейрону штучно встановлений сигнал, що дорівнює нулю. Давайте обчислимо яким буде стан нейрона при такому вхідному сигналі. Ваговий суматор обчислить добуток вхідного сигналу на ваговий коефіцієнт. У цьому випадку, вагова сума буде дорівнювати нулю, оскільки вхідний сигнал дорівнює саме цьому значенню. Але нуль є більшим за поріг активації нейрона, який дорівнює –0,5. Отже нейрон перейде у збуджений стан, і на його виході буде сигнал, рівний одиниці.

Отже стан нейрона змінився, що потребує проведення ще одного обчислення. Добуток вхідного сигналу на ваговий коефіцієнт буде дорівнювати –1, що є меншим за поріг активації нейрона. Отже при такому значенні вагової суми нейрон перейде у незбуджений стан і на його виході буде становлений сигнал, що дорівнює нулю. Отже стан нейрона знову змінився, що потребує проведення ще одного обчислення.

Та при наступному обчисленні виявиться, що вихідний сигнал нейрона знову стане рівним одинці, а при наступному – рівним нулю. Отже така система є незбіжною і її обчислити фізично неможливо, оскільки програма одразу «провалиться» у нескінчений цикл і залишить там назавжди.

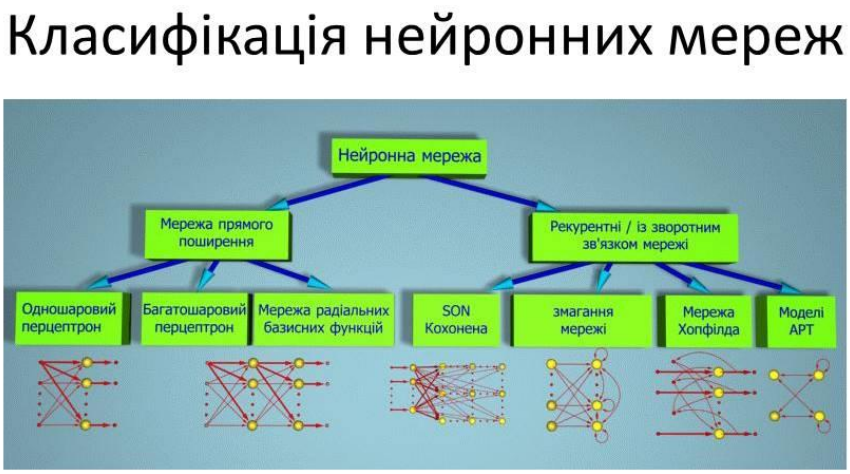
Таким чином, далеко не всі нейронні мережі із зворотними зв'язками можуть бути практично реалізовані. І це, до певного часу, було вагомою проблемою для практичного застосування нейронних мереж цього типу, оскільки не було ніякої гарантії, що подібна система у процесі роботи не стане незбіжною і не перестане функціонувати. Проте частину з проблем було успішно вирішено, і зараз системи із зворотними зв'язками використовуються так само, як і традиційні персептрони, оскільки зворотний зв'язок дозволяє реалізувати набагато більше функцій, ніж метод прямого поширення інформації. І на наступній лекції ми з вами обов'язково із ними познайомимся. А на сьогодні все – дякую за увагу!

Класифікація нейронних мереж



Лекція 6. Нейронні мережі із зворотними зв’язками

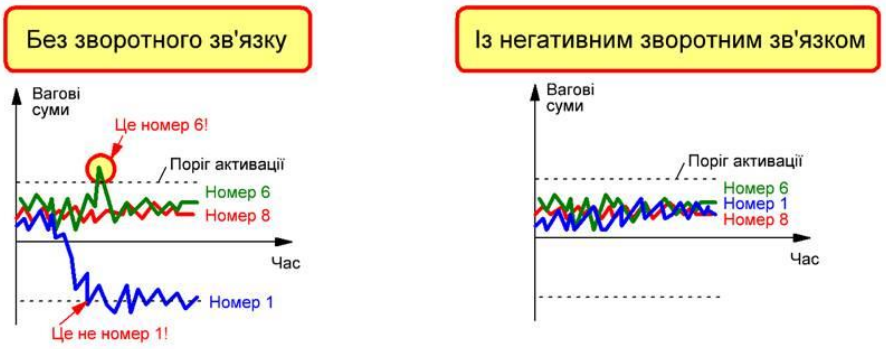
На попередніх лекціях ми з вами познайомилися із класифікацією нейронних мереж, і з’ясували, що усі нейронні мережі поділяються на дві великі групи: мережі із прямим поширенням інформації та мережі із зворотними зв’язками. Ми докладно розібрали принципи побудови мереж першої групи і з’ясували, що вони мають певні особливості, які можна віднести до класу недоліків. А саме, у мережах без зворотних зв’язків принципово не можна реалізувати деякі функцій, зокрема функцій короткочасної пам’яті.



А от введення зворотних зв’язків допомагає нейронній мережі прийняти рішення і перевести її у певний стабільний стан, у якому вона буде утримуватися протягом певного часу. У прикладі, який ми розглядали на одній із попередніх лекцій, було показано, що введення позитивного зворотного зв’язку допомагає навіть людині швидше прийняти рішення. У випадку, коли людина на зупинці тролейбуса намагається роздивитися його номер, достатньо лише миті навіть нечіткого розпізнавання для того, щоб перейти у стан або «Я сідаю у цей тролейбус», або «Доведеться ще трохи почекати».

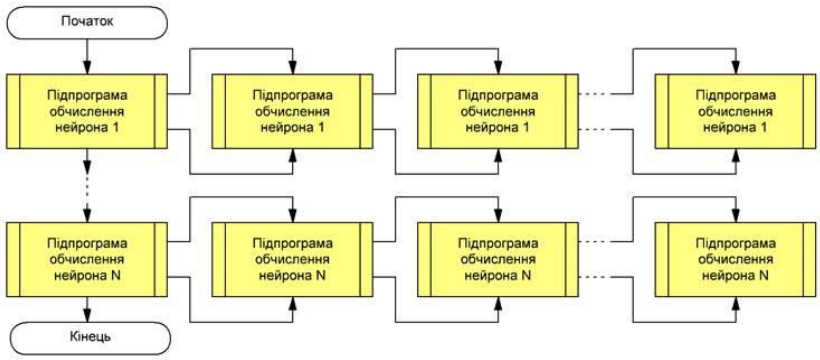
Ми з вами також докладно розглянули чим відрізняється позитивний зворотний зв’язок від негативного. І з’ясували, що коли нейронна мережа призначається для прийняття рішень, то краще використовувати позитивний зворотний зв’язок, оскільки негативний буде утримувати нейронну мережу у стані невизначеності, і мережі буде складно приймати рішення.

Розпізнавання номеру тролейбуса



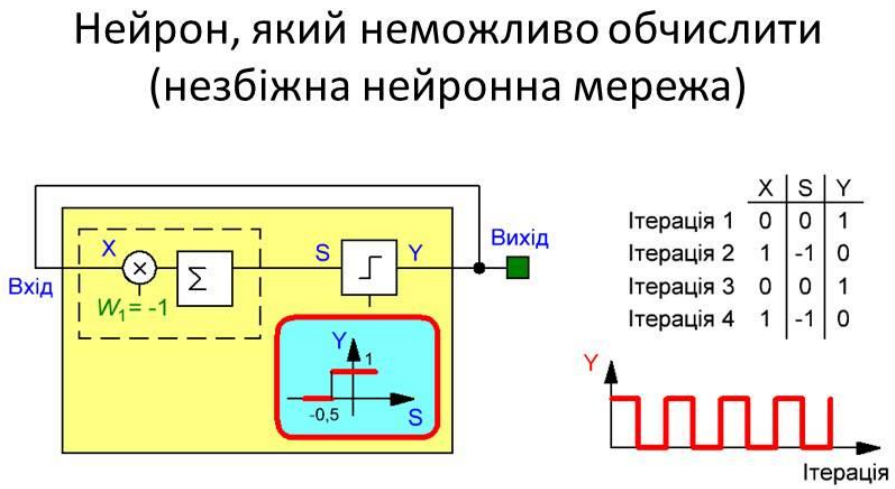
Ще однією особливістю нейронних мереж із зворотними зв’язками, яку треба обов’язково згадати, є проблема збіжності. Ви пам’ятаєте, що для обчислення нейронних мереж із зворотними зв’язками необхідно використовувати рекурсивні методи програмування. Але рекурсивні методи програмування, по-перше, потребують більшої кількості арифметико-логічних операцій, а по-друге, використання рекурсії може стати причиною переходу програмного забезпечення у нескінченний цикл. І ця остання особливість протягом тривалого часу була серйозною проблемою для практичного використання нейронних мереж.

Алгоритм рекурсивної програми



На одній із попередніх лекцій я вже наводив вам приклад незбіжної нейронної мережі – мережі, стан нейронів якої неможливо обчислити через наявність зворотного зв’язку. У наведеному прикладі, який ви зараз бачите на екрані, результат обчислення стану єдиного нейрону при кожній наступній ітерації буде суттєво відрізнятися від попереднього, тому програма, яка буде обчислювати цю систему ніколи самостійно не припиниться.

Але з часом проблему незбіжності нейронних мереж із зворотними зв’язками частково вирішили, і зараз існують доволі багато типів нейронних мереж із зворотними зв’язками, із якими ми й ознайомимося у цьому матеріалі. А розгляд нейронних мереж цього типу ми розпочнемо із мережі Гопфілда, яка була популяризована Джоном Гопфілдом ще у 1982 році.



Мережа Гопфілда

Та перед тим як докладно розглянути цю нейронну мережу, слід зазначити, що нейронні мережі Гопфілда мають аналоги, як в живій, так і в неживій природі. І одним із природних аналогів мережі Гопфілда є спінове скло.

Спінове скло (Spin glass) – це дуже розбавлений магнітний сплав, у якому на тисячу атомів немагнітного матеріалу припадає кілька штук магнітних. Наприклад, спінове скло утворюється коли атоми феромагнітного заліза, випадковим чином вставляють у кристалічну решітку немагнітних атомів міді. Незважаючи на те, що атомів заліза небагато, вони радикальним і дуже загадковим чином змінюють магнітні властивості матеріалу. Кожен атом заліза поводиться як маленький магніт, орієнтація якого (спін) впливає на інші атоми заліза, розташовані поруч із ним, так само як орієнтація інших магнітних атомів впливає на його орієнтацію. У звичайному магніті орієнтація усіх елементарних магнітів спрямована в одному напрямку. А от у спіновому склі відбувається так звана геометрична фрустрація. У даному випадку слово «фрустрація» означає, що геометричні властивості кристалічної решітки через наявність протидіючих міжатомних сил унеможливлюють одночасну орієнтацію усіх елементарних магнітів в одному напрямку, тому у спіновому склі орієнтація усіх елементарних магнітів є результатом певного «компромісу» магнітних полів. І зміна орієнтації навіть одного магніту призведе до зміни орієнтації усіх інших.

Приблизно те ж саме відбувається у приміщеннях, у яких знаходяться люди, що взаємно заважають один одному. Подібна система постійно знаходиться в одному з стабільних станів – коли люди заважають один одному щонайменше. Та варто тільки одній людині переміститися у інше місце в кімнаті, як зростає рівень внутрішньої напруженості, що призводить до переміщення усіх інших людей у місця де вони, знову ж таки, будуть якнайменше заважати один одному.

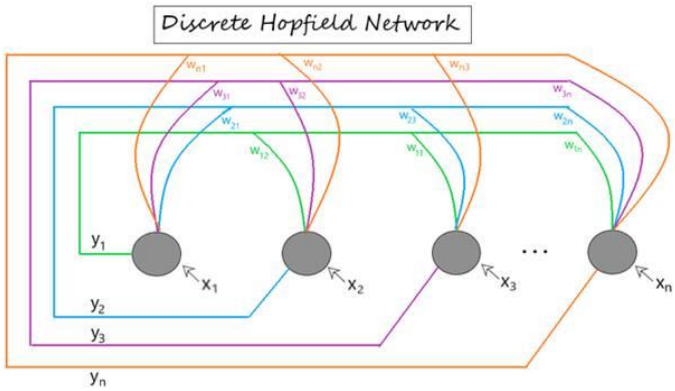
Спінове скло

Системи, аналогічні спіновому склу, постійно знаходяться в одному із кількох стабільних станів для виведення з якого потрібно додаткове зовнішнє зусилля (додаткова «енергія»)

Таким чином, системи, аналогічні спіновому склу, постійно знаходяться в одному із кількох стабільних станів, і для того щоб вивести систему із цього стабільного стану потрібно певне зовнішнє зусилля, яке у теорії подібних систем називають енергією. Тобто, якщо людина у такій кімнаті недалеко зміститься із свого місця, то інші присутні у цьому приміщенні можуть, завдяки наявності внутрішніх зв'язків, повернути її на своє місце, тобто повернутися у попередній стабільний стан. Але якщо збуджувач системи має достатньо велику «енергію», то він таки змінить своє місце, що призведе до примусового переміщення усіх інших елементів у новий стабільний стан.

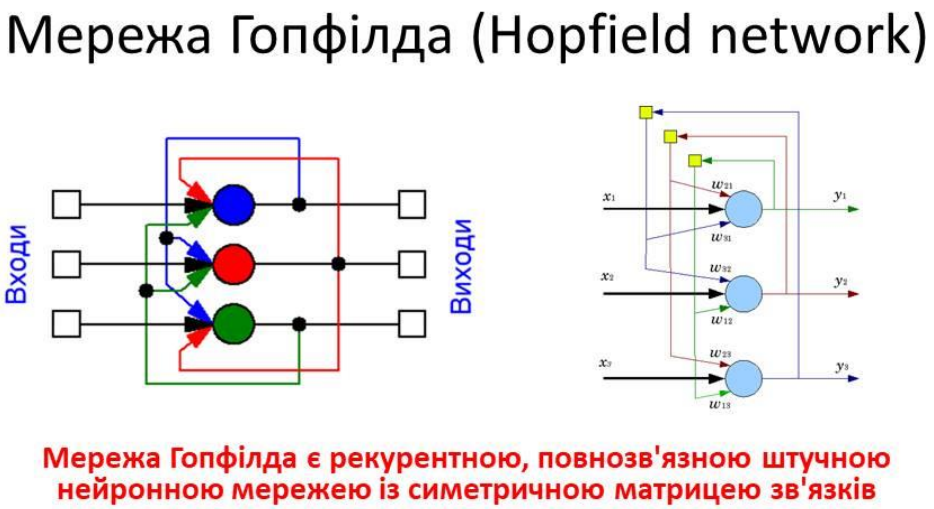
Нейронна мережа Гопфілда

Джон Джозеф Гопфілд
1933 рн

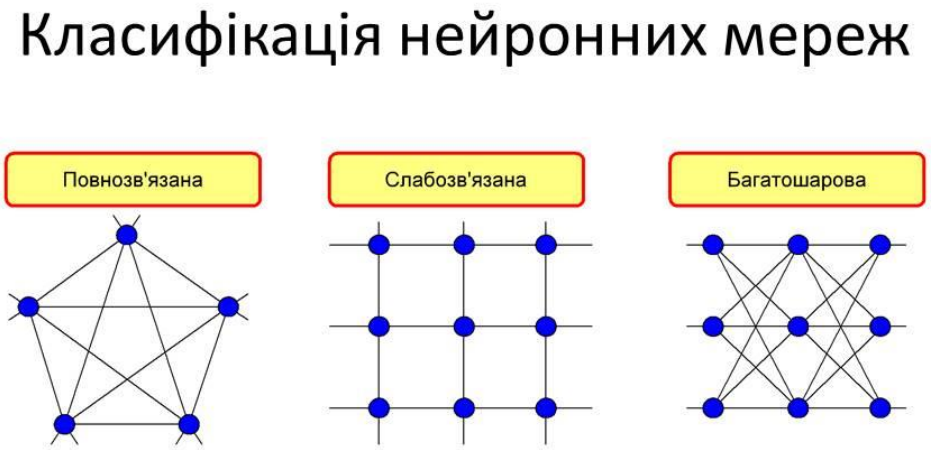


Саме на такій моделі поведінки і була спроектована мережа Гопфілда, хоча вперше її описав Літтл у 1974 році. А сам Літтл створив цю мережу на основі сумісної роботи Ернста Ізінга та Вільгельма Ленца. Але роботи Ізінга, Ленца та Літтла відомі лише у вузькому колі фахівців, а Гопфілд у 1982 році популяризував цю мережу, що стало причиною завершення першої «зими» штучного інтелекту та поновлення робіт в області штучних нейронних мереж.

Відповідно сучасної класифікації, мережа Гопфілда є рекурентною, повнозв'язаною, штучною нейронною мережею з симетричною матрицею зв'язків. Мережа Гопфілда не має чітко виражених вхідних та вихідних шарів. Кожен нейрон цієї мережі одночасно є і вхідним і вихідним нейроном, тому кількість входів нейронної мережі Гопфілда завжди дорівнює кількості виходів. Зверніть увагу на те, як формуються зворотні зв'язки. Вихід одного нейрона подається на входи інших нейронів причому усіх, окрім самого себе. Таким чином, кількість входів кожного штучного нейрона мережі Гопфілда також дорівнює кількості нейронів. Отже, якщо у нас є мережа Гопфілда із трьома виходами, то вона складається із трьох нейронів, кожен з яких має три входи, два з яких підключені до виходів інших нейронів, а третій вільний вхід призначений для введення вхідних сигналів.

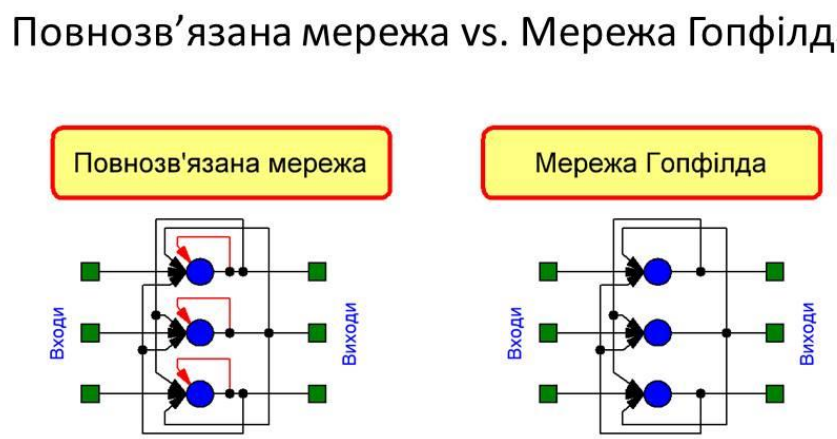


У визначенні мережі Гопфілда з'явилося два нових слова – повнозв'язана та рекурентна, які нам наразі поки що невідомі. Повнозв'язаними називають нейронні мережі, усі нейрони яких мають зв'язки із усіма іншими нейронами. Мережі для яких це правило виконується частково називаються слабозв'язаними нейронними мережами. Ну а мережі, у яких зв'язок підтримується тільки між певними нейронами, які можна виділити у шари, а це вже знайомі вам мережі із прямим поширенням інформації, називаються, відповідно, багатошаровими нейронними мережами.

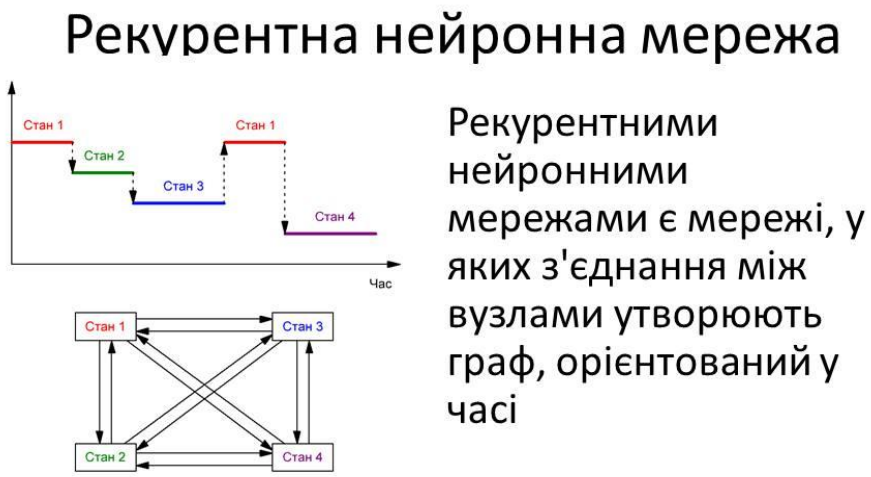


Мережа Гопфілда відноситься до повнозв'язаних нейронних мереж, хоча формально це не зовсім так. Річ у тому, що у повноз'язаній мережі усі нейрони повинні мати зв'язки із усіма іншими нейронами, у тому числі і з самими собою. Тобто для того, щоб перетворити мережу

Гопфілда на повноцінну повнозв’язану мережу до кожного нейрона потрібно додати іще один вхід, який необхідно з’єднати із його виходом. Але хоч у мережі Гопфілда ці зв’язки і відсутні, усе одно вона більше відноситься по повноз’язаних мереж ніж мереж із слабими зв’язками.

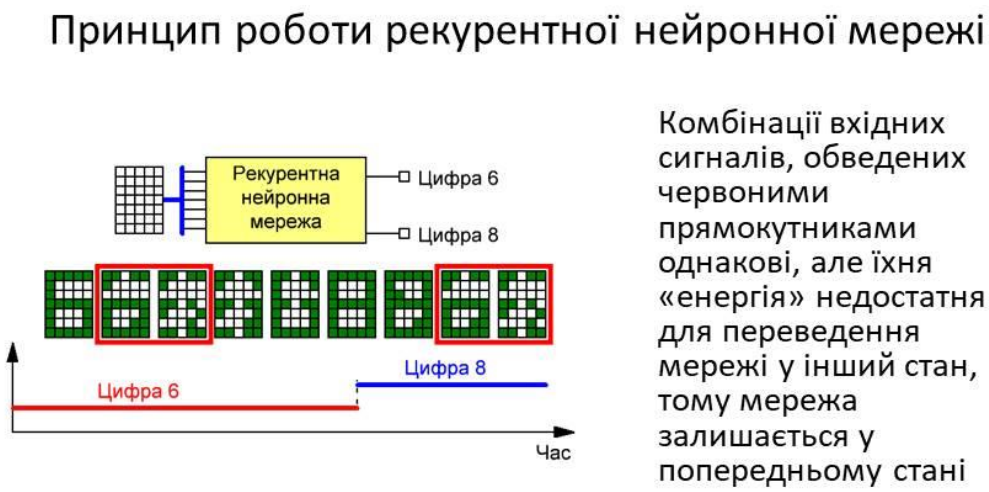


Другим поки що незнайомим для нас словом є слово «рекурентна». Рекурентними нейронними мережами є мережі, у яких з'єднання між вузлами утворюють граф, орієнтований у часі. Така організація нейронної мережі дозволяє утримувати її у певному стані протягом тривалого часу, допоки комбінація вхідних сигналів не буде мати енергію, достатню для переводу мережі у інший стабільний стан. Розглянемо це на прикладі.



Нехай у нас є система, призначена для розпізнавання цифр 8 та 6, які дуже схожі між собою. Це нейронна мережа має певну кількість входів і лише два виходи. Зверніть увагу, ми зараз не розглядаємо конкретно мережі Гопфілда. Ми зараз розглядаємо довільну рекурентну мережу – а це доволі широкий клас нейронних мереж. Отже, у нас є рекурентна нейронна мережа, що має два виходи, які відповідають виявленню цифр 6 та 8. Завдяки наявності внутрішньої пам’яті, реалізованою за допомогою внутрішніх зв’язків, система може знаходитися лише в одному із двох станів. Тобто вона сигналізує, що у нас є цифра 6 або 8.
Отже нехай на вході мережі з’явилася комбінація сигналів, що перевела її у стан, що відповідає розпізнаванню цифри «6». Із часом вхідні сигнали починають спотворюватися, наприклад через наявність шуму або погіршення умов освітлення. Але нейронна мережа, завдяки наявності зворотних зв’язків, що утворили короткочасну пам’ять, пам’ятає що перед цим вона розпізнала цифру «6». І у такому стані вона буде знаходитись навіть тоді, коли зображення спотвориться до невпізнанності. Тобто, навіть якщо зображення не буде взагалі ніякого, мережа буде пам’ятати свій попередній стан та знаходитись у стані «я розпізнала цифру 6».

Та ось на зображенні починає поступово формуватися цифра «8». Але система усе одно буде знаходитись у стані «я розпізнала цифру 6» навіть якщо зображення вже буде більше схоже на цифру «8» ніж на «6». Але рано чи пізно настане такий момент, коли загальна «енергія» вхідних сигналів досягне такої величини, що система зможе подолати внутрішній бар'єр і швидко, майже миттєво, перейде у стан «я розпізнала цифру 8». Тепер система знаходиться у другому стабільному стані – «я розпізнала цифру 8» із якого її вивести буде так само важко, які при находженні у стані «я розпізнала цифру 6». Тому коли зображення знову почне спотворюватися, і стане таким же самим як і при спотворенні цифри «6», то система усе одно буде знаходитись у стабільному стані.



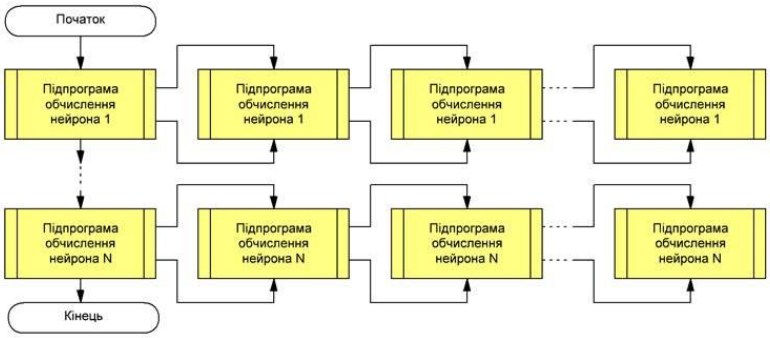
Це добре видно на зображеннях, що обведені червоними прямокутниками. Комбінації вхідних сигналів для цих моментів часу однакові, але у першому випадку мережа знаходиться у стані «я розпізнала цифру 6», а у другому – «я розпізнала цифру 8», оскільки «енергії» цих сигналів недостатньо для виводу мережі із стабільного стану.

Це явище еквівалентно гістерезису. Проте коли говорять про гістерезис, то зазвичай мають на увазі функцію від однієї змінної, що може приймати два стану. А у нашому випадку нейронна мережа може знаходитись у декількох стабільних станах, для зміни яких потрібно докласти певних зусиль. Тому подібні мережі, зазвичай, розглядають з енергетичної точки зору. Якщо енергії від комбінації вхідних сигналів достатньо, щоб перевести мережу у інший стабільний стан, то вона до нього перейде, як якщо ні, тобі результати обчислення, хай навіть через кілька ітерацій розрахунку, збіжаться до попереднього стану.

Зверніть увагу, стабільні стани рекурентної нейронної мережі, у загальному випадку, можуть мати різні енергетичні рівні. Це означає, вивести мережу із одного стабільного стану може

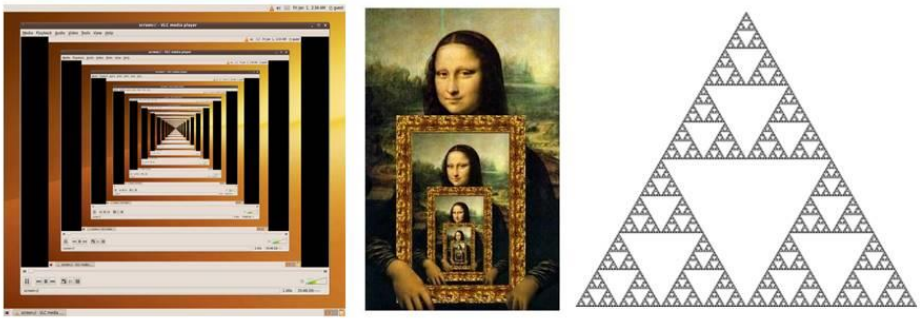
виявитися набагато простіше, ніж з іншого. Проте це вже буде конкретною специфікою самої мережі і її навчання.
На попередній лекції ми звертали увагу на те, що для обчислення стану нейронних мереж із зворотними зв’язками використовують рекурсивні методи програмування, тому подібні системи часто називають рекурсивними. Але річ у тому, що саме поняття «рекурсія» відноситься не тільки до галузі програмування.

Алгоритм рекурсивної програми



Рекурсія (лат. recursio) – це метод визначення класу чи об'єкту через попереднє задання одного чи декількох, зазвичай більш простих, його базових випадків чи методів, а потім заданням на їхній основі правила побудови класу, який визначається. Іншими словами, рекурсія – це часткове визначення об'єкта через самого себе, або визначення об'єкта з використанням раніше визначених об’єктів. Рекурсія використовується, у тих випадках, коли можна виділити самоподібність задачі.

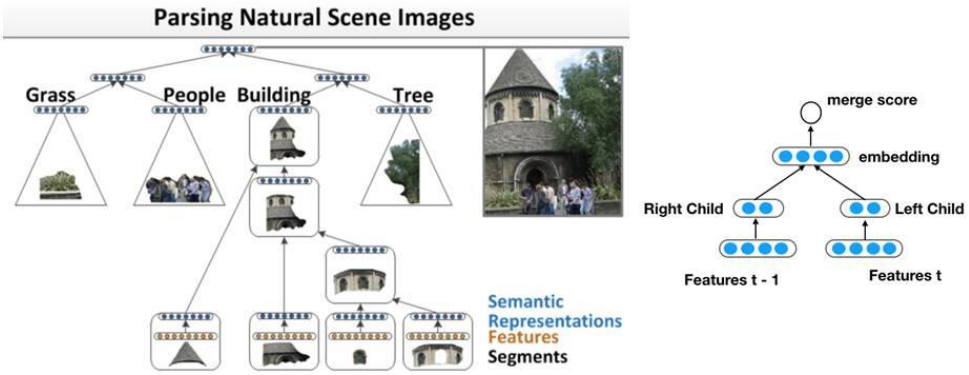
Рекурсія



Рекурсія – це часткове визначення об'єкта через самого себе

Тому, коли мова заходить про нейронні мережі, то під словом «рекурсія» слід мати на увазі більш широке поняття, яке відносить не тільки до технологія побудови програмного забезпечення. У даному випадку під терміном «рекурсивна нейронна мережа» мається на увазі нейронна мережа, що має ієрархічну структуру, яка подібна дереву. Елементи цього дерева подібні один одному, зокрема мають однакові розмірності матриці вагів. І тому при їх обчисленні рекурсивні методи використовують не тільки для обчислення стану конкретного нейрона, а і для обчислення загального стану усієї мережі.

Рекурсивна нейронна мережа



Прикладом рекурсивних нейронних мережі є системи, призначені для розпізнавання складних зображень. У даному випадку одна складна сцена розбивається на кілька більш простих сцен, далі знову йде розбивка на ще більш прості сцени, далі іще на більш простіші в так допоки зображення на розіб'ється на окремі примітиви, які легко розпізнати. Як бачите сама нейрона мережа у даному випадку має динамічну структуру – кожна нейронна мережа більш високого рівня містить у собі кілька внутрішніх нейронних мереж, кожна з яких у свою чергу, також можу містити кілька власних нейронних мереж і так далі. Кількість розгалужень та рівнів подібної системи може динамічно змінюватися. Зрозуміло, що практично побудувати подібну нейронну мережу, використовуючи лише статичні методи зв'язку нейронів неможливо. Тому у даному випадку і використовується рекурсія, яка, теоретично, допускає використання будь якої кількості вкладених елементів.

Рекурентні vs. Рекурсивні нейронні мережі

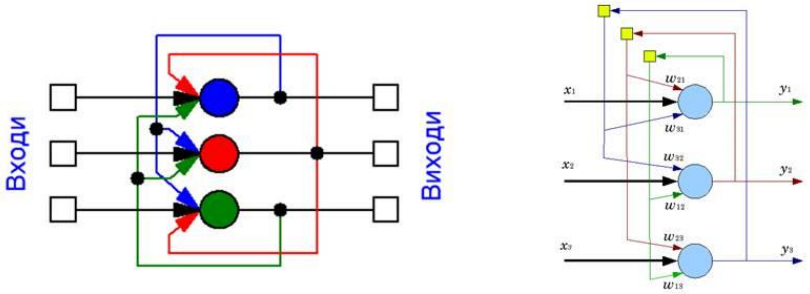
- **Рекурентні нейронні мережі** (PHM, Recurrent Neural Networks, RNN) — це клас штучних нейронних мереж, у якому з'єднання між вузлами утворюють граф орієнтований у часі
- **Рекурсивні нейронні мережі** (PHM, англ. Recursive Neural Networks, RNN) — це клас глибинних нейронних мереж, створюваних рекурсивним застосуванням одного й того ж набору ваг до різних елементів структури

Рекурентні нейронні мережі відносяться до класу рекурсивних нейронних мереж

Рекурсивні нейронні мережі є самим складними нейронними мережами, які існують на даний час, і вони також є найменш вивченими. Проте саме вони дозволяють реалізувати самі складні функції, притаманні штучному інтелекту. Тому можна сказати, що саме ці типи нейронних мереж, скоріш за все будуть основою майбутніх інтелектуальних систем. Ну а поки що, докладної інформації про рекурсивні нейронні мережі доволі мало, а їх різновидів, тобто часних випадків є доволі багато. У любому випадку вам слід відрізняти рекурсивні нейронні мережі від рекурентних нейронних мереж. Зокрема пам'ятати, що рекурентні нейронні мережі є підкласом рекурсивних нейронних мереж, тобто рекурентні нейронні мережі є рекурсивними нейронними мережами але з обмеженими можливостями.

Отже мереже Гопфілда, із якої ми почали наш огляд є рекурентною, повнозв'язною штучною нейронною мережею із симетричною матрицею зв'язків. Тепер ми точно знаємо сенс усіх слів, що входять у це означення, а отже можемо рухатися трохи далі.

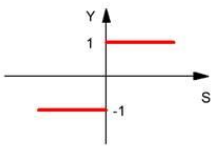
Мережа Гопфілда (Hopfield network)



Мережа Гопфілда є рекурентною, повнозв'язною штучною нейронною мережею із симетричною матрицею зв'язків

В активаційних елементах нейронів мережі Гопфілда використовуються порогова функція активації із біполярним вихідним сигналом, тобто сигналом, що може приймати значення або –1, або 1. Для обчислення стану нейронів потрібно використовувати рекурсивні методи програмування. До недоліків нейронної мережі Гопфілда слід віднести доволі обмежений обсяг пам’яті. Дана мережа спроможна запам’ятати лише обмежену кількість образів, яку можна обчислити за формулою, яку ви зараз бачите на екрані. Якщо кількість образів буде більшою, то мережа перестане їх запам’ятовувати і навчити її буде неможливо. Крім того, мережа Гопфілда має певні проблеми із навчанням. Не зважаючи на те, що мережа Гопфілда завжди буде знаходись в одному із стабільних станів, ніхто не гарантує того, що цей стан біде правильний. Це відбувається через те, що мережа може зійтися до так званих хибних атракторів, які іноді називають «химерами» – зображень, що складаються із фрагментів різних образів.

Мережа Гопфілда



$$M \approx \frac{N}{2 \ln N}$$

Особливості
- Використання порогової функції активації
- Використання рекурсивних методів обчислення стану нейронів

Недоліки
- Обмежений обсяг пам’яті (M), при більших значеннях мережа перестає працювати
- Наявність обмеженої кількості станів не гарантує правильних відповідей

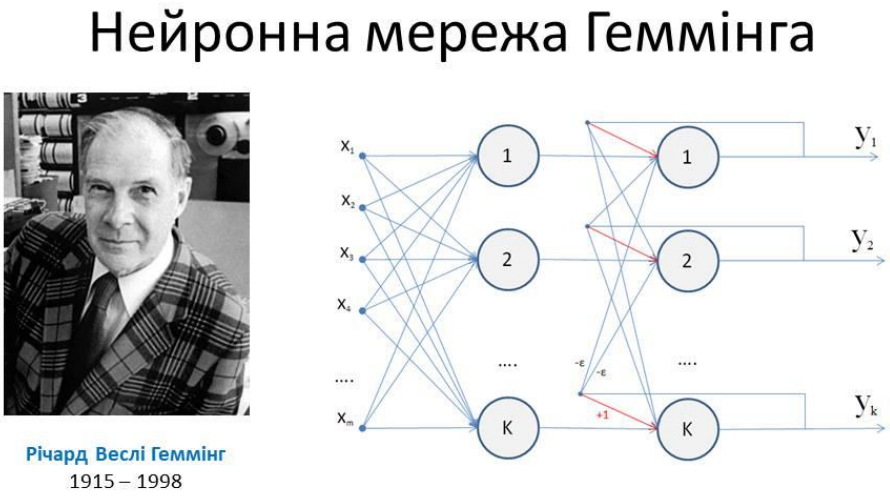
Застосування
- Відновлення пошкоджених зображень
- Системи із асоціативною пам’яттю
- Комбінаторні задачі оптимізації

Мережі Гопфілда були популярними у 80-х роках минулого сторіччя. Як було сказано на початку лекції, популяризація мереж Гопфілда дозволила закінчити першу «зиму» штучного інтелекту та активізувати роботи у цій галузі, що потім призвело до появи більш досконалих мереж Наразі до мереж Гофілда виявляють лише академічний інтерес, зоча це не означає, що їх не можна використовувати у практичних системах. Наприклад, мережі Гопфілда доволі успішно справляються із розпізнаванням пошкоджених образів, тобто їх можна використовувати для відновлення пошкоджених зображень. Крім того мережі Гопфілда можна використовувати для систем, що потребують наявності асоціативної пам’яті, та також у застосунках, де необхідно вирішувати певний клас комбінаторних задач.

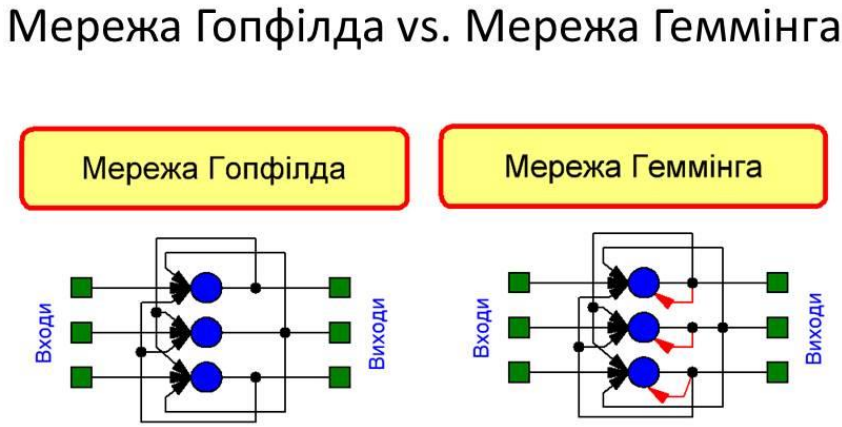
Нейронні мережі Геммінга

Нейронні мережі Геммінга використовуються для реалізації асоціативної пам'яті в тих випадках, коли немає необхідності, щоб мережа видавала на виході образ у явному вигляді, а лише його номер (або код). У порівнянні з мережею Гопфілда мережа Геммінга має менші витрати на пам'ять та обсяг необхідних обчислень.

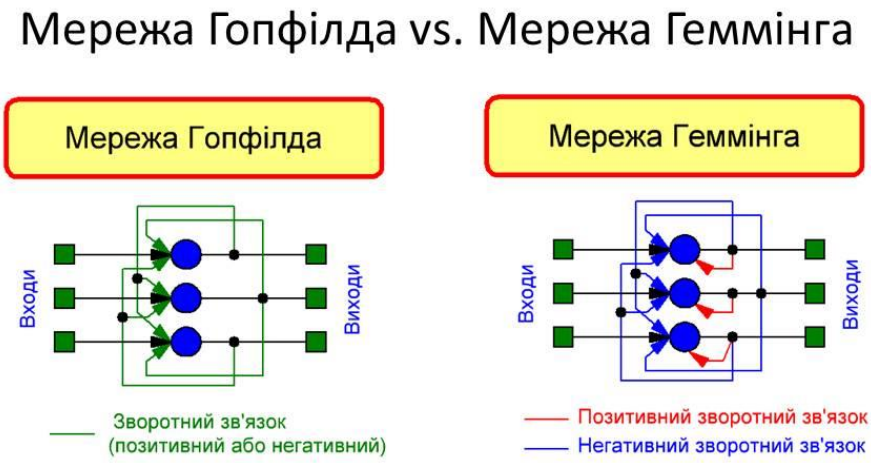
Нейронна мережа Геммінга складається з двох шарів, кожен з яких містить число нейронів M, що дорівнює кількості образів, що зберігаються. Нейрони першого шару є вхідними. Нейрони другого шару пов'язані між собою зворотними зв'язками. Причому зворотний зв’язок між виходом одного нейрона та входами інших нейронів є негативним ці зв’язки виділені на рисунку синіми кольорами, а зв’язок між виходом та входом того ж самого нейрона є позитивним і він виділений на рисунку червоним кольором.



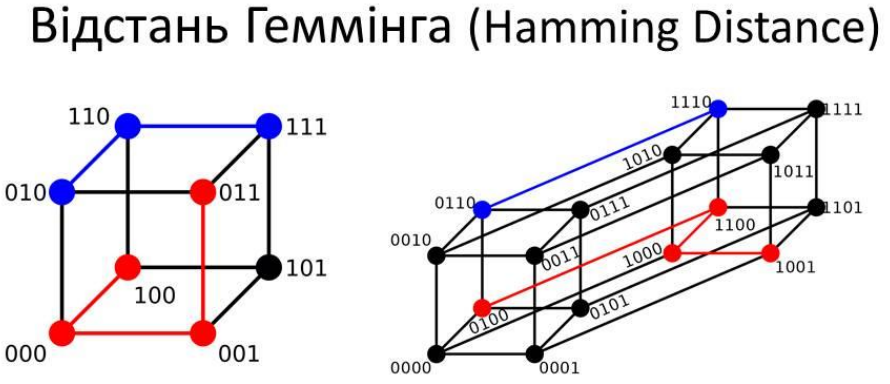
Це означає, що поява позивного сигналу на вході нейрона починає додатково збуджувати цей нейрон і гальмувати усі інші. Мережа Геммінга дуже схожа на мережу Гопфілда. Єдиною головною відмінністю мережі Геммінга від мережі Гопфілда є наявність додаткового позитивного зворотного зв’язку у кожному нейроні. Зверніть увагу, що ви вже бачили сьогодні точно таке зображення, коли ми розглядали питання що таке повнозв’язана мережа. Тоді ми говорили, що мережа Гопфілда відновиться до повнозв’язаних мереж, хоча вона і не має декількох зв’язків, і для перетворення мережі Гопфілда на повнозв’язану мережу потрібно додати ще додаткові зв’язки між виходом нейрона і його входом. Тепер ми бачимо, що додавання таких зв’язків перетворює мережу Гопфілда на мережу Геммінга. Але є ще одна відмінність, на яку необхідно звернути увагу.



Річ у тому, що у мережах Гопфілда явно не вказано який тип зворотних зв'язків між нейронами використовується: позитивний або негативний. Кінцеве значення вагових коефіцієнтів на цих входах підбирається у процесі навчання і, теоретично воно може бути любим. А от у мережі Геммінга типи зворотних зв'язків вказується явно: на рівні одного нейрона зворотний зв'язок повинен бути позитивним, а між цим нейроном та іншими – негативним.



Із нейронними мережами Геммінга тісно пов'язане таке поняття як відстань Геммінга. Відстань Геммінга (Hamming distance) є числом позицій, у яких відповідні цифри двох двійкових слів однакової довжини різні. У загальнішому випадку відстань Геммінга застосовується для рядків однакової довжини будь-яких абеток, що складаються з певної кількості символів, і служить метрикою відмінності об'єктів однакової вимірності, тобто функцією, що визначає відстань в метричному просторі. Відстань Геммінга фактично вимірює мінімальну кількість замін, необхідних для зміни одного рядка в інший, або мінімальну кількість помилок, які могли перетворити одну стрічку в іншу, або кажучи простими словами – наскільки одна інформація відрізняється від іншої. Ну а якщо говорити про загальні речі, то у більш загальному контексті відстань Хеммінга є однією з метрик рядків для вимірювання відстані між двома послідовностями.

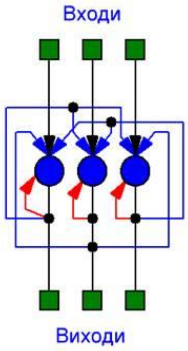


Відстань Геммінга – число позицій, у яких відповідні цифри двох двійкових слів однакової довжини різні

Мережі Геммінга фактично є подальшим розвитком мереж Гопфілда, свого роду їх поліпшеною версією. Тому моделі Геммінга можуть запам'ятати більшу кількість образів, тобто комбінації вхідних сигналів, порівняно із мережею Гопфілда при тієї ж самої кількості нейронів. Сфера застосування мереж Геммінга приблизно та ж сама, що і у мереж Гопфілда. Мережі Геммінга доволі успішно справляються із розпізнаванням образів, і можуть їх

відновлювати у випадку пошкоджень. Мережі Геммінга також можна використовувати у системах, що потребують наявності асоціативної пам'яті, та також у застосунках, де необхідно вирішувати певний клас комбінаторних задач

Мережі Геммінга



Мережі Геммінга є подальшим розвитком мереж Гопфілда

Застосування

- Відновлення пошкоджених зображень
- Системи із асоціативною пам'яттю
- Комбінаторні задачі оптимізації

Особливості

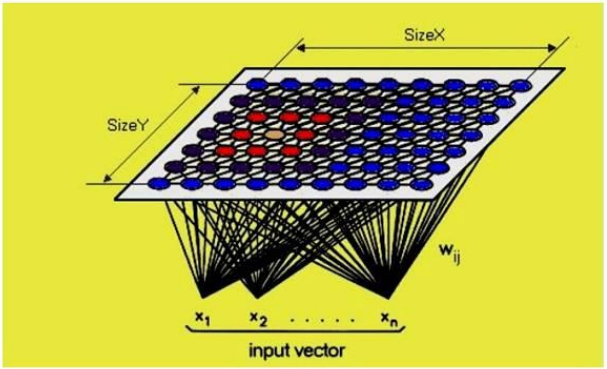
- Можуть запам'ятати більшу кількість зв'язків порівняно із мережами Гопфілда при тої ж кількості нейронів

Нейронні мережі Кохонена

Нейронні мережі Теуво Кохонена були представлені у 1982 році як спеціальний вид нейронних мереж для обробки зображень і звука. Основним призначенням мереж Кохонена є перетворення складних багатомірних даних в більш просту структуру малої розмірності, як правило дволмірної. Але виявилось, що вони гарно пристосовані для кластерного аналізу, коли треба виявити приховані закономірності у великих масивах даних. Мережа Кохонена складається з нейронів, які об'єднуються в кластери. Найбільш близькі нейрони відповідають схожим об'єктам, а віддалені – не схожим. В основі побудови мережі лежить конкурентне навчання, коли вихідні вузли (нейрони) конкурують між собою за «перемогу». У ході змагань в процесі навчання нейрони вибірково налагоджуються для різних вхідних прикладів

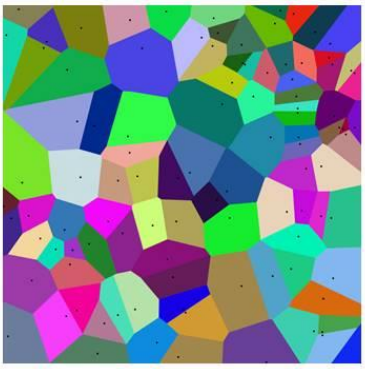
Нейронна мережа Кохонена

Теуво Калеві Кохонен
1934 – 2021



Основу нейронної мережі Кохонена є так званий «шар Кохонена» – певна кількість вихідних нейронів, які зв'язуються між собою. Мережа Кохонена є одношаровою, але на відміну від мережі Гопфілда, у якій немає чіткого поділення на вхідні та вихідні нейрони, в мережі Кохонена є окремий шар вхідних нейронів, та окремий шар вихідних нейронів. В базовій версії цієї мережі кожен вхідний нейрон зв'язаний із кожним вихідним нейроном, що дуже нагадує нейронні мережі із прямим поширенням інформації, зокрема одношарові персептрони Розенблата та Румерхальта.

Мережа Кохонена та кластерний аналіз



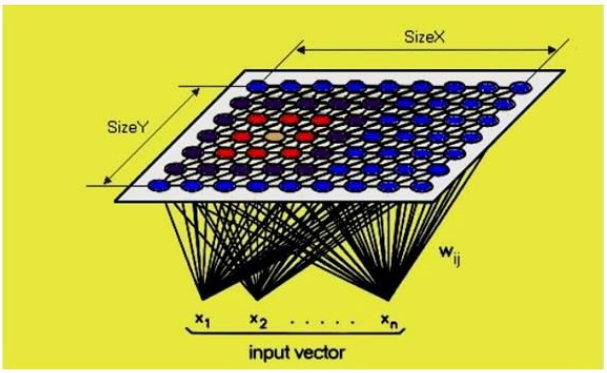
Після навчання мережі одномірне (частіше – двомірне) вихідне поле (шар Кохонена) дозволяє наочно показати до якого класу (кластеру) відноситься поточна комбінація вхідних сигналів

Вихідні нейрони мережі Кохонена працюють за принципом «**переможець забирає все**». Вихідний сигнал нейрона із найбільшим рівнем стає рівним 1, а усіх інших нейронів – 0.

Вхідні нейрони утворюють вхідний шар, якій містить по одному нейрону для кожного вхідного поля. Як і у звичайній нейронній мережі вхідні нейрони не беруть участі у процесі навчання. Їх задачею є передача значень вхідних полів початкової вибірки на нейрони вихідного шару. Кожен зв'язок між вхідними та вихідними нейронами має певну вагу, яка в процесі ініціалізації встановлюється випадковим чином в інтервалі [0;1]. Потім ці вагові коефіцієнти налаштовуються в процесі навчання. Після навчання мережі формується одномірне, але частіше – двомірне, вихідне поле, що має прямокутну або шестикутну форму. Це дозволяє наочно показати до якого класу (кластеру) відноситься поточна комбінація вхідних сигналів. Вихідні нейрони мережі Кохонена працюють за принципом «переможець забирає все». Вихідний сигнал нейрона із найбільшим рівнем стає рівним 1, а усіх інших нейронів – 0. Це дозволяє під час роботи мережі дуже швидко класифікувати до якої категорії, тобто до якого кластеру, відноситься та чи інша комбінація вхідних сигналів.

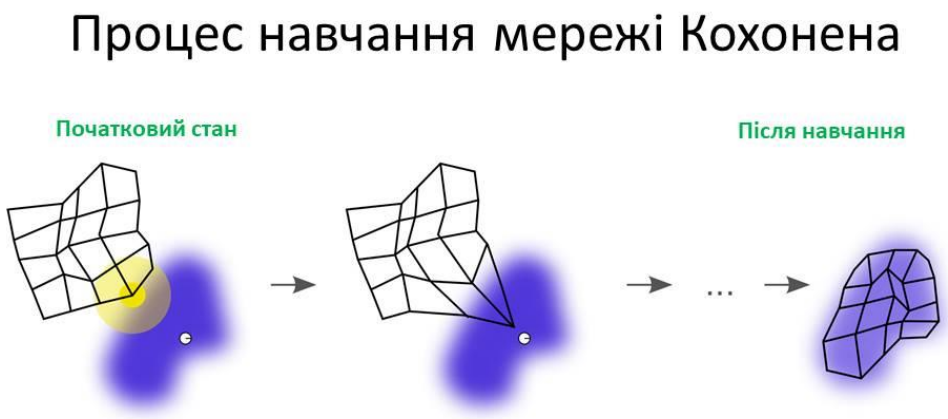
Нейронна мережа Кохонена

Теуво Калеві Кохонен
1934 – 2021



Процес навчання мережі Кохонена поділяється на три етапи. На першому етапі, що називається конкуренцією, вихідні нейрони конкурують між собою за те, щоб вектори їх вагових коефіцієнтів виявились як можна ближче до вектору ознак об'єкта. Вихідний нейрон, який виявляється найближчим до об'єкта, оголошується переможцем. На другому етапі, що називається об'єднанням, нейрон-переможець стає центром деякої групи сусідніх нейронів. У мережі Кохонена усі нейрони, розташовані поряд із нейроном-переможцем, нагороджуються правом підлатуванням вагових коефіцієнтів. Тому, не дивлячись на те, що нейрони у вихідному шарі не з'єднуються безпосередньо, вони мають схожі набори ваг завдяки сусідству з нейроном-переможцем. Ну а на третьому етапі відбувається заключне підстроювання вагових коефіцієнтів нейронів, що розташовані поряд з нейроном-

переможцем. Цей процес називають адаптацією. У результаті такого навчання вихідне поле нейронної мережі Кохонена поділяється на певні області – кластери, які дозволяють швидко класифікувати об’єкт за певною ознакою.

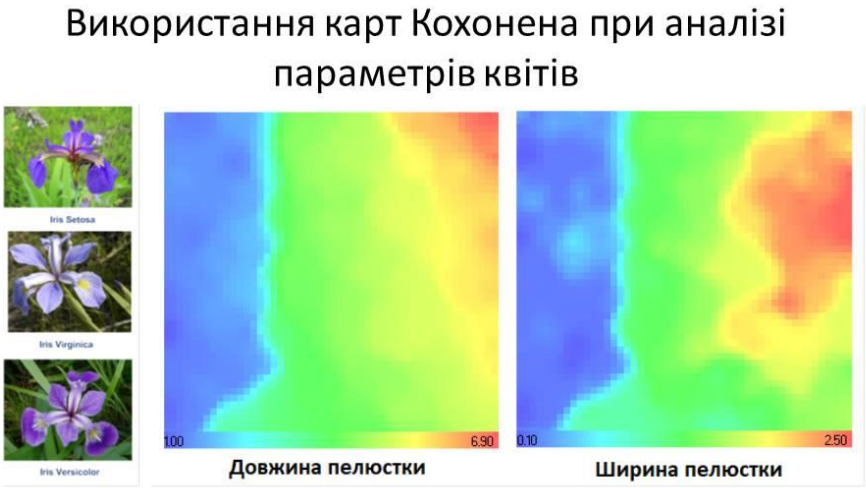


Головною особливістю мережі Кохонена є здатність до самонавчання та самоорганізації. Із навчанням нейронних мереж ми вже трішки знайомі. Але усі методи навчання нейронних мереж, які ми раніше розглядали – то бути методи навчання із вчителем. Тобто коли нейронній мережі надаються навчальні комбінації вхідних сигналів, на які вона повинна реагувати відповідним чином. А от мережі Кохонена можуть навчатися без вчителя. Тобто на входи мережі можна подавати тестові сигнали, а мережа вже буде самостійно їх ідентифікувати та класифікувати – до якого кластеру їх віднести.

Але нейронна мережа Кохонена спроможна і до самоорганізації. Що це таке? Усі мережі, які ми вивчали до цих пір, ну, можливо, окрім рекурсивних нейронних мереж, які ми пару хвилин назад розглядали, мали фіксовану структуру. Тобто кількість нейронів та зв’язки між ними задавалися людиною у процесі створення мережі. Ми вже не раз говорили, що коли ваговий коефіцієнт певного входу штучного нейрона дорівнює нулю, то це означає розрив зв’язку. Але зв’язок – апаратний чи програмний – усе одно залишається. І процесор усе одно множить вхідний сигнал цього нейрона на його ваговий коефіцієнт, хоч він і дорівнює нулю. Отже під час створення тих нейронних мереж, які ми перед цим розглядали, структуру та зв’язки між нейронами усе одно створює людина. А от мережа Кохонена спроможна до самоорганізації, тобто вона сама може вирішувати скільки нейронів у шару Кохонена їй потрібно і які вагові коефіцієнти вони повинні мати.

Самоорганізаційна карта Кохонена (англ. Self-organizing map – SOM) – це нейронна мережа з некерованим навчанням, яка використовується для перетворення багатовимірного простору ознак в простір з меншою розмірністю (частіше за все – двовимірний). Самоорганізаційна карта Кохонена створює дискретне представлення вхідних просторів навчальних вибірок, які називаються картами (англ. map), тому використання цього типу нейронної мережі є методом для зниження розмірності.

Карти Кохонена призначені для візуалізації багатовимірних об’єктів на двовимірних картах, де відстані між об’єктами відповідають відстаням між їх векторами у багатовимірному просторі, а самі значення ознак відображаються різними кольорами і відтінками. Таким чином, можна провести аналогію карт Кохонена зі звичайною географічною картою, яка лініями відображає кордони країн (кластерів), а різними кольорами – рельєф поверхні (значення параметрів). При аналізі певних об’єктів для кожної ознаки, наприклад, для ширини або довжини пелюсток квітів створюються окремі карти, що дозволяють дуже швидко класифікувати певні об’єкти за певними ознаками.

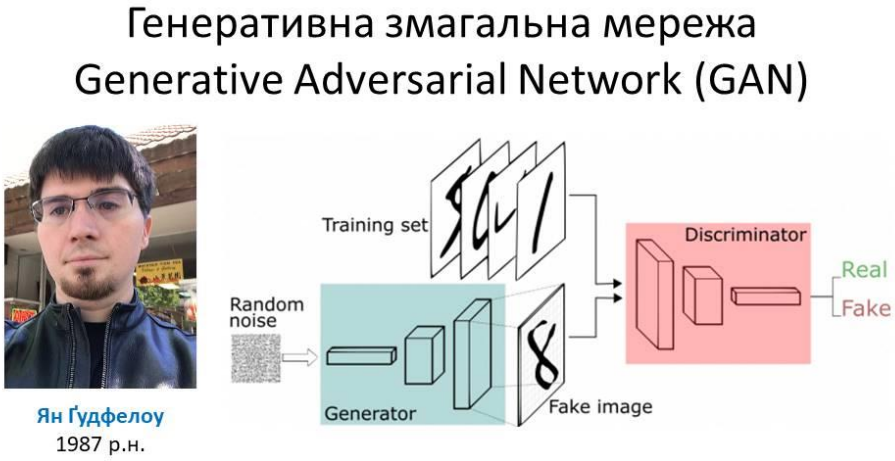


Отже, нейронні мережі та карти Кохонена дозволяють дуже швидко, і головне – наочно, проводити аналіз багатоозначних об’єктів та явищ, що є корисним інструментом у процесі прийняття рішень. Мережі Кохонена активно використовуються в метеорології та океанографії. Вони використовуються при визначенні пріоритетів під час вибору, наприклад, найкращих проектів. Вони використовуються в процесі ошуку корисних копалин, у тому числі нафти та газу. Вони також використовуються у різних видах аналізу, починаючи від пошуку причин несправностей у техніці і закінчуючи аналізом фондових ринків. Ну а одним із головних напрямків використання мереж Кохонена, як ви вже напевно самі здогадалися, є

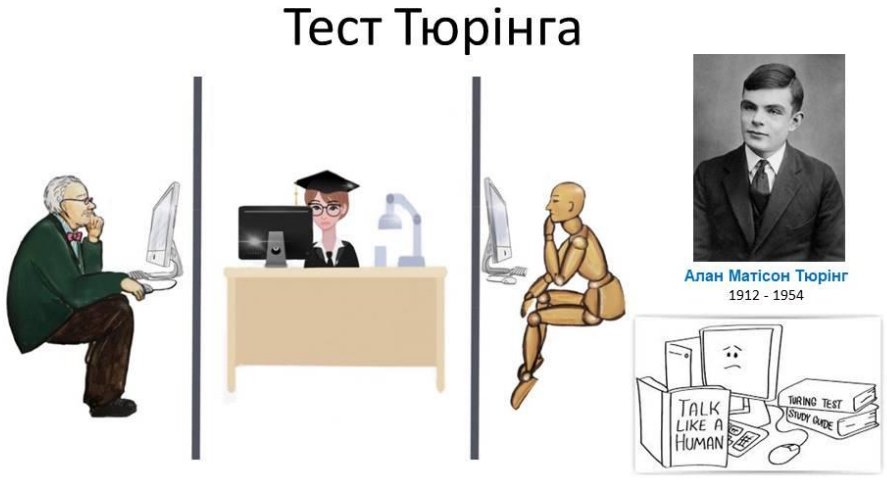
створення різних ілюстрацій, оскільки головною перевагою мереж Кохонена є саме наочна ілюстрація положення певного об'єкта серед об'єктів, що подібні йому.

Генеративні змагальні мережі

Генеративні змагальні мережі (Generative adversarial networks, GANs) – це клас алгоритмів штучного інтелекту, що, так само як і мережі Кохонена, використовують принципи навчання без вчителя. Генеративні змагальні мережі є системою, що складається із двох штучних нейронних мереж, які змагаються одна з одною у форматі гри. Вони були вперше запроваджені у 2014 році Яном Ґудфелоу, що працював у таких компаніях як Apple та Google. Така мережа, а точніше комбінація нейронних мереж, дозволяє створювати зображення, які для типової людини мають багато реалістичних елементів, тобто виглядають як справжні, і інколи навіть професійні експерти не можуть відрізнити реальні зображення від згенерованих.



Генеративна змагальна мережа складається із двох частин: творця та експерта. Творець, якого зазвичай називають генератором, створює певні об'єкти, а експерт, який в оригінальній версії генеративних змагальних мереж називається дискримінатором їх оцінює та виносить своє рішення: правдоподібне це рішення або ні. Генеративна нейронна мережа створює свої зображення, як правило, з певного набору готових зображень, який часто називають латентним простором певного розподілу даних. А дискримінаційна нейронна мережа повинна відрізняти представників справжнього розподілу даних від кандидатів, вироблених генератором. Метою тренування мережі є збільшення частоти помилок експерта, тобто у процесі тренування намагаються «обдурити» дискримінатора шляхом створення екземплярів, які повинні походити на справжні зображення.



Ідея створити конкурентне середовище (генератор проти дискримінатора) була запропонована Лі, Гаучі та Гросом в 2013 році. Їх метод використовується для прогнозування поведінки і називається навчання по Тюрінгу, оскільки сама ідея подібного навчання дуже схожа на відомий тест Тюрінга. Стандартна інтерпретація цього тесту звучить наступним чином: «Людина (експерт) взаємодіє з одним комп'ютером та однією людиною. На підставі відповіді на запитання вона повинна визначити, з ким він розмовляє: з людиною чи комп'ютерною програмою. Завдання комп'ютерної програми – ввести людину в оману, змусивши зробити неправильний вибір». Але у даному випадку у ролі експерта виступає не людина, а дискримінатор, якому, до речі, дозволяється впливати на процес отримання наборів даних, що робить його також активним учасником процесу навчання.
Принцип роботи генеративної змагальної мережі нерідко описується через метафори. Наприклад, генеративна частина мережі уподібнюється фальшивомонетнику чи підроблювачу картин, а оцінювальна – експерту, який прагне розпізнати підробку. Інший приклад – образ двох боксерів, один з яких навчався у майстра, а другий змушений наслідувати учня.

Принцип навчання генеративної змагальної мережі

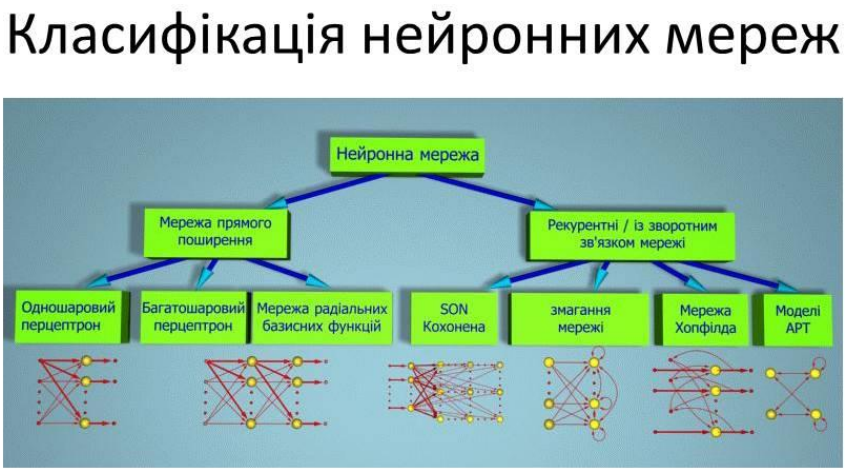
Генеративні змагальні мережі використовуються для створення фотореалістичних зображень з метою візуалізації нових дизайнів інтер'єру, взуття, сумок, одягу або сцен у комп'ютерних іграх. Наприклад, у популярному застосунку, призначеному для генерації людських обличь, джерелом первісних даних виступають реальні фотографії, на основі яких генеративна змагальна мережа намагається створити штучні особи, варіюючи комбінації таких латентних параметрів, як колір волосся, пропорції обличчя, розріз очей, форма носа, розмір вух, наявність бороди та інших.

Приклади зображень, згенерованих нейронною мережею

Відомо, що генеративні змагальні мережі використовуються Facebook. Нещодавно генеративні змагальні мережі змоделювали закономірності руху у відео. Вони використовуються для створення об’ємних моделей об'єктів, а також для покращення зображень та об’ємного моделювання в астрономії. У 2017 році для суттєвого поліпшення якості фотографій використовувалася удосконалена генеративна змагальна мережа з автоматичною генерацією текстур, причому від даної системи вимагалось скоріше створення реалістичних текстур ніж піксельна деталізація. Результатом була висока якість зображення при високій роздільній здатності. Таким чином, на сьогоднішній день генеративні змагальні мережі є одним з самих перспективних типів нейронних мереж, які активно застосовуються не тільки у якості об’єкта для академічних досліджень а і на практиці у великій кількості різноманітних застосунків.

Висновки

Отже як бачите, на сьогоднішній день існує доволі велика кількість нейронних мереж. І класифікація, яку ви зараз бачите на екрані, є доволі умовною і далеко не повною. Наразі неможливо якимось чином створити повну класифікацію, оскільки, по-перше, нейронних мереж люди вже навигадували доволі багато, а по-друге, не всі типи нейронних мереж мають практичне застосування і подальший розвиток. Крім того, не слід забувати, що кожний із типів нейронних мереж має кілька різновидів, причому деякі різновиди мають такі комбінації ознак, які в корні змінюють оригінальну ідею, яка була закладена під час створення їх оригінальних версій.



Наприклад, тільки рекурсивних нейронних мереж вже існує близько 20 різновидів, кожен з яких має свої переваги, недоліки та області практичного застосування. І ця класифікація також не є ні повною, ні статичною, оскільки над цим питанням працювало і працює дуже велика кількість як науковців, так і практиків. І хто знає, може саме зараз, коли ви дивитесь цей матеріал, хтось вигадає новий тип нейронної мережі, який створить переворот в області штучного інтелекту.

Проте для того, щоб тільки познайомитись із нейронними мережами, зрозуміти що це таке, для чого це потрібно, де воно використовується або де воно може використовуватися, наведеного матеріалу цілком достатньо. Отже далі, якщо ви вирішите працювати із нейронними мережами, то вам потрібно буде самостійно вивчити багато додаткового матеріалу. А цією лекцією ми, на жаль, закінчуємо ознайомлення зі штучними нейронними мережами і далі будемо знайомитись із іншими напрямками використання та застосування штучного інтелекту. А вам я дякую за увагу і маю надію, що ці матеріали принесли вам велику користь на шляху підвищення вашої кваліфікації.

Лекція 7. Генетичні алгоритми

На попередній лекції ми закінчили розгляд доволі великої теми, яка називалась «Нейронні мережі». На сьогоднішній день нейронні мережі є, напевно, самим великим розділом штучного інтелекту, і ви мали нагоду наочно переконатися, що у цій галузі вже доволі багато чого зроблено і, головне, потрібно ще багато чого зробити. А це означає що орієнтування на роботу у цій галузі дозволить вам забезпечити себе роботою із пристойною заробітною платою, щонайменше, на кілька десятків років. Проте штучний інтелект не обмежується тільки нейронними мережами, хоч вони, свого часу, і дали великий поштовх у цій галузі. Є ще багато напрямів розвитку цієї області, і ми поступово будемо їх вивчати.
А сьогодні ми з вами познайомимося із генетичними алгоритмами – це спеціальні алгоритми, за допомогою яких можна швидко знайти більш-менш оптимальне рішення задач, що мають велику кількість можливих варіантів. Проте переди тим як перейти безпосередньо до вивчення генетичних алгоритмів слід спочатку згадати саму першу лекцію дисципліни «Методи та системи штучного інтелекту» для того, щоб зрозуміти до якої категорії ці алгоритми відносяться.

Нейронні мережі не є єдиним напрямком розвитку штучного інтелекту

А на першій лекції було сказано, що існує два головних напрямки розвитку штучного інтелекту: прагматичний та біонічний. Прагматичний напрямок розвитку штучного інтелекту не потребує створення копії людського мозку. Тобто у цьому напрямку розумова діяльність людини приймається як якийсь «чорний ящик», який не потрібно якось докладно вивчати. Головне щоб результат роботи штучного інтелекту був подібний до результату роботи людини. Тому якщо кінцевий результат функціонування штучної системи хоча б приблизно збігається з результатом діяльності людини, то таку систему вже визнають інтелектуальною незалежно від способів отримання цього результату. За такого підходу не ставиться задача відтворити у штучній комп'ютерній системі структури та процеси, які відбуваються у мозку людини. Головне, щоб кінцевий результат розв'язання задачі був такий же, як і випадку розв'язання її людиною.

Напрями розвитку штучного інтелекту

- Прагматичний
- Біонічний

А от біонічний напрямок базується на припущенні про те, що якщо в штучній системі відтворити структури й процеси людського мозку, то й результати розв'язку задач такою

системою будуть подібні до результатів, які отримує людина. У цьому напрямку досліджень виділяються три головні напрямки: нейробіонічний, структурно-еврістичний та гомеостатичний. Із нейробіонічним напрямком ми вже трохи познайомилися – саме до нього відносяться нейронні мережі, бо вони імітують будову нервової системи живих істот, у тому числі, і головного мозку людини. А от генетичні алгоритми відносяться до гомеостатичного напрямку штучного інтелекту, тому давайте більш докладно розберемося що це за таке цікаве слово «гомеостатичний», і що воно означає.

Сучасне тлумачення слова «гомеостаз» означає стан рівноваги деякого динамічного середовища, у якому відбуваються певні процеси. Здебільшого цей термін вживається в медицині та біології. Наприклад, в медицині терміном «гомеостаз» пояснюються механізми роботи систем органів кровообігу, дихання, травлення, виділення тощо. За допомогою гомеостазу у крові живих істот підтримується необхідний рівень кисню, глюкози та біологічно активних хімічних речовин, які забезпечують взаємодію клітин і органів. На більш високому рівні гомеостаз формує макромеханізми поведінки живих істот. Наприклад, при низьких температурах пташки можуть тулитися один до одного задля збереження тепла і, відповідно, економії енергії. Причому відомо, що вони не просто туляться один до одного, а ще й постійно міняються у групі – крайні пташки, що витрачають більше тепла періодично потрапляють всередину – де втрати тепла менше, а ті, що були всередині, на деякий час потраплять на межі групи, щоб дати зігрітися іншим пташкам.

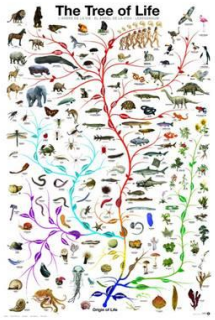
У техніці під категорії «гомеостатичні» попадають усі системи, у яких реалізовані механізми автоматизованого контролю якихось параметрів. Наприклад, до цієї категорії відносяться звичайний обігрівач повітря або добре усім відома праска, що мають вбудовані терморегулятори, які автоматично підтримують температуру повітря у кімнаті або температуру активної поверхні у заданих межах. До цієї категорії також відносять системи автопілоту транспортних засобів, у тому числі і тих, що літають або плавають, систему круїз-контролю автомобілів та багато інших систем де реалізовані автоматизовані системи регулювання на основі негативного зворотного зв’язку.

Приклади гомеостатичної поведінки технічних систем

- Термостати нагрівальних приладів (системи із негативним зворотним зв'язком)
- Круіз-контроль автомобіля
- Автопілоти літальних засобів та автомобілів
- Системи автоматизованого регулювання

Системи, що функціонують на основі генетичних алгоритмів, також відносяться до гомеостатичних систем, проте негативний зворотний зв'язок в них реалізований неявно і результатом їх роботи є не підтримання якогось параметру у заданих межах, а пошук найкращого варіанту поведінки системи у цілому. Тому поведінка систем, що працюють під керуванням генетичних алгоритмів, дуже схожа на механізм еволюції живих істот, а сам механізм створення нових елементів генетичної системи дуже схожий на механізм утворення молекул ДНК нових істот. Саме через таку подібність ці алгоритми і отримали назву «генетичні». Отже сьогодні ми з вами будемо розглядати системи, що подібні до біологічного соціуму, а тому перед тим як перейти до конкретної практичної реалізації систем, що працюють під керівництвом генетичних алгоритмів, слід спочатку зрозуміти як працюють їх біологічні аналоги, а точніше їх біологічні прототипи.

Генетичний алгоритм (Genetic Algorithm)

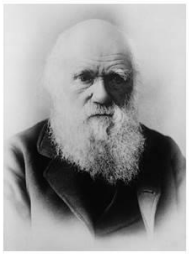


Генетичний алгоритм – еволюційний алгоритм пошуку, що використовується для вирішення задач оптимізації і моделювання шляхом послідовного підбору, комбінування і варіації шуканих параметрів з використанням механізмів, що нагадують біологічну еволюцію

Автором теорії еволюції – теорії яку, напевно, найбільше за всі інші теорії критикують, є Чарльз Дарвін, який у 1859 році випустив книгу «Походження видів шляхом природного добору або збереження обраних рас у боротьбі за життя». У цій книзі Дарвін наводить довгий ланцюжок аргументів для підтвердження того, що групи організмів, які сьогодні називають популяціями, поступово розвиваються завдяки природному відбору. При цьому одночасно він спростовував теорію «створених видів», згідно якої усе різноманіття живих істот є штучно створеним та незмінним у часі.

Відповідно до основних положень Теорії еволюції Чарльза Дарвіна, головним механізмом пристосування конкретних живих істот до довколишнього середовища, яке постійно змінюється, є випадкова змінність самих живих істот, яка передається у спадок їх дітям. Поступово це призводить до змін членів всередині популяції, які завдяки наявності природного відбору та спадкоємності, найкраще пристосовуються до довколишнього середовища. Причому із часом накопичення змін стає настільки великим, що нащадки стають зовсім не схожими на своїх батьків, що жили за багато поколінь до них, і тоді можна говорити вже про створення нових видів, тобто про створення нових популяцій.

Теорія еволюції



Чарльз Роберт Дарвін
1809 – 1882

Теорія еволюції – наукова теорія, що пояснює механізми зміни форм живих організмів, їхніх спільнот та причини утворення біорізноманіття на Землі у процесі еволюції

- Кожна популяція спроможна збільшувати свою чисельність
- Чисельність популяції залишається приблизно однаковою.
- Ресурси, необхідні для життя популяції, обмежені та відносно стабільні у часі.
- Особи у популяції повинні конкурувати за ресурси.
- Особи у популяції істотно відрізняються один від одного.
- Особливості кожної істоти є спадковими.
- Особи, менш пристосовані до навколишнього середовища, мають менше шансів вижити (природний відбір)
- Природний відбір поступово призводить до появи нових видів (популяцій), які найбільш пристосований до навколишнього середовища

Саме за такою схемою і працює генетичний алгоритм. Основним елементом генетичного алгоритму є особа – певний програмний об'єкт, який має певний перелік ознак. Сукупність осіб утворює популяцію – певний список програмних об'єктів, що має постійну або змінну кількість. У практичних реалізаціях генетичного алгоритму кількість осіб у популяції під час роботи генетичного алгоритму, зазвичай, є постійною, хоча первинна кількість – кількість, яка задається перед початком роботи генетичного алгоритму, залежить від складності конкретної задачі і може бути змінною. Поточний набір усіх осіб, що входять до популяції, називається поколінням. Під поколінням маються на увазі усі особи, які входять у популяцію у даний час.

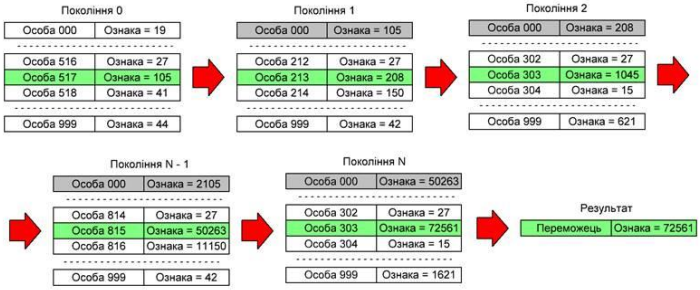
Основні поняття генетичного алгоритму

- **Особа** – програмний об'єкт, який має перелік ознак
- **Популяція** – сукупність осіб постійної або змінної чисельності (список програмних об'єктів)
- **Покоління** – поточний набір членів популяції

Особи поточного покоління спочатку дають нащадків (нове покоління) а потім знищуються (гинуть)

Генетичний алгоритм працює допоки кількість поколінь не досягне певного значення. Тобто на початку роботи генетичного алгоритму створюється перше, або нульове, покоління яке має певну кількість осіб. Ці особи спочатку дають нащадків, тобто створюють нове покоління, а потім гинуть, тобто програмно знищуються. Нове покоління також дає своїх нащадків, які утворюють нове покоління, і після цього знову знищуються. І так відбувається допоки не пройде певна кількість поколінь. Після цього генетичний алгоритм завершує свою роботу.

Результат роботи генетичного алгоритму



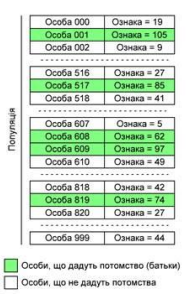
Отже, як бачите, макроалгоритм генетичного алгоритму дуже схожий на алгоритм життя живих істот. Там також, особи, що живуть у даний час, тобто у поточному поколінні, спочатку народжують дітей, які утворюють нове покоління, а потім вмирають. Єдиною різницею є те, що кількість поколінь у реальному житті теоретично не обмежена і особи із різних поколінь, тобто батьки і діти, можуть жити одночасно. А у програмній реалізації є певні обмеження, які полягають у тому, що батьки і діти не можуть жити одночасно, а кількість поколінь штучно обмежується, після цього уся популяція за винятком однієї – самої найкращої особи – обов'язково гине. А тепер переходимо до самого цікавого – яким же чином у генетичному алгоритмі утворюються діти, тобто нові особи, які сформують нове покоління.

Давайте спочатку розберемося, а хто ж із осіб поточної популяції зможе дати потомство, тобто зможе створити дітей. Відповідно до теорії еволюції Чарльза Дарвіна, потомство, теоретично, можуть дати усі особи, проте із різною імовірністю. Особи, що найкраще пристосовані до довколишнього середовища, тобто ті особи, що мають кращий доступ до обмежених ресурсів, дадуть потомство із більшою імовірністю, ніж особи, що через ті або інші причини не можуть дістатися до ресурсів і забезпечити продовження свого рожу. Саме це і називається природний відбір, який і забезпечує подальший розвиток біологічних систем.

Проте у програмній реалізації аналога біологічної спільноти немає причин для моделювання природного середовища. У даному випадку функцію природного відбору виконує програміст, що створює генетичний алгоритм. Згадаємо, що кожна особа має певний набір ознак, які відповідають її пристосованості до довколишнього середовища. На основі цього набору для кожної особи обчислюється певний середньозважений критерій, за допомогою якого і буде проведено відбір найкращих представників поточного покоління, які потім і стануть батьками для осіб наступного покоління.

Давайте розглянемо це на прикладі. Нехай у нас є популяція що налічує 1000 осіб. Перше покоління цієї популяції створюється випадковим чином, наприклад за допомогою генератора випадкових чисел. Кожна із осіб має певний критерій за яким буде проведений їх відбір. Нехай критерієм відбору буде максимальне значення цієї ознаки. Отже, після того як будуть обчислені ознаки усіх осіб, буде обрано деяка кількість осіб із максимальними значеннями цього параметру, які і дадуть потомство. Діти з'являються не просто так. Вони є результатом спеціальної процедури, яка називається схрещенням батьків. Про особливості схрещення ми поговоримо трішки пізніше, а поки подивимось до чого призведуть подібні маніпуляції із особами. Отже, після того, як сформована поточна популяція і обчислені ознаки усіх осіб, виокремлюється певна група осіб, що мають найкраще значення ознаки, які стануть батьками і дадуть потомство. Інші особи, що не ввійшли у групу батьків, будуть просто знищені потомства ніякого не дадуть. У наведеному прикладі, із тисячі осіб потомство дадуть лише п'ять, хоча у практичних реалізаціях генетичних алгоритмів кількість батьків підбирається експериментально і може досягати 50% від розміру популяції, а інколи і більше.

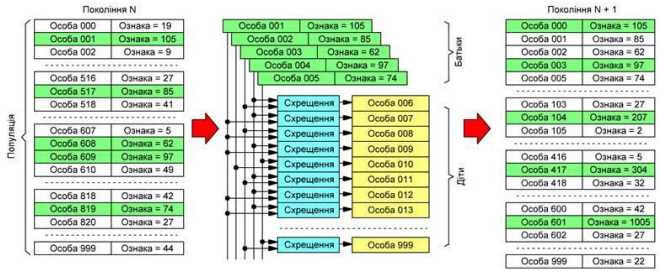
Основні поняття генетичного алгоритму



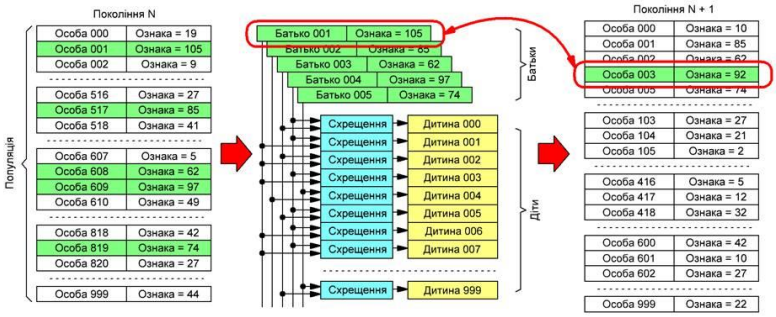
- **Батьки** – особи із поточного покоління, що мають найкращі ознаки і дадуть потомство (дітей)
- **Діти** – особи наступного покоління, утворені у результаті схрещення батьків
- **Схрещення** – процес утворення дітей від двох батьків

Після того, як будуть обрані найкращі представники поточного покоління, тобто батьки, починається процес створення дітей. Для створення дитини необхідно, щонайменше, двоє батьків, які обираються із батьківської групи випадковим чином. Батьки обов'язково повинні бути різними, створення дитини від двох екземплярів одного і того ж батька, як буде показано пізніше, не призведе до підвищення рівня різноманіття у популяції, і тому заборонено на рівні алгоритму створення нового покоління.

Створення нового покоління



Зверніть увагу на один момент. У реальному житті особи, навіть самі найкращі, помирають, тобто знищуються, у будь-якому випадку. Але, оскільки процес створення дітей має певний елемент випадковості, то може статися так, що навіть найкращі представники наступного покоління будуть гірші ніж їх попередники, і популяція замість того, щоб розвиватися, почне вироджуватися. У реальному житті ми з цим нічого не можемо зробити, а от у технічних системах нам нічого не варто зберегти найкращих представників поточного покоління і перенести їх у майбутнє. У такому випадку ми ніколи не втратимо найбільш оптимальне рішення, яке колись було знайдено. Більше того, такий метод дає можливість знаходити більш оптимальне рішення навіть через кілька поколінь.



Повне знищення батьків може призвести до виродження популяції

Таким чином потрібно сформувати певну кількість поколінь. У кожному поколінні обраються найкращі представники, які стають батьками і автоматично переходять у наступне покоління. На основі цих батьків створюються діти, які можуть бути кращі або гірші за своїх батьків. Та оскільки найкращі батьки обов'язково переходять у наступне покоління, то найкращий варіант рішення ніколи не загубиться і у кожному новому поколінні буде давати потомство, допоки потомство не стане краще за цього батька. Після проходження певної кількості поколінь, яке задається програмістом, найкраща особа стає переможцем у цій гонці за життя, і той варіант вирішення задачі, який у ній міститься, і вважається найбільш оптимальним.

Цей макроалгоритм поки є доволі простим і зрозумілим. Більше того, він має аналоги у реальному житті, тому нам не важко його уявити. Проте поки що незрозумілим, яким чином подібна поведінка призведе до збільшення рівня пристосованості усієї системи, тобто яким чином діти стають, або можуть стати, кращими за батьків. Але ж ми поки що не розглядали головну процедуру генетичного алгоритму – процес утворення нових членів популяції, тобто схрещення батьків, результатом якого стають діти. Незважаючи на те, що цей процес є набагато простішим ніж у живих істот, хоча б через те, що у осіб популяції немає статі і складних алгоритмів утворення батьківських пар, усе одно він є доволі складним.

Для кращого розуміння алгоритму схрещення давайте розглянемо конкретний приклад застосування генетичних алгоритмів, а саме – рішення задачі комівояжера. У класичній версії цієї задачі комівояжеру, якого можна назвати, наприклад, торговим агентом, необхідно об'їхати певний перелік міст таким чином, щоб побувати у кожному місті лише один раз, потім повернутися у точку виїзду, і при цьому довжина шляху повинна бути найкоротша. Задачі комівояжера у тому чи іншому вигляді використовуються для організації логістики та перевезень у великих компаніях, для побудови маршрутів оптимальних за часом, обсягом вантажу або витратами палива, у системах прийняття рішень, коли на вході є багато параметрів і потрібно знайти оптимальну послідовність дій, при організації конвеєрів на виробництві, при складанні туристичних маршрутів, при складанні розкладів і навіть для аналізу ефективності фінансових інструментів в економіці.

Задача комівояжера

Практичне застосування:

- Для організації логістики та перевезень у великих компаніях
- Для побудови оптимальних маршрутів за часом, обсягом вантажу або витратами палива
- У системах прийняття рішень, коли на вході є багато параметрів і потрібно знайти оптимальну ланцюжок дій
- При організації конвеєрів на виробництві
- При складанні туристичних маршрутів
- Для аналізу ефективності фінансових інструментів в економіці

Проблема рішення задачі комівояжера полягає у тому, що алгоритму, що дозволяв би швидко і точно вирішити цю задачу, у природі не існує – цю задачу можна вирішити лише шляхом повного перебору варіантів. Та річ у тому, що кількість можливих варіантів рішення задачі комівояжера має факторіальну залежність від кількості міст. Сучасний комп'ютер шляхом прямого перебору зможе визначити маршрут, у якому налічується 10 – 11 міст. Для того, що визначити найкоротшій маршрут, що має 12 міст, вже необхідно витратити приблизно один робочий день. А для того, щоб шляхом прямого перебору визначити маршрут, що містить 20 міст, потрібно кілька тисяч років. А от використання генетичних алгоритмів дозволяє вирішити цю задачу набагато швидше, хоч і точність її вирішення може бути не завжди максимальною.

Задача комівояжера

Кількість міст	Кількість варіантів
2	2
3	6
4	24
5	120
6	720
7	5040
8	40320
9	362880
10	3628800

Кількість міст	Кількість варіантів
11	39916800
12	479001600
13	6227020800
14	87178291200
15	1307674368000
16	20922789888000
17	355687428096000
18	6402373705728000
19	121645100408832000

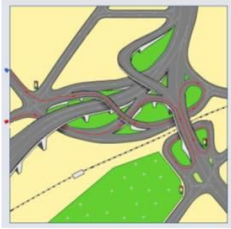
Вхідними даними для рішення задачі комівояжера є матриця відстаней між містами. Ця матриця містить нулі на головній діагоналі, оскільки для того, щоб поїхати у те саме місто, у якому ви зараз знаходитесь, не потрібно нікуди їхати. Розрізнюють симетричну та асиметричну матрицю відстаней. При симетричній матриці відстань від містами не залежить від напрямку руху, тобто якщо їхати, наприклад із Міста 1 у Місто 5 то відстань буде така ж сама як у випадку якщо їхати із Міста 5 у Місто 1.

Відстань між містами, км
(симетрична матриця)

Місто	Місто 1	Місто 2	Місто 3	Місто 4	Місто 5
Місто 1	0	10	20	50	100
Місто 2	10	0	30	20	70
Місто 3	20	30	0	75	11
Місто 4	50	20	75	0	15
Місто 5	100	70	11	15	0

Проте на практиці, це далеко не завжди так. Річ тому, що дороги, по яким слід проїжджати між містами можуть бути різні. Наприклад, на деяких транспортних розв'язках для того щоб повернути наліво, слід проїхати кілька додаткових кілометрів, у той час, як поворот направо не викликає збільшення довжини маршруту. Тому у зальному випадку, матриця відстаней між містами є асиметричною. Тобто якщо їхати із одного міста в інше то треба проїхати одну відстань, а якщо повертатися – то зовсім іншу.

Відстань між містами, км
(асиметрична матриця)



Для того щоб повернути наліво на деяких транспортних розв'язках потрібно проїхати кілька додаткових кілометрів

Місто	Місто 1	Місто 2	Місто 3	Місто 4	Місто 5
Місто 1	0	10	20	50	100
Місто 2	10	0	30	20	70
Місто 3	15	30	0	75	11
Місто 4	50	25	75	0	15
Місто 5	120	70	14	15	0

У даному випадку, для рішення задачі комівояжера за допомогою генетичних алгоритмів створюється популяція із певної кількості осіб, кожна з яких є рішенням задачі, тобто містить певний порядок об'їзду, який для першого покоління осіб заповнюється випадковими значеннями. У наведеному прикладі, головною ознакою, за якою потім будуть обирати батьків для наступного покоління і, відповідно, переможця, є довжина маршруту. Чим менша довжина маршруту, тим менше часу буде витрачено комівояжером на подорож і тим менше пального буде витрачено на рух транспортного засобу, у якому подорожує комівояжер. Зверніть увагу, що у прикладі, що ви бачите на екрані – коли потрібно об'їхати усього п'ять міст – задачу можна вирішити шляхом прямого перебору, оскільки існує усього 120 варіантів об'їзду. Але при більшій кількості міст кількість варіантів набагато більша ніж кількість осіб у популяції. У даному випадку, ми, для наочності, створили популяцію, що налічує 50 осіб, що також менше ніж кількість можливих варіантів розв'язання цієї задачі.

Створення першої популяції

Ознака - довжина марштуру

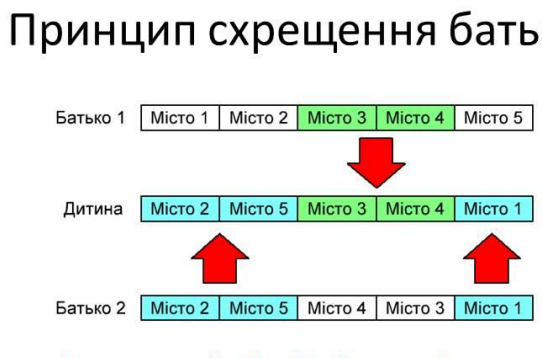
Особа 00	Місто 1	Місто 2	Місто 3	Місто 4	Місто 5	180 км
Особа 01	Місто 3	Місто 2	Місто 5	Місто 1	Місто 4	516 км
Особа 02	Місто 4	Місто 1	Місто 2	Місто 5	Місто 3	302 км
Особа 49	Місто 2	Місто 5	Місто 3	Місто 4	Місто 1	150 км

Особи для першої популяції створюються випадковим чином
Критерій вибору батьків – мінімальна довжина маршруту

Отже, ми випадковим чином створили перше покоління нашої популяції, обчислили ключову ознаку кожної особи – у даному випадку – довжину маршруту і обрали деяку кількість, наприклад, п'ять осіб, тобто варіантів об'їзду, що є найкоротшими. Тепер настав час створити нове покоління популяції. У неї автоматично попадають усі п'ять найкращих представників попередньої популяції, але іще 45 представників нам потрібно створити шляхом схрещування батьків. Відбувається це наступним чином.

У класичному варіанті реалізації генетичного алгоритму, із групи батьків випадковим чином обираються дві різні особи, які будуть схрещуватися. Далі від одного із батьків випадковим чином обирається фрагмент рішення, який копіюється у дитину. У прикладі, що ви зараз бачите на екрані, від першого батька було взято лише два елементи. Два – це випадкове

значення, тобто в одному схрещенні можна взяти третій та четвертий елемент, а у другому – від того ж самого батька можна взяти елементи від другого до четвертого, або від першого до третього і так далі. Елементи порядку об'їзду, що залишились після копіювання, беруться, відповідно, від другого батька. Отже, генератор випадкових чисел застосовують навіть при схрещуванні, тому результат схрещування одних і тих самих батьків кожного разу буде різним. Тобто, теоретично, від двох однакових батьків завжди будуть утворюватися різні діти.



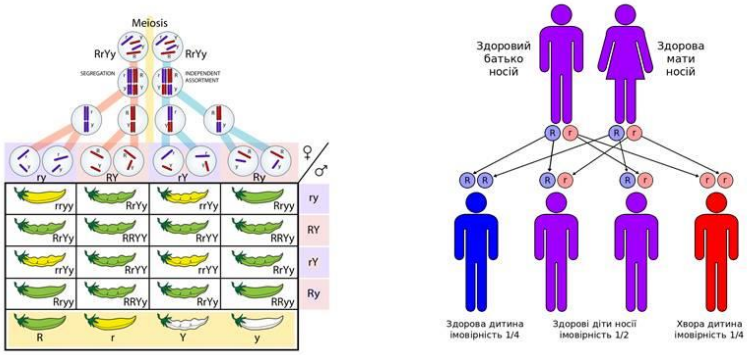
Дитина містить ознаки обох батьків, сформовані випадковим чином

Звісно, на практиці, усе буває не так добре, як у наведеному прикладі. Може бути таке, що обрані фрагменти батьківських осіб можуть мати однакові послідовності міст що призведе до неправильного формування дитини. Тому під час схрещування слід виконувати перевірку коректності послідовності дитини у своєчасно корегувати, наприклад, випадковим чином заповнюючи її містами, яких ще немає у дитини. Зверніть увагу, що це є особливістю конкретно саме цієї версії алгоритму, тобто генетичного алгоритму, призначеному для рішення задачі комівояжера. Це я веду до того, що алгоритм схрещування є унікальним для кожного генетичного алгоритму і залежить від конкретної задачі.

Обрані фрагменти батьківських осіб можуть мати однакові міста

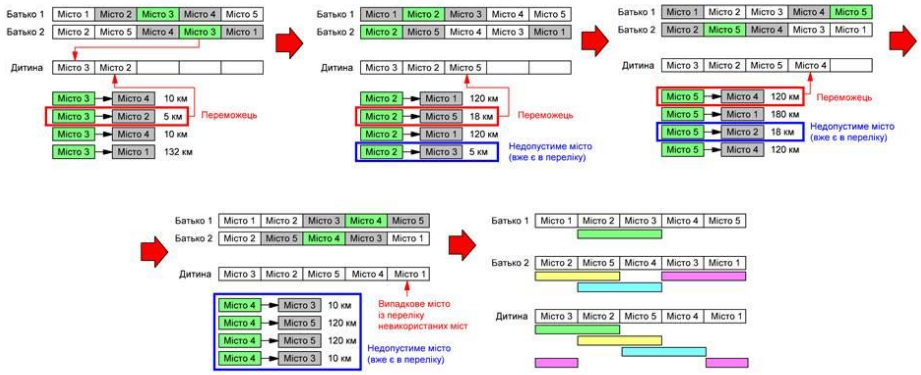
Той алгоритм схрещування, що був показаний раніше є універсальним, або класичним, або стандартним – називайте його як хочете. Суть його полягає у тому, що випадковим чином береться частина від однієї послідовності і об'єднується із частиною іншої послідовності. Саме так відбувається статеве розмноження у біологічних істот. Кожен нащадок живих істот містить у собі певну кількість хромосом від батька і певну від матері, які комбінуються випадковим способом. Саме це і забезпечує різноманіття форм всередині популяції, що значно підвищує її рівень адаптації до довколишнього середовища. Саме через схожість такого механізму створення нових осіб із механізмом розмноження живих істот, що відбувається на генному рівні, ці алгоритми і отримали назву «генетичних». До речі, у технічній літературі, присвяченій генетичним алгоритмам, осіб популяції інколи називають хромосомами. Проте, на мою особисту думку, для розуміння механізми роботи генетичного алгоритму використання терміну «особа» замість «хромосома» дозволяє більш простіше пояснити його динаміку.

Принцип статевого розмноження



Проте при вирішенні задачі комівояжера добре себе зарекомендував трішки інший алгоритм схрещування. У даному випадку, із однієї із батьківських хромосом випадковим чином береться певна позиція, яка стає першим елементом у послідовності дитини. Потім в обох батьківських особах знаходяться елементи, що відповідають тому ж самому місту. Після цього формується список із міст, що розташовані зліва і справа по відношенню до цієї позиції. Таких позицій є чотири. У загальному випадку усі позиції можуть бути різні, проте неминучі і повтори. Із цього переліку обирається місто, розташоване найближче до поточного міста. Саме воно стає наступним містом у новому маршруті. На першій стадії схрещування, коли у послідовності дитини є лише одне місто, найближче місто автоматично попаде у перелік, проте на наступних стадіях, може виникнути ситуація, коли найближче місто вже є у переліку. У цьому випадку це місто викреслюється із переліку можливих наступних міст і береться найближче місто, якого його ще немає і переліку міст дитини. Якщо ж виявиться, що усі можливі міста із батьківського списку вже є в переліку дитини, тоді складається окремий перелік міст, яких у цьому переліку ще немає, і з нього випадковим чином обирається любе вільне місто. В іншій версії схрещування, наступне місто із переліку вільних міст, що не входять до батьківських переліків сусідніх міст, обирається не випадковим чином а по критерію найближчої відстані до поточного міста. Такий алгоритм, як показує практика, дозволяє із більшою вірогідністю знайти найкоротший маршрут, тобто більш ефективно вирішити поставлену задачу. Зверніть увагу, що у даному випадку дитина містить лише розрізнені фрагменти своїх батьків, але не потрібно забувати, що ці фрагменти є найкращими.

Алгоритм схрещення батьків для генетичного алгоритму,
призначеному для вирішення задачі комівояжера



Проте це ще не все. Генетичні алгоритми не були б ефективними, якби діти формувалися лише за наведеними статичними правилами. Останнім кроком формування дитини є мутація, яка збільшує імовірність кращого пристосування до довколишнього середовища. Мутації,

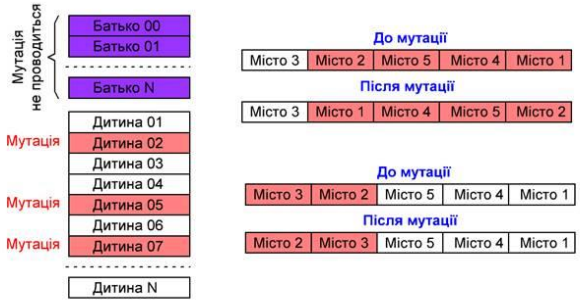
так само як і схрещення батьків, можуть відбуватися за різними алгоритмами. Наприклад, мутація може бути проведена безпосередньо після схрещення батьків, тобто одразу після формування дитини, або вже після створення усіх дітей у популяції. На кінцевий результат це впливає – впливає лише сам факт наявності мутації Одразу слід зауважити, що мутації підлягають на усі діти, а лише певна частина, наприклад, 10% від усієї кількості народжених дітей. І ще, мутації підлягають лише діти – батьки не повинні підлягати мутації, оскільки при мутаціях батьків є ризик втрати найкращого рішення, що існує на даний час.

Мутації

Випадкові мутації збільшують імовірність кращого пристосування до довколишнього середовища

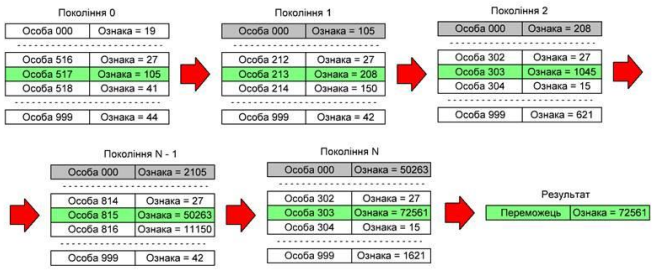
При рішення задачі комівояжера мутація проводиться наступним чином. В дитині випадковим чином виділяється фрагмент, у якому змінюється послідовність об’їзду міст. Початкове місце та кількість елементів, над якими буде проведена мутація, формуються випадковим чином, тому в одному випадку може бути замінено лише два елементи, а іншому – повністю весь порядок об’їзду може бути замінений на протилежний. В останньому випадку не забуваємо, що відстані між містами можуть залежати від напрямку проїзду, тому зміна напрямку об’їзду у тому ж самому кільцевому маршруті також може призвести до збільшення або зменшення його довжини.

Алгоритм проведення мутації



Власне кажучи – ось і весь принцип роботи генетичного алгоритму. Як бачите, майже увесь алгоритм заснований на генерації випадкових чисел. Випадковими є формування, першого покоління популяції, випадковим є вибір батьків для схрещення, напіввипадковим є сам процес схрещування і, нарешті, випадковим є процес мутації. Отже на всіх етапах використовується майже одна суцільна випадковість. І тому саме цікаве полягає у тому, що результат роботи генетичного алгоритму із високою вірогідністю, завдяки використанню саме цієї випадковості, знаходить оптимальне рішення набагато швидше, ніж у випадку використання методу прямого перебору.

Принцип роботи генетичного алгоритму



Практичне застосування генетичних алгоритмів показує, що обчислення маршруту, що налічує 30 міст, із використанням сучасних комп'ютерів потрібно лише кілька секунд, у той час як при вирішенні цієї задачі методом прямого перебору потрібно часу трішки менше ніж вічність. Проте до практичного рішення задач комівояжера цей генетичний алгоритм потрібно трішки доробити.

Задача комівояджера

Кількість міст	Кількість варіантів	Кількість міст	Кількість варіантів
2	2	11	39916800
3	6	12	479001600
4	24	13	6227020800
5	120	14	87178291200
6	720	15	1307674368000
7	5040	16	20922789888000
8	40320	17	355687428096000
9	362880	18	6402373705728000
10	3628800	19	121645100408832000

Почнемо з того, що на практиці комівояжер не може почати свою подорож із будь-якого міста, оскільки він зараз фізично знаходиться в одному із міст і не може телепортуватися у іншу точку простору. А порядок об'їзду, як ми бачити формується випадковим чином. Проте ця задача вирішується доволі просто. Для того, щоб комівояжер почав свою подорож із конкретного міста, достатньо лише циклічно змінити порядок об'їзду. Тобто послідовно зміщувати елементи порядку об'їзду допоки не першому місці не стане потрібне місто, наприклад, Місто 1. Це можна зробити один раз у самому кінці роботи алгоритму, коли вже обчислені усі покоління популяції і визначена особа переможець. Зверніть увагу, що циклічна перестановка міст у порядку об'їзду не змінює загальної довжини маршруту, оскільки він, у класичному варіанті задачі комівояжера, є кільцевим.

Встановлення конкретної точки виїзду

Циклічна перестановка міст не впливає на довжину маршруту

Більша складна ситуація, виникає коли комівояжер повинен почати свій маршрут в одному місці, а закінчити його – в іншому, і ці міста – початкове та кінцеве – заздалегідь визначені і не можуть змінюватися. На практиці така ситуація виникає коли, наприклад, автомобіль виїжджає із одного міста і повинен приїхати у пункт призначення і заїхати по дорозі у певні точки, які можуть розташовуватися доволі далеко від основної дороги. У даному випадку, потрібно змінити алгоритм формування порядку об'їзду як при створенні першого покоління популяції, так і при схрещенні батьків. Для цього першою і останньою точками генетичного алгоритму завжди є конкретні міста, наприклад початок і кінець маршруту, а усі інші точки вже необхідно формувати або випадковим способом – при створенні першого покоління популяції, або за спеціальним алгоритмом схрещення батьків – при створенні наступних поколінь.

Обчислення некільцевих маршрутів, що мають конкретний початок і кінець

Фіксовані значення	Випадкові значення			Фіксовані значення
Початок	Місто 2	Місто 1	Місто 3	Кінець

Фіксовані значення	Результат схрещення батьків			Фіксовані значення
Початок	Місто 3	Місто 1	Місто 2	Кінець

Фіксовані точки прописуються жорстко алгоритмі

Так само жорстко можуть бути прописані і певні фрагменти маршруту. Наприклад, при доставці товарів вантажівка, завантажена на складі, перед тим як виїхати на маршрут повинна спершу заправитись на певній заправці. У даному випадку першим і останнім елементом маршруту є склад, а другим містом у маршруті обов'язково повинна бути заправка. А от усі інші елементи маршруту вже формуються відповідно до роботи генетичного алгоритму.

Обчислення маршрутів, що мають фіксовані фрагменти

Фіксовані значення		Випадкові значення		
Склад	Заправка	Місто 1	Місто 3	Місто 2

Фіксовані значення		Результат схрещення батьків		
Склад	Заправка	Місто 3	Місто 1	Місто 2

Фіксовані точки прописуються жорстко алгоритмі

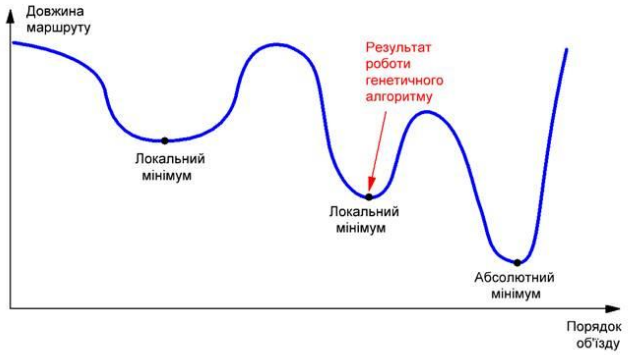
У деяких випадках фіксовані фрагменти об'їзду можуть бути всередині маршруту. Наприклад, в одній компанії, де мені довелось колись впроваджувати систему розрахунку оптимальних маршрутів на основі генетичних алгоритмів, було кілька VIP-клієнтів, які після доставки товару та розвантаження автомобіля давали експедитору доволі велику суму грошей. Оскільки вести самостійно подібну суму було небезпечно, то на це місце приїжджав спеціальний наряд поліції, який супроводжував автомобіль до банку. У даному випадку, для того щоб їхати до інших клієнтів разом із поліцією вже могла іти мова, тому після такого VIP-клієнта у маршруті завжди повинен був бути банк. А от після заїзду у банк можна було заїжджати і до інших клієнтів, але вже без поліції. Єдиним обмеженням таких маршрутів, хоч і не принциповим, було те, що у маршруті повинен бути тільки один VIP-клієнт і, відповідно, один банк. Це обмеження не є принциповим і його можна було б зняти, проте ніхто ніколи не ставив в одному маршруті двох VIP-клієнтів – загальна вага вантажу та час на розвантаження автомобіля фізично не дозволяли це зробити.

Обчислення маршрутів, що мають фіксовані порядки об’їзду

Фіксовані порядки об’їзду прописуються жорстко алгоритмі

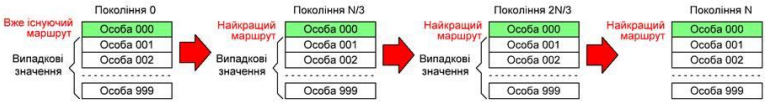
Ще однією особливістю використання генетичних алгоритмів при вирішенні задачі комівояжера є неможливість збільшення довжини вже існуючого маршруту. А таке може бути у випадку, коли, наприклад знайти оптимальне рішення з першого разу не вдалося. При використанні генетичних алгоритмів таке буває і доволі часто, оскільки кількість можливих комбінацій, які беруть участь у генетичних алгоритмах, набагато менше за кількість можливих комбінацій, які взагалі існують. Це призводить до того, що при вирішенні задачі комівояжера результати роботи генетичних алгоритмів доволі швидко можуть збігатися до певного локального мінімуму із якого вони зможуть вибратися лише у випадку появи вдалої мутації, яка може і не відбутися.

Результат роботи генетичного алгоритму



Для запобігання цьому використовують кількаразовий перезапуск усієї популяції. Крім цього, коли створюють перше покоління, то випадковим чином формують усі особи окрім першої, яку формують відповідно до маршруту, який вже існує. Це запобігає втраті вже існуючого оптимального рішення, яке, наприклад, може сформувати людина вручну. Коли відбувається перезапуск популяції, тоді знищують усіх існуючих осіб, окрім самої першої, яка, зазвичай містить найкращий варіант рішення, що вже було знайдено. Якщо ж генетичний алгоритм на зміг знайти рішення більш оптимальне ніж те, що було введено в нього із самого початку, наприклад людиною, то воно так і залишиться найкращим і не буде втрачено під час роботи. Отже, випадковість та штучний інтелект це дуже добре, але можливість штучного втручання людини із природним інтелектом також слід передбачити.

Періодичне перезавантаження генетичного алгоритму



Періодичне перезавантаження генетичного алгоритму збільшує імовірність знаходження глобального мінімуму (максимуму)

Різних модифікації задачі комівояжера є доволі багато. Наприклад, інколи потрібно приїхати у місто у певний проміжок часу, інколи потрібно враховувати час знаходження у місті, інколи із одного міста потрібно обов'язково поїхати у друге конкретне місто. При цьому необхідно обов'язково враховувати особливості транспортних розв'язок та мінімізувати кількість лівих поворотів на маршруті. Усе у кінцевому рахунку дозволить зменшити витрати пального або час, що буде проведений у дорозі.

Можливі варіанти задачі комівояжера

- Приїхати у місто у певний проміжок часу
- Урахувати час на знаходження у місті
- Урахувати особливості VIP-міст (після цього міста обов'язково потрібно приїхати у інше конкретне місто)
- Урахувати усі особливості транспортних розв'язок
- Мінімізувати кількість лівих поворотів
- **Зменшити витрати пального або час у дорозі**

35

Варіантів практичного застосування задачі комівояжера, і, відповідно, генетичних алгоритмів є доволі багато. Самим яскравим прикладом їх застосування є використання їх для розрахунку оптимальних маршрутів доставки товарів. Коли ми приходимо у магазин, наприклад, у супермаркет, то бачимо на полицях певні товари і зовсім не замислюємося над тим, яким чином ці товари були доставлені. А цю задачу кожного дня вирішує певна категорія підприємств, які називаються дистриб'юторами. Працюють вони наступним чином.

Де потрібні практично вирішувати задачі комівояжера?

36

Ви знаєте, що є виробники товару – це великі підприємства, що мають великі обсяги виробництва, і, відповідно, великі обсяг продаж. А ми купуємо товар у місцях, що називаються торговими точками.
Торгові точки можуть бути самим різними. Це можуть бути великі торгівельні мережі, невеликі генделики, кафе, бари, ресторани або взагалі конкретні люди, зареєстровані як фізичні особи-підприємці. Зрозуміло, що крупному виробнику не цікаво працювати із малими реалізаторами на кшталт ФОП «Тьотя Соня», що торгує на одеському Привозі і спроможна за рік продати таку кількість товару, яку крупна торгівельна мережа продає за кілька годин. Саме для цього існують дистриб'ютори товару.

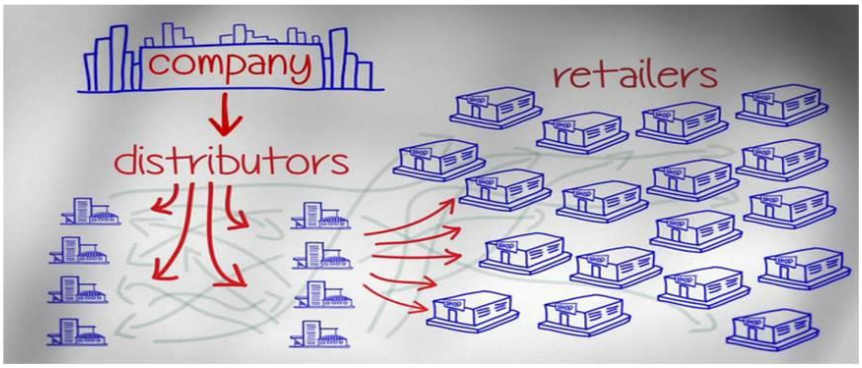
Типи торгових точок

Крупна торгівельна мережа

ФОП «Тьотя Соня з Привоза»

Вони є проміжною ланкою між виробником і конкретним реалізатором і призначені, у першу чергу, для забезпечення товарами представників малого та середнього бізнесу, які через свої невеликі обсяги продаж, не можуть взаємодіяти безпосередньо із виробниками. У даному випадку дистриб'ютор купує у виробника велику партію товару, який потім розподіляє малими частинами по конкретним невеликим торговим точкам у своєму районі.

Як попадає товар у торгових точок?



39

Отже, однією із задач дистриб'ютора є доставка товару по конкретним торговим точкам, відповідно до їх замовлень. Одразу необхідно зауважити, що такий спосіб доставки потребує меншої кількості пального і меншої кількості часу, ніж у випадку, коли малому підприємцю необхідно самостійно їздити за товаром, тому малі та середні підприємці охоче співпрацюють із дистриб'юторами. У цьому випадку, дистриб'ютору потрібно щоденно вирішувати певну кількість задач комівояжера для того, щоб доставити товар у певні торгові точки, кількість яких разом із кількістю товару, що ними замовляється, кожен день різна.

Щоденна задача дистриб'юторської компанії

- Транспортний парк – 80 автомобілів вантажопідйомністю від 1 т до 15 т
- Кожен із 80 автомобілів може робити до 3 ходок
- Близько 20 торгових точок у одній ходці

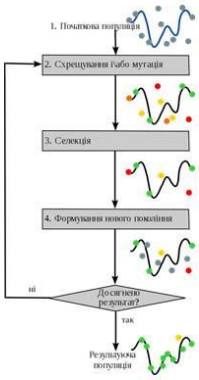
Загалом 80·3·20 ≈ 5 000 клієнтів (кожен день різні торгові точки та різна кількість товару)

Для вирішення цієї задачі вручну необхідно 5 логістів, що починають роботу об 19.00 та закінчують близько 7.00

40

Наприклад, одна із дистриб’юторських компаній, де мені колись довелося впроваджувати програмне забезпечення для скорочення витрат палива, у якому використовувались генетичні алгоритми, мала приблизно 80 вантажівок вантажопідйомністю від 1 до 15 тон, кожна з яких за робочий день могла зробити до трьох ходок. В одній ходці могло бути до 20 точок. Отже за день подібна компанія могла завести товар до 5000 торгових точок, які кожен день були різними. До впровадження програмного забезпечення цю задачу вручну вирішувало 5 логістів, що починали працювати приблизно із сьомої години вечора і закінчували роботу близько сьомої години ранку. Звісно, що при ручному формуванні про побудову оптимальних маршрутів мова навіть не йшла – цим логістам вистачало часу тільки на фізичне формування 240 маршрутів. Встановлення програмного забезпечення дозволило не тільки розвантажити логістів, які тепер могли їхати додому набагато раніше, а ще й істотно скоротити витрати паливу на доставку товару, оскільки програмне забезпечення на основі генетичних алгоритмів розв’язує подібні задачі швидше та оптимальніше ніж людина.

Налаштування генетичного алгоритму



- Правило обчислення ключової ознаки
- Розмір популяції
- Кількість поколінь
- Алгоритм схрещення батьків
- Рівень мутації
- Кількість перезавантажень популяції

Отже генетичні алгоритми є доволі корисним напрямком штучного інтелекту, яка має певну область для практичного застосування. Наостанок я ще раз хочу звернути увагу на те, що кожен генетичний алгоритм є унікальним, хоча загальний алгоритм його функціонування приблизно однаковий у всіх випадках. Тому такі ключові параметри як правило обчислення ключової ознаки, численність та тривалість життя популяції, алгоритм схрещення батьків, рівень мутації дітей та кількість перезавантажень залежить від конкретної задачі, і тому вам їх доведеться експериментувати самостійно. А це означає, що мені залишається лише побажати вам терпіння та наснаги у цьому складному, але вельми корисному та доволі прибутковому напрямку розвитку штучного інтелекту. Дякую за увагу!

Лекція 8. Використання нейронних мереж в засобах кібербезпеки та захисту інформації

На сьогоднішній день поняття "штучний інтелект" (ШІ) стало невід'ємною частиною нашого повсякденного життя. Хоча системам з елементами штучного інтелекту ще далеко до повного розуміння проблеми та знаходження її рішень, у сферах оперативних завдань та виявлення аномалій у процесах вони перевершують людські можливості і компетентність. Штучний інтелект відіграє ключову роль у виявленні й оцінці помилок, які можуть бути допущені людьми. В майбутньому системи на основі штучного інтелекту у сфері кібербезпеки зможуть захищати організації від інтернет-загроз, виявляти шкідливі програми, забезпечувати високі стандарти безпеки та сприяти розробці ефективних стратегій запобігання та відновлення після кібератак.

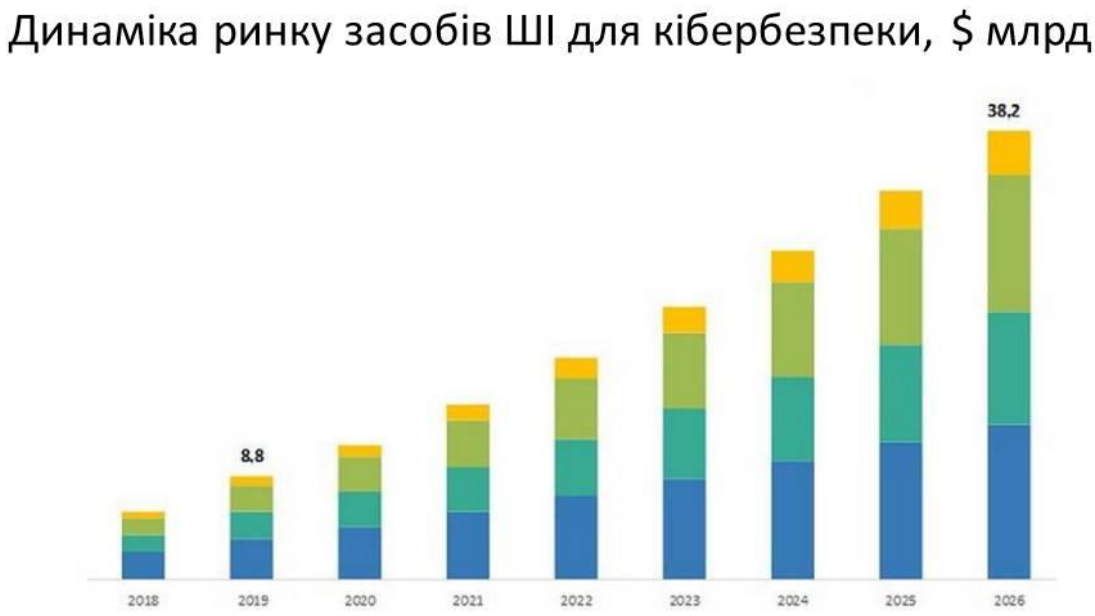
За оцінкою Gartner, витрати на системи інформаційної безпеки і управління ризиками у 2022-му році збільшилися до $ 174 млрд, з них приблизно $ 50 млрд будуть спрямовані на захист клієнтських систем. Продажі хмарних платформ і додатків для забезпечення безпеки виростуть до $ 1,63 млрд в 2023-му році, а систем забезпечення безпеки додатків до $ 4,5 млрд. Зростає і ринок послуг в області інформаційної безпеки, за останній рік він збільшився з $ 62 млрд до $ 66,9 млрд. Однак самі по собі гроші не можуть вирішити питання. Більшість фахівців з інформаційної безпеки сьогодні перевантажені аналізом журналів, запобіганням спроб злому, розслідуванням можливих випадків шахрайства тощо. Дефіцит кадрів великий, тому в індустрії безпеки все з більшою надією дивляться на рішення в області штучного інтелекту.

Витрати на системи інформаційної безпеки і управління ризиками (2023 рік)

- 174 млрд. USD – загальні витрати
- 50 млрд. USD – захист клієнтських систем
- 1,63 млрд. USD – хмарні платформи і додатки для забезпечення безпеки
- 1,63 млрд. USD – системи забезпечення безпеки додатків
- 66,9 млрд. USD – ринок послуг в області інформаційної безпеки

Через великий брак фахівців в галузі кібербезпеки та захисту інформації останньою надією залишаються лише рішення в галузі штучного інтелекту

За оцінкою MarketsandMarkets, в 2019-2026 рр. зростання ринку засобів ШІ для забезпечення кібербезпеки буде рости в середньому на 23,3% в рік, з $ 8,8 млрд до $ 38,2 млрд. Однак, незважаючи на уявлення щодо потенційних можливостей засобів штучного інтелекту їх застосування залишається переважно епізодичним та несистематизованим. На даний час у кібербезпеці відсутня загальна концепція запровадження штучного інтелекту, не визначені найважливіші методи штучного інтелекту, які можуть бути використані у кібербезпеці, та не встановлено роль, яку можуть відігравати ці методи (особливо, що стосується машинного навчання, дата-майнінгу, глибокого навчання та експертних систем) для захисту організацій у кіберпросторі.



Штучний інтелект – це сукупність теоретичних та практичних підходів у галузі інформаційних технологій, які передбачають створення систем, що можуть функціонувати розумно та незалежно, подібно до механізму прийняття рішень у мозку людини. Завдяки ШІ машина зможе навчатися досвіду обробляючи великі обсяги даних та розпізнаючи шаблони у них. Наприклад, Apple Siri, розпізнавання облич та самокерований автомобіль засновані на машинному навчанні та обробці природних мов, що є окремою галуззю ШІ. Крім того, ШІ включає багато суміжних областей та технологій, таких як машинне навчання, глибоке навчання, нейронні мережі, обробка природних мов та інші.

Для більш глибокого розуміння, систематизуємо основні напрямки досліджень у галузі штучного інтелекту, які реалізовані практично і якими ми користуємося майже щодня:

Машинне навчання – це множина технологій, які дозволяють комп‘ютерам мислити за допомогою математичних алгоритмів на основі зібраних даних та конкретних інструкцій та правил. Замість програмування комп‘ютера на кожному кроці, цей підхід реалізує технології, які дозволяють йому поетапно вчитися на даних без інструкцій програміста. Прикладом машинного навчання є віртуальні особисті помічники, які допомагають знаходити інформацію та дають вказівки щодо конкретних завдань, коли їх запитують через голос. Іншими прикладами є: відеоспостереження, яке відстежує незвичну поведінку, послуги соціальних медіа для зв‘язку з ―людьми, яких ви можете знати‖ на основі постійного самонавчання відповідно до уподобань людей.

Нейронні мережі (*Глибоке навчання*) являє собою подальший розвиток машинного навчання і є комп‘ютерною моделлю, яка виконує завдання класифікації безпосередньо із зображень, тексту чи звуку. Ця модель реалізується з використанням широкого набору багатовимірних даних та багатошарової архітектури нейронної мережі. Глибоке навчання досягає точності розпізнавання при застосуванні більш складних моделей. Прикладами застосування цієї моделі у реальному житті є автоматизоване керування автомобілем, розпізнавання облич та знаків, промислова автоматизація та ін.

Природна обробка мови, сукупність методів, які дозволяють машині аналізувати мову людини, зокрема у чат-ботах для програм підтримки клієнтів, використовують машинне навчання та обробку природної мови.

ШІ застосовує технології, які можна використовувати у багатьох галузях промисловості, таких як фінансові установи, освіта та охорона здоров‘я. Зважаючи на вищевикладене цілком доцільним є потенційне використання методів ШІ для забезпечення кібербезпеки організацій та підприємств.

Основні напрямки ШІ

- **Машинне навчання** – множина технологій, які дозволяють комп'ютерам мислити за допомогою математичних алгоритмів на основі зібраних даних та конкретних інструкцій та правил
- ***Нейронні мережі (Глибоке навчання)*** – подальший розвиток машинного навчання, що є комп'ютерною моделлю, яка виконує завдання класифікації безпосередньо із зображень, тексту чи звуку
- **Природна обробка мови** – сукупність методів, які дозволяють машині аналізувати мову людини

Без сумніву, величезний потенціал технологій штучного інтелекту може бути використаний для підвищення кібербезпеки. Кількість даних, що генеруються в сучасному світі, постійно збільшується, при цьому інформація зберігається та передається у різній формі з використанням мережі Інтернет. Більше того, безпечна передача даних відіграє життєво важливу роль у боротьбі з кіберзлочинами, що досягається шляхом дотримання принципів кібербезпеки. З ростом прогресу в галузі інформаційних технологій кіберпростір перетворюється на полігон для вчинення різних кіберзлочинів а, згідно з поглядами військових фахівців, стає четвертим (разом з сушею, морем, повітрям) театром воєнних дій.
Штучний інтелект у кібербезпеці – це достатньо широка область знань, яка потенційно може використовуватись в організаціях для зменшення ризиків та збільшення доходу, виявлення кіберзагроз та шахрайства. Відстежувати нові віруси та шкідливе програмне забезпечення стає все складніше і тому засоби на основі технологій штучного інтелекту здатні полегшити виявлення та реагування на загрози, використовуючи статистичні дані про кібератаки, щоб визначити найкращий перелік дій щодо протидії. ШІ може бути більш ефективним при виявленні шкідливого програмного забезпечення, ніж людина. ШІ впроваджується в організаціях з кількома рівнями безпеки, такими як Інформування про безпеку та Управління подіями і це допомагає поліпшити аналітикам безпеки виявлення будь-яких загроз усередині мережі організації.
Чим швидше можна виявити порушення цілісності даних, тим менші витрати на їх відновлення. Постійне збільшення часу на усунення порушень пов'язане зі збільшенням тяжкості зловмисних нападів, які зазнали більшість компаній. Автоматизація безпеки та інтелектуальні засоби, що забезпечують контроль у ситуаційному центрі безпеки, можуть допомогти поліпшити здатність організації зменшити збитки, спричинені порушеннями.
У рішеннях з кібербезпеки використовуються багато типів програм ШІ, зокрема SIEM системи, різноманітні фільтри спаму, засоби захищеної автентифікації користувачів та засоби прогнозування інцидентів злому. Ці програми навчаються за допомогою бази даних попередньої поведінки та можуть ідентифікувати кожну окрему поведінку як шкідливу чи ні. За даними компанії IBM, збитки від порушення даних у всьому світі будуть знижені, якщо організації застосовуватимуть автоматизовані рішення безпеки. Організації, які не застосовували автоматизацію безпеки, зазнали витрат на порушення, які були на 95 % вищими, ніж порушення в організаціях з повністю розгорнутою автоматизацією.

Потенційні можливості методів штучного інтелекту
для застосування у сфері кібербезпеки

- Експертні системи
- Машинне навчання
- Нейронні мережі (глибоке навчання)
- Дата майнінг
- Інтелектуальні агенти

Експертні системи. Експертні системи є одним з найвідоміших інструментів штучного інтелекту і являють собою програмні пакети, які допомагають отримати відповіді на запити, які надає клієнт або надає інший пакет програм. Ці системи включають вміст знань, в якому знання експертів зберігаються в певній галузі застосування. Ці системи також включають механізм міркувань для доступу до відповідей з урахуванням наданої інформації та іншої додаткової інформації стосовно умов навколишнього середовища.
Експертна система програмується для пошуку відповідей на запити в певній області застосування, що відображаються або користувачем, або іншим продуктом. У кібербезпеці вони застосовуються для вибору заходів безпеки та визначення шляхів використання обмежених активів. Експертні системи безпеки допомагають персоналу організацій у боротьбі з кібератаками. Це здійснюється шляхом звірки логів атаки з базою знань у випадку, якщо це відомий процес, або її ігнорування, коли процес невідомий. У разі відсутності такої процедури в базі знань, експертна система використовує алгоритми механізму виведення та знаходить наближене рішення на основі досвіду.

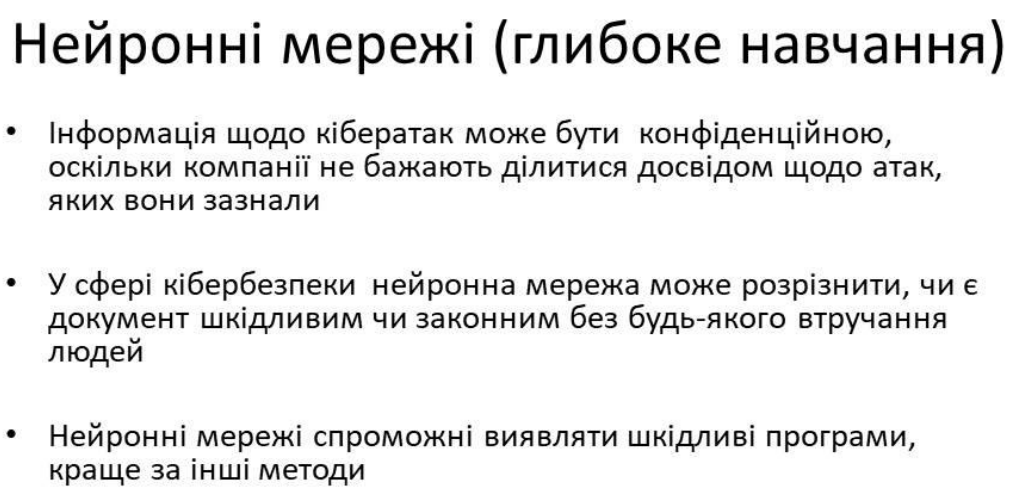
Експертні системи

- Програмні пакети, які допомагають отримати відповіді на запити, які надає клієнт або надає інший пакет програм
- У кібербезпеці експертні системи застосовуються для вибору заходів безпеки та визначення шляхів використання обмежених активів
- Експертні системи безпеки допомагають персоналу організацій у боротьбі з кібератаками шляхом звірки логів атаки з базою знань у випадку, якщо це відомий процес, або її ігнорування, коли процес невідомий

Машинне навчання. Машинне навчання – це область штучного інтелекту, яка дозволяє комп‘ютеру навчатись, використовуючи дані зразків (паттернів), не запрограмовані передбачити кожну можливу ситуацію. «Два найпоширеніші типи машинного навчання – це навчання з учителем та без учителя. Навчання з учителем використовується, коли доступний набір достовірно відомих екземплярів атак, зокрема, для вирішення проблем класифікації. Мета контрольованого навчання – навчити комп'ютер передбачати значення або точно класифікувати вхідний екземпляр атаки. Навчання без учителя використовується, коли набір достовірних даних недоступний. Кластеризація – це неконтрольована техніка навчання, яка приводить до подібних випадків групування в кластери. Кластеризація використовується для виявлення закономірностей у даних. У деяких випадках кластеризація виконується для класифікації немаркованого набору даних та використання отриманого класифікованого набору даних для контрольованого навчання.

Оскільки загрози кібербезпеки постійно змінюються та розвиваються, необхідна автоматизована та негайна реакція. Отже, методи машинного навчання, особливо глибоке навчання, яке не обов'язково вимагає попереднього навчання або опори на попередні класифікації, надані експертами, можуть бути особливо життєво важливими при застосуванні підходів ШІ до кібербезпеки.

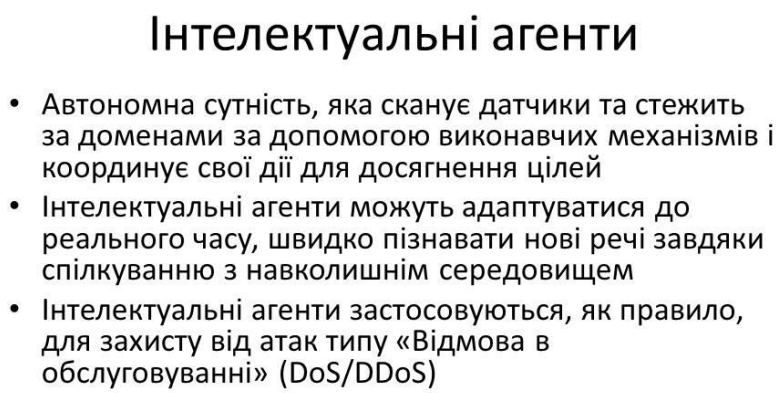
Нейронні мережі (глибоке навчання). Відсутність великих масивів даних щодо кібератак є загальним викликом у дослідженнях з кібербезпеки. Часто це пояснюється вимогами щодо конфіденційності, коли компанії не бажають ділитися досвідом щодо атак, яких вони зазнали, але, разом з тим, база відомих загроз поступово все ж таки наповнюється, що дає можливість застосовувати методи глибокого навчання. Основою для таких методів є великі і часто незбалансовані набори даних, які часто використовуються для ручної кластеризації. У сфері кібербезпеки нейронна мережа може розрізнити, чи є документ шкідливим чи законним без будь-якого втручання людей. Ця технологія дозволяє виявляти шкідливі програми, демонструючи при цьому кращі результати у порівнянні з іншими методами.



Дата майнінг. Дата майнінг – це пошук суттєвих закономірностей та тенденцій у великій базі даних. Методологія аналізу даних спрямована на отримання цінної інформації та пошук прихованих закономірностей з величезної кількості баз даних, які не можуть бути виявлені статистичними методами. Це широка область досліджень, яка включає машинне навчання, бази даних, статистику, експертні системи, візуалізацію, високопродуктивні обчислення, нейронні мережі та методи представлення знань. Дата майнінг підтримується хостом, який фіксує дані різними способами (наприклад, кластеризація, класифікація, аналіз посилань, узагальнення, регресійні моделі та аналіз послідовностей).

Основні приклади програм для дата майнінгу для кібербезпеки:
– методи виявлення атипових видів діяльності;
– аналіз посилань для відстеження вірусів;
– класифікація та групування декількох кібератак на основі їх профілів;
– прогнозування можливих майбутніх атак на основі отриманої інформації.

Інтелектуальні агенти. Інтелектуальний агент (ІА) – це автономна сутність, яка сканує датчики та стежить за доменами за допомогою виконавчих механізмів і координує свої дії для досягнення цілей. Інтелектуальний агент також може вивчати або використовувати інформацію для досягнення своїх цілей. Агенти можуть адаптуватися до реального часу, швидко пізнавати нові речі завдяки спілкуванню з навколишнім середовищем, а також мати можливість зберігання та відновлення моделей на основі пам'яті. Інтелектуальні агенти застосовуються, як правило, для захисту від атак типу «Відмова в обслуговуванні» (DoS/DDoS). Крім того, вони є ефективними при пошуку необхідної інформації у мережі, розподіленої обробки даних та ін.



Практика застосування ШІ у кібербезпеці. Ступінь зацікавленості служб інформаційної безпеки в штучному інтелекті залежить від галузі застосування. Консалтингова компанія Capgemini опублікувала результати опитування 850 керівників вищої ланки з 10 країн (Австралії, Великобританії, Німеччини, Індії, Італії, Іспанії, Нідерландів, США, Франції, Швеції). При цьому 20% респондентів займали пост IT-директора, 10% – керівника служби IT-безпеки. Компанії представляли сім сфер діяльності – виробництво споживчих товарів, ритейл, банківський сектор, страхування, автомобілебудування, ЖКГ та телекомунікації. За оцінкою Capgemini, якщо до 2019 г. лише кожна п'ята організація використовувала штучний інтелект для кібербезпеки, то 2020 році таких організацій вже понад 60%. Майже половина опитаних (48%) заявила, що бюджети на ШІ в області кібербезпеки збільшаться в 2021 фінансовому році в середньому на 29%. Пріоритетні варіанти використання ШІ для

кібербезпеки – забезпечення безпеки мережі і захист даних. Рішення інтернету речей поки відстають, але і з'явилися вони лише в останні роки.

Рівень фінансування сфер застосування засобів штучного інтелекту

№	Сфера застосування засобів ШІ	Ступінь використання фінансів на захист, %
1.	Мережева безпека	75
2.	Безпека даних	71
3.	Безпека кінцевих точок	68
4.	Безпека систем ідентифікації	65
5.	Безпека додатків	64
6.	Хмарна безпека	59
7.	Безпека Інтернету речей	53

Огляд методів та стратегій кібербезпеки засобами штучного інтелекту

Методи машинного навчання

Методи машинного навчання відіграють критичну роль у кібербезпеці, дозволяючи автоматизовано аналізувати великі обсяги даних та виявляти закономірності й аномалії, що можуть свідчити про кібератаки.

«Супервізоване» або «кероване» навчання (англ. Supervised Learning) використовується для класифікації даних, де система навчається на маркованих даних (наприклад, нормальні та аномальні дії) і потім застосовує цей досвід для нових даних. Приклади алгоритмів: логістична регресія, дерева рішень, SVM (Support Vector Machines)

«Ненаглядне» або «некероване» навчання (англ. Unsupervised Learning) використовується для виявлення невідомих патернів у даних без попереднього маркування. Приклади алгоритмів: кластеризація (k-means, DBSCAN), алгоритми зниження розмірності (PCA, t-SNE)

Глибоке навчання (англ. Deep Learning) включає використання нейронних мереж для складних завдань, таких як розпізнавання зображень або обробка природної мови. Рекурентні нейронні мережі (RNN) та згорткові нейронні мережі (CNN) є популярними методами для аналізу послідовних даних та зображень відповідно.

Методи машинного навчання

«Супервізоване» або «кероване» навчання (англ. Supervised Learning)	**Використовується для класифікації даних, де система навчається на маркованих даних (наприклад, нормальні та аномальні дії) і потім застосовує цей досвід для нових даних. Приклади алгоритмів: логістична регресія, дерева рішень, SVM (Support Vector Machines)**
«Ненаглядне» або «некероване» навчання (англ. Unsupervised Learning)	Використовується для виявлення невідомих патернів у даних без попереднього маркування. Приклади алгоритмів: кластеризація (k-means, DBSCAN), алгоритми зниження розмірності (PCA, t-SNE)
Глибоке навчання (англ. Deep Learning)	Включає використання нейронних мереж для складних завдань, таких як розпізнавання зображень або обробка природної мови. Рекурентні нейронні мережі (RNN) та згорткові нейронні мережі (CNN) є популярними методами для аналізу послідовних даних та зображень відповідно.

Аналіз поведінки користувачів (UBA)

User Behavior Analytics (UBA) використовує ШІ та ML (Machine Learning) для моніторингу та аналізу поведінки користувачів у реальному часі з метою виявлення аномалій, які можуть свідчити про внутрішні загрози або компрометацію облікових записів.
Моніторинг дій збирає дані про дії користувачів, включаючи логін, доступ до файлів, використання мережевих ресурсів та інші операції.
Виявлення аномалій використовує моделі ML для порівняння поточної поведінки з історичними даними та виявлення відхилень, які можуть бути ознаками загрози.
Адаптивність системи UBA можуть адаптуватися до змін у поведінці користувачів, забезпечуючи постійне оновлення моделей для більш точного виявлення загроз.

Аналіз поведінки користувачів (UBA)

Моніторинг дій	Збирає дані про дії користувачів, включаючи логін, доступ до файлів, використання мережевих ресурсів та інші операції
Виявлення аномалій	Використовує моделі ML для порівняння поточної поведінки з історичними даними та виявлення відхилень, які можуть бути ознаками загрози
Адаптивність системи	UBA можуть адаптуватися до змін у поведінці користувачів, забезпечуючи постійне оновлення моделей для більш точного виявлення загроз

Системи виявлення вторгнень (IDS)
Intrusion Detection Systems (IDS) використовують ШІ для аналізу мережевого трафіку та виявлення підозрілих дій, які можуть свідчити про вторгнення.
Мережеві IDS (NIDS) моніторять мережевий трафік і аналізують пакети даних для виявлення аномалій. Застосовуються методи сигнатурного та евристичного аналізу.
Хостові IDS (HIDS) моніторять активність на окремих пристроях, такі як журнали подій, цілісність файлів, системні виклики, для виявлення вторгнень.
Гібридні системи комбінують NIDS та HIDS для більш комплексного підходу до виявлення загроз.

Системи виявлення вторгнень (IDS)

Мережеві IDS (NIDS)	Моніторять мережевий трафік і аналізують пакети даних для виявлення аномалій. Застосовуються методи сигнатурного та евристичного аналізу
Хостові IDS (HIDS)	Моніторять активність на окремих пристроях, такі як журнали подій, цілісність файлів, системні виклики, для виявлення вторгнень
Гібридні системи	Комбінують NIDS та HIDS для більш комплексного підходу до виявлення загроз

Аналіз журналів подій (SIEM)
Security Information and Event Management (SIEM) системи об'єднують та аналізують журнали подій з різних джерел для виявлення загроз і забезпечення відповідності вимогам безпеки.
Централізоване збирання даних збирають дані з мережевих пристроїв, серверів, додатків та інших джерел.
Кореляція подій використовують алгоритми для кореляції подій з різних джерел, що дозволяє виявити складні атаки, які можуть бути непомітні при аналізі окремих подій.
Робота в режимі реального часу забезпечують моніторинг у реальному часі та автоматичне реагування на виявлені загрози.

Аналіз журналів подій (SIEM)

Централізоване збирання даних	**Збирають дані з мережевих пристроїв, серверів, додатків та інших джерел**
Кореляція подій	Використовують алгоритми для кореляції подій з різних джерел, що дозволяє виявити складні атаки, які можуть бути непомітні при аналізі окремих подій
Робота в режимі реального часу	Забезпечують моніторинг у реальному часі та автоматичне реагування на виявлені загрози

Технології Big Data
Технології Big Data дозволяють обробляти та аналізувати великі обсяги даних з високою швидкістю, що є критичним для виявлення та реагування на кіберзагрози.
Розподілена обробка – використання платформ, таких як Hadoop та Apache Spark, для обробки великих обсягів даних у розподіленому середовищі.
Аналіз у реальному часі – використання технологій потокової обробки, таких як Apache Kafka, для аналізу даних у реальному часі.
Інтеграція даних – здатність інтегрувати дані з різних джерел, включаючи мережевий трафік, журнали подій, соціальні мережі та інші джерела.

Технології Big Data

Розподілена обробка	**Використання платформ, таких як Hadoop та Apache Spark, для обробки великих обсягів даних у розподіленому середовищі**
Аналіз у реальному часі	Використання технологій потокової обробки, таких як Apache Kafka, для аналізу даних у реальному часі
Інтеграція даних	Здатність інтегрувати дані з різних джерел, включаючи мережевий трафік, журнали подій, соціальні мережі та інші джерела

Виклики впровадження штучного інтелекту в кібербезпеці

Як і будь-яка технологія, штучний інтелект має свої виклики та обмеження. Так само як організації можуть використовувати його для зміцнення профілів кібербезпеки, зловмисники можуть використовувати цю ж технологію для здійснення ворожих атак.
Ось деякі з інших потенційних ризиків і обмежень:
Помилкові спрацьовування або негативи: ШІ може генерувати хибні спрацьовування – ідентифікуючи нешкідливі дії як зловмисні – і хибні негативи, роблячи протилежну помилку. Помилкові спрацьовування та негативи можуть бути складними для навігації, оскільки вони відволікають цінні ресурси на неіснуючі загрози або ігнорують наявні.
Етичні проблеми: Хоча ШІ є революційним у сфері кібербезпеки, він створює етичні та регуляторні проблеми, особливо щодо конфіденційності даних. Наприклад, медичні організації повинні дотримуватися суворих правил HIPAA, щоб забезпечити конфіденційність інформації про пацієнтів.
Дефіцит навичок: ШІ є життєздатним варіантом лише для тих організацій, які мають доступ до кваліфікованих фахівців з кібербезпеки. Оскільки 63% організацій стверджують, що найбільший дефіцит навичок у них спостерігається у сфері ШІ та машинного навчання, подолання цього розриву може бути непростим завданням. Маючи необхідні навички та досвід, організації можуть усвідомити загальну цінність своїх інвестицій у ШІ та машинне навчання.
Якість даних. ШІ настільки ж ефективний, як і дані, на яких ви його тренуєте, і потребує великої кількості високоякісних даних, щоб підвищити точність. Багатьом організаціям важко отримати доступ до достатньої кількості цих високорівневих даних через проблеми конфіденційності та розмежованості.

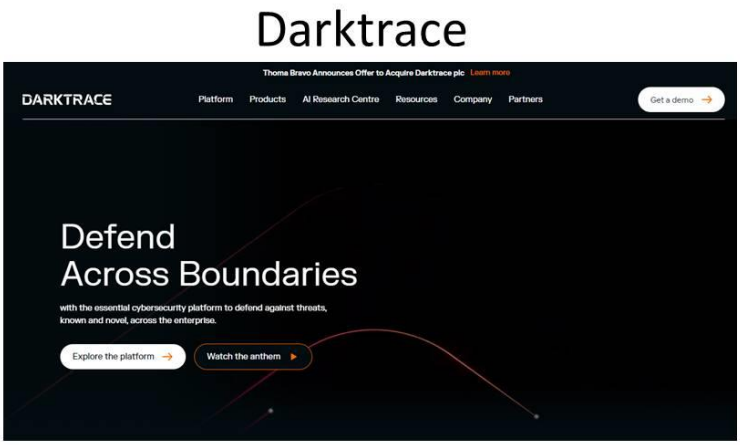
Виклики впровадження штучного інтелекту в кібербезпеці

- Помилкові спрацьовування або негативи
- Етичні проблеми
- Дефіцит навичок
- Якість даних

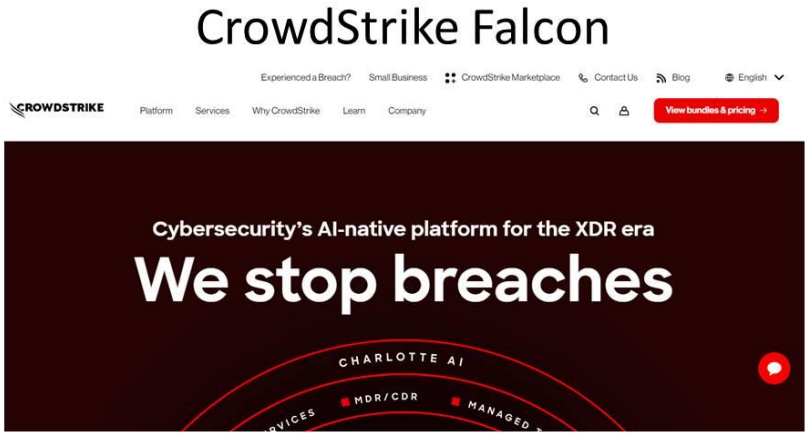
Програмні платформи для кібербезпеки

Darktrace стала піонером у галузі кібербезпеки на основі ШІ, пропонуючи складну платформу, яка використовує алгоритми самонавчання для виявлення кіберзагроз і реагування на них у режимі реального часу. В основі технології Darktrace лежить корпоративна імунна система, яка постійно навчається та адаптується до унікальних цифрових шаблонів у мережі організації.
Цей адаптивний штучний інтелект аналізує дані з різних джерел, включаючи хмарні служби, програми SaaS, системи електронної пошти, пристрої IoT, промислові системи управління та традиційну мережеву інфраструктуру. Встановлюючи базову лінію «нормальної» поведінки для кожного користувача, пристрою та підключення, Darktrace може швидко ідентифікувати аномалії, які можуть вказувати на потенційне порушення безпеки.
Функція платформи Cyber AI Analyst робить ще один крок у боротьбі з загрозами, автоматизуючи процес розслідування та звітування. Ця функція, керована штучним інтелектом, може скоротити час і зусилля, необхідні командам безпеки для сортування та реагування на інциденти, часто завершуючи аналіз, який займає години за лічені хвилини.
Ключові особливості платформи Darktrace:
– ШІ, що самонавчається, постійно адаптується до поведінки мережі, не вимагаючи ручних правил або підписів

– виявлення загроз у реальному часі в поєднанні з можливостями автономного реагування для швидкого пом’якшення загроз;
– інтуїтивно зрозуміла візуалізація загроз, що забезпечує контекстне розуміння масштабів і впливу інцидентів безпеки;
– всебічне покриття в різноманітних цифрових середовищах, від хмарних сервісів до промислових систем керування;
– розслідування інцидентів за допомогою штучного інтелекту та звітування через Cyber AI Analyst, що значно оптимізує операції безпеки.



CrowdStrike Falcon
CrowdStrike Falcon представляє нову парадигму платформ кібербезпеки, пропонуючи хмарне рішення, яке використовує потужність ШІ та машинного навчання для захисту від складних кіберзагроз. Архітектура платформи побудована навколо єдиного полегшеного агента, який об’єднує численні функції безпеки, включаючи антивірус наступного покоління (NGAV), виявлення кінцевих точок і реагування на них (EDR), керований пошук загроз і керування вразливими місцями.
Моделі штучного інтелекту Falcon постійно тренуються на основі трильйонів подій безпеки щодня, що дозволяє платформі адаптуватися до нових загроз і забезпечувати проактивний захист. Цей величезний набір даних у поєднанні з вдосконаленими алгоритмами машинного навчання дозволяє Falcon виявляти та запобігати навіть найскладнішим атакам, включаючи загрози без шкідливого програмного забезпечення та файлів.
Хмарний дизайн платформи пропонує кілька переваг, зокрема безпроблемне розгортання, автоматичні оновлення та уніфіковану консоль керування. Ця архітектура спрощує операції безпеки, зменшує складність і гарантує, що організації завжди мають доступ до найновіших можливостей аналізу загроз і захисту.
Ключові особливості платформи CrowdStrike Falcon:
– виявлення загроз на основі штучного інтелекту та можливості автоматичного реагування для запобігання та пом’якшення атак у реальному часі;
– власна хмарна архітектура забезпечує спрощене розгортання, легку масштабованість і автоматичне оновлення;
– єдиний легкий агент, що об’єднує кілька модулів безпеки для комплексного захисту кінцевої точки;
– розширена поведінкова аналітика та аналіз загроз для виявлення та запобігання складним загрозам нульового дня;
– уніфікована консоль керування та модульна конструкція платформи для оптимізації операцій безпеки та легкої інтеграції.



Platform Vectra AI

Platform Vectra AI позиціонує себе на передньому краї кібербезпеки на основі штучного інтелекту. Це інноваційне рішення використовує вдосконалений штучний інтелект і алгоритми машинного навчання, щоб забезпечити постійний моніторинг у режимі реального часу всієї цифрової екосистеми організації.

Механізм платформи Vectra AI Platform аналізує величезну кількість мережевих метаданих, включаючи моделі трафіку, поведінку користувачів і взаємодію з хмарою. Цей комплексний підхід дозволяє платформі виявляти як відомі загрози, так і тонкі ознаки потенційних атак, які можуть уникнути традиційних заходів безпеки.

Однією з видатних особливостей платформи Vectra AI Platform є її здатність автоматично визначати пріоритет виявлених загроз на основі їх потенційного впливу та серйозності. Ця можливість допомагає командам безпеки зосередити свої зусилля на найбільш критичних інцидентах, значно покращуючи час реагування та загальну ефективність безпеки.

Ключові особливості платформи Vectra AI:

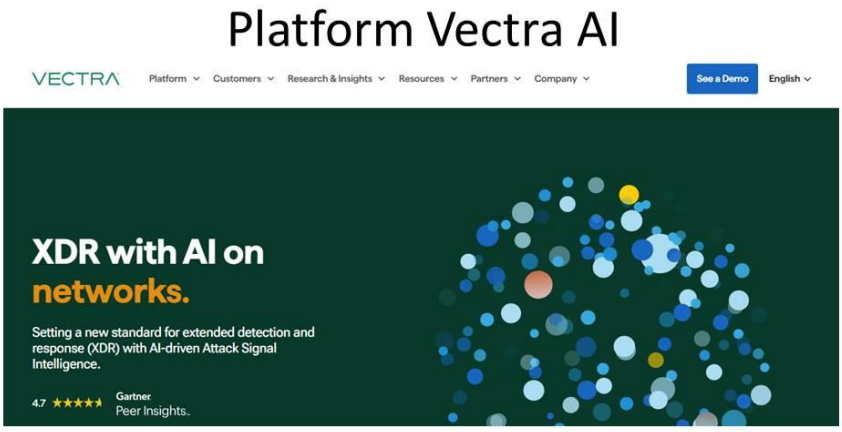
– виявлення загроз на основі ШІ постійно відстежує мережевий трафік, поведінку користувачів і хмарне середовище;

– автоматизоване визначення пріоритетів загроз за допомогою машинного навчання для ранжування виявлених загроз на основі серйозності та потенційного впливу;

– пошук загроз за допомогою штучного інтелекту, активний пошук прихованих загроз у всій цифровій екосистемі;

– повна видимість локальних мереж, робочих навантажень у хмарі, програм SaaS і поведінки користувачів;

– повна інтеграція з існуючими інструментами безпеки для скоординованої та ефективної реакції на загрози.

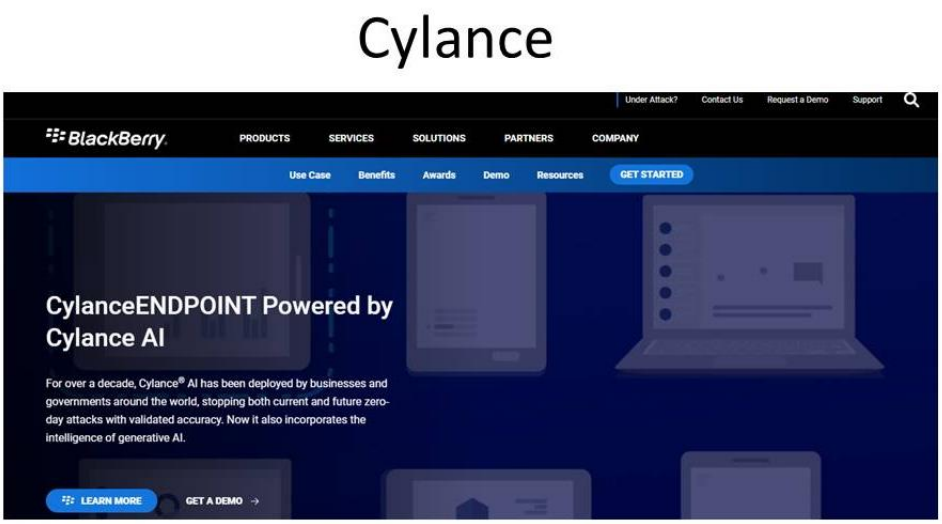


Платформа Internxt

Internxt представляє зміну парадигми хмарного сховища, приділяючи пріоритет конфіденційності користувачів і безпеці даних за допомогою інноваційних технологій, розширених ШІ. Ця іспанська компанія, заснована у 2020 році, швидко завоювала визнання завдяки своєму підходу до шифрування без знань і децентралізованій архітектурі.
В основі послуги Internxt лежить процес шифрування на стороні клієнта. Перш ніж будь-які дані покинуть пристрій користувача, вони шифруються за допомогою передових алгоритмів, гарантуючи, що лише користувач володіє ключами розшифровки. Цей підхід гарантує, що навіть сам Internxt не зможе отримати доступ або переглянути збережені файли, забезпечуючи безпрецедентний рівень конфіденційності.
Децентралізована архітектура платформи, вдосконалена алгоритмами штучного інтелекту, додатково підвищує безпеку, фрагментуючи файли та розподіляючи їх між кількома серверами. Це не тільки покращує стійкість даних, але й робить фактично неможливим неавторизовані сторони відновити повні файли.
Ключові особливості:
– шифрування з нульовим знанням, що забезпечує абсолютну конфіденційність користувача та конфіденційність даних;
– програмне забезпечення з відкритим кодом, незалежно перевірене провідною європейською фірмою безпеки на прозорість і надійність;
– розширена децентралізована архітектура, яка фрагментує та розповсюджує файли для підвищення безпеки та стійкості;
– інтуїтивно зрозумілий користувальницький інтерфейс у веб-додатках, настільних і мобільних додатках для безперешкодного керування файлами;
– щедрий рівень безкоштовного сховища (до 10 ГБ) і конкурентоспроможні ціни на платні плани, що робить безпечне хмарне сховище доступним.

Платформа Cylance
Cylance, яка зараз працює як дочірня компанія BlackBerry Limited, зробила революцію в безпеці кінцевих точок, використовуючи можливості штучного інтелекту та машинного навчання. Основний продукт компанії, CylancePROTECT, використовує унікальний підхід до виявлення та запобігання загрозам.
На відміну від традиційних антивірусних рішень, які покладаються на виявлення на основі сигнатур, CylancePROTECT використовує AI для аналізу ДНК файлу та прогнозування його потенційної зловмисної поведінки. Цей прогностичний підхід дозволяє програмному забезпеченню виявляти та блокувати як відомі, так і невідомі загрози, включно з атаками нульового дня, перш ніж вони можуть запуститися та завдати шкоди.
Моделі ШІ Cylance навчаються на мільйонах як шкідливих, так і безпечних файлів, що дозволяє системі робити дуже точні прогнози щодо наміру файлу. Цей підхід не тільки

забезпечує більш ефективний захист, але й значно зменшує використання системних ресурсів, зазвичай пов’язане з традиційними антивірусними рішеннями.
Ключові особливості:
– виявлення зловмисного програмного забезпечення на основі штучного інтелекту, що проактивно визначає та блокує як відомі, так і невідомі загрози, не покладаючись на оновлення сигнатур;
– легкий агент кінцевої точки, який використовує до 20 разів менше потужності процесора, ніж традиційні антивірусні рішення;
– ефективне запобігання загрозам нульового дня, безфайловим зловмисним програмним забезпеченням і атакам на основі пам’яті за допомогою прогнозного аналізу файлів;
– CylancePROTECT Home Edition розширює захист корпоративного рівня AI на особисті пристрої співробітників без шкоди для конфіденційності;
– можливість постійного запобігання загрозам, яка не вимагає постійних оновлень або активного підключення до Інтернету, спрощуючи керування безпекою.



Висновки

Кількість атак на інформаційні системи зростає щороку високими темпами. При цьому атаки стають все більш витонченими, а збиток від них – усе вищим. До потенційних цілей тепер відносяться як мережева інфраструктура, так і пристрої інтернету речей та розумні домашні пристрої. «Класичні» засоби антивірусного боротьби вже не здатні впоратися з такими епідеміями, і на допомогу приходять рішення на базі штучного інтелекту.
Виглядаючи по-різному стосовно сучасних рішень з кібербезпеки, методи ШІ надійні та більш гнучкі і здатні покращувати системи захисту від все більшої кількості випереджаючих кіберзагроз. Разом з тим, незважаючи на інтенсивні зміни, які ШІ переніс у область кібербезпеки, відповідні системи ще не готові повністю адаптуватися до середовища, а також робити зміни у своєму стані. На сьогоднішній день ШІ ще не став основною панацеєю для безпеки. У той момент, коли людський інтелект має намір атакувати інтелектуальну систему безпеки, ця система зазнає збою. У той же час це не означає, що ми не повинні використовувати методи ШІ для захисту. Навпаки, ми повинні знати його обмеження та використовувати їх належним чином.

РЕКОМЕНДОВАНА ЛІТЕРАТУРА

Основна

1. Aggarwal, Charu C. Neural Networks and Deep Learning: A Textbook. Springer, 2023. 529 p. ISBN 9783031296420.

2. Géron, Aurélien. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. 3rd ed. O'Reilly Media, 2022. 814 p. ISBN 9781098125974.

3. Russell, Stuart, Norvig, Peter. Artificial Intelligence: A Modern Approach. 4th ed. Pearson, 2020. 1136 p. ISBN 9780134610993.

4. Goodfellow, Ian, Bengio, Yoshua, Courville, Aaron. Deep Learning. MIT Press, 2016. 800 p. ISBN 9780262035613.

5. Nielsen, Michael A. Neural Networks and Deep Learning. Determination Press, 2015. 222 p. ISBN 9780999171407.

Допоміжна

6. Bishop, Christopher M. Pattern Recognition and Machine Learning. Springer, 2006. 738 p. ISBN 9780387310732.

7. Chollet, François. Deep Learning with Python. 2nd ed. Manning Publications, 2021. 504 p. ISBN 9781617296864.

8. Joshi, Prateek. Artificial Intelligence with Python. Packt Publishing, 2017. 423 p. ISBN 9781786464392.

9. Шаховська Н. Б., Камінський Р. М., Вовк О. Б. Системи штучного інтелекту. Львів: Видавництво Львівської політехніки, 2018. 392 с. ISBN 9789669412140.

10. Литвин В. В., Пасічник В. В., Яцишин Ю. В. Інтелектуальні системи. Львів: Новий Світ-2000, 2019. 406 с. ISBN 9789664182956.

Інформаційні ресурси

11. TensorFlow. URL: https://www.tensorflow.org/

12. PyTorch. URL: https://pytorch.org/

13. Keras. URL: https://keras.io/

14. Orange Data Mining. URL: https://orangedatamining.com/

15. Weka Machine Learning Project. URL: https://www.cs.waikato.ac.nz/ml/weka/

16. RapidMiner. URL: https://rapidminer.com/

Навчальне видання

Русу Олександр Петрович

ШТУЧНІ НЕЙРОННІ МЕРЕЖІ

Опорний конспект лекцій для здобувачів вищої освіти,
які навчаються за спеціальностями галузі «Інформаційні технології»

Українською мовою