ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО–НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра комп‘ютерної інженерії

Пояснювальна записка

до магістерської роботи

на ступінь вищої освіти магістр

на тему: «**ВДОСКОНАЛЕННЯ СИСТЕМИ ПОШУКУ МЕДИЧНИХ ПРЕПАРАТІВ ЗА КЛІНІЧНИМ ДІАГНОЗОМ НА ОСНОВІ ШТУЧНОЇ НЕЙРОННОЇ МЕРЕЖІ** »

Виконав: студент 6 курсу, групи ПДМ–61

спеціальності 121 Інженерія програмного забезпечення

(шифр і назва спеціальності)

Маринскас В.М

(прізвище та ініціали)

Керівник Трінтіна Н.А

(прізвище та ініціали)

Рецензент_____

(прізвище та ініціали)

Київ – 2022

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти - «Магістр» Спеціальність - 121 «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інженерії програмного
забезпечення
_____О.В. Негоденко
“ _____” _____2022 року

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Маринскас Вадим Миколайович
(прізвище, ім‘я, по батькові)

1. Тема роботи: «Вдосконалення системи пошуку медичних препаратів за клінічним діагнозом на основі штучної нейронної мережі»

Керівник роботи Трінтіна Наталія Альбертівна к. тех. н., доцент
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затвердженні наказом вищого навчального закладу від «12» жовтня 2022 року №122

2. Строк подання студентом роботи 31.12.2022
3. Вихідні дані до роботи: Матеріали переддипломної практики, вдосконалення системи пошуку медичних препаратів за клінічним діагнозом на основі штучної нейронної мережі
4. Зміст розрахунково – пояснювальної записки (перелік запитань, які потрібно розробити):
   4.1 Огляд предметної області
   4.2 Розробка моделей та методів
   4.3 Розробка програмного забезпечення моделей
   4.4 Проведення моделювання та аналіз отриманих результатів
5. Перелік графічного матеріалу:
   5.1. Мета, об’єкт та предмет дослідження.

5.2. Класифікація різних методів прискорення алгоритмів навчання.

5.3. Математична модель багатошарового персептрона.

5.4. Алгоритм навчання запропонованого проміжного пз

5.5. Модуль шаблону користувача.

5.6. Алгоритм Левенберга-Маркварда.

5.7. Результати досліджень

5.8. Висновки.

5.9. Публікації та апробація роботи

6. Дата видачі завдання «14» жовтня 2022 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів роботи	Примітка
1	Отримання завдання на магістерську роботу	14.10.2022	Виконано
2	Огляд предметної області	17.10.2022	Виконано
3	Аналіз методів та засобів навчання нейронних мереж	20.11.2022	Виконано
4	Розробка моделей та методів	22.11.2022	Виконано
5	Розробка програмного забезпечення вдосконалення навчання	23.11.2022	Виконано
6	Модулювання та аналіз результату	26.11.2022	Виконано
7	Написання та оформлення пояснювальної записки	27.11.2022	Виконано
8	Розробка графічних та презентаційних матеріалів	11.12.2022	Виконано
9	Захист магістерської	17.01.2023	Виконано

Студент _____ Маринскас В.М
(підпис) (прізвище та ініціали)

Керівник роботи _____ Трінтіна Н.А
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Текстова частина магістерської роботи 103 сторінки, 35 рисунків, 68 джерел, 4 таблиці.

Ключові слова: класифікація медичних препаратів, медичні препарати, нейронна мережа, лікар, штучний інтелект, багатошаровий персептрон, математика, фармакологія, заходи самоадаптивного навчання, теорія інформації та інтеграція статистики.

Мета роботи: підвищення ефективності пошуку медичних препаратів за допомогою штучної нейронної мережі

Об'єкт дослідження: процес вдосконалення навчання нейронних мереж

Предмет дослідження: методи вдосконалення нейронних мереж

Методи дослідження. Метод самоадаптивного алгоритму та метод Левенберга-Марквардта, обробка та аналіз інформації, алгоритм пошуку.

У роботі проведено аналіз систем пошуку за допомогою штучної нейронної мережі, які використовуються для зручності та об'єктивності пошуку медичних препаратів за клінічним діагнозом та допомоги лікарю правильніше обрати медичний препарат який краще підійде пацієнту.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

SAB - Self-Adaptive Mechanisms

DCNN - Deconvolutional Neural Network

CNN - Inverse Neural Network

ШІ - Штучний Інтелект

RNN - Recurrent Neural Network

MLP - Rumelhart Multilayer perceptron

FFANN - Feed Forward Artificial Neural Network

EBP - Engelhardt Bain Process

SABPA - Self-Adaptive Mechanisms BP

DB - DataBase

LM - алгоритм Левенберга-Маркаврдта

СЕ - Пара Навчання

ШНМ - Штучна Нейронна Мережа

ANN - Automated Neural Networks

ВСТУП

В світі існує багато хвороб та хворих людей, які без допомоги лікарів не можуть існувати. Людський мозок здатний обробляти та запам’ятовувати велику кількість інформацію, але він так влаштований, що якщо інформацію яка вивчила людина не повторюється то вона просто забувається. Тому і потрібна ця система пошуку на основі штучної нейронної мережі. Для того щоб лікарю було простіше та швидше знайти потрібний препарат за клінічним діагнозом. Саме для цього буде досліджуватись вдосконалення системи пошуку за клінічним діагнозом. Вдосконалення буде полягати в наступному. За допомогою математичних та теоретичних обчислень розробити вдосконалення швидкості навчання багатошарової нейронної мережі.

Ціль: вдосконалити навчання штучної нейронної мережі яка підвищує продуктивність та швидкість навчання. Це буде робитись шляхом поповнення ваг і рівнів упередженості мережі. Правило навчання є одним з факторів, який визначає, як швидко або наскільки точні можуть бути розроблені штучні мережі.

*Актуальність теми***:** Люди багато хворіють в когось вроджені хвороби, а в когось набуті. Тому багато хто звертається до лікаря щоб вирішити ту чи іншу проблему пов’язану із здоров’ям. За допомогою цієї системи пошуку лікар буде ефективніше та правильно підбирати препарати для пацієнта. Потрібно буде заповняти деякі важливі поля, а система пошуку в парі з штучною нейронною мережею буде допомагати лікарю виписати найкращий варіант препаратів.

Постановка задачі. Для зручності та об’єктивності вибору препарату за клінічним діагнозом.

Мета роботи: підвищення ефективності пошуку медичних препаратів за допомогою штучної нейронної мережі

Об’єкт дослідження: процес вдосконалення навчання нейронних мереж

Предмет дослідження: методи вдосконалення нейронних мереж

Методи дослідження. Метод самоадаптивного алгоритму та метод Левенберга-Марквардта, обробка та аналіз інформації, алгоритм пошуку.

Наукова новизна даної роботи полягає в наступному:

- Вдосконалення системи пошуку медичних препаратів за клінічним діагнозом.
- Детальний розбір питання діагнозів та препаратів для кращої картини ситуації.
- Розробка структури для швидшого пошуку препаратів.
- Збільшення швидкості навчання методу самоадаптивного зворотнього поширення.

Результати дослідження.

Дослідження розділено на декілька етапів, а саме :

- Дослідження історії виникнення медицини та виникнення препаратів.
- Дослідження системи пошуку та їх аналогів.
- Розробка макета для наглядного прикладу.

Галузь використання. Завдяки цій системі пошуку лікарі в змозі швидко та без ніяких проблем знайти потрібний препарат.

1 ОСНОВИ ФАРМАКОЛОГІЇ

Фармакологія - наука про лікарські засоби, що вивчає взаємодію лікарських засобів з організмом. Включає в себе загальну фармакологію, що вивчає основні закономірності взаємодії ліків з організмом; спеціальну фармакологію, що займається вивченням фармакокінетики та фармакодинаміки окремих лікарських засобів.

Історія фармакології має давній початок. Спочатку люди не мали певних знань про лікарські препарати. Люди застосовували дію речовин природного походження шляхом проб і помилок. На той період історії лікознавство пов’язане з іменами Гіппократ, Клавдій Гален. Та в кінці XVIII століття на початку XIX століття розпочався науковий етап на цьому етапі були різні відкриття у різних науках. Почали створюватись експерементальні лабораторії де вивчались властивості різних речовин. Відкривались університети з кафедрами фармакології.

Медицина розвивається і далі, та досягнуло не аби яких успіхів в своїй сфері, але без ІТ цей процес був би не таким успішним. Тому і розробляється ця система пошуку медичних препаратів за клінічним діагнозом для допомоги медицині розвиватись ще більше та допомогти багатьом людям вирішити свої проблеми з якими вони прийшли до лікаря.

1.1 Фармакодинаміка лікарських засобів

Фармакодинаміка - розділ загальної фармакології, що займається вивченням механізму дії, видів дії, фармакологічних ефектів лікарських речовин.

Механізм дії - спосіб досягнення фармакологічного ефекту шляхом взаємодії лікарських речовин з певними структурами організму (взаємодія з нервовими закінченнями, вплив на певні центри, стимулювання або блокування виділення організмом медіаторів).

Як наслідок певного механізму дії для кожного лікарського препарату і лікарської речовини притаманні певні фармакологічні ефекти, тобто зміни в організмі під дією лікарського засобу.

Види дій лікарських засобів:

- Місцева - дія препарату, що проявляється на місці застосування (введення)
- Резорптивна (резорбція-всмоктування) - дія, що розвивається після всмоктування речовини та потрапляння її в кров.
- Рефлекторна - дія, що проявляється в наслідок взаємодії лікарської речовини з рецепторами.
- Пряма - контакт лікарської речовини з тканинами.
- Вибіркова- дія препарату на певні типи рецепторів.
- Зворотна - характерна для всіх препаратів які використовують не міцний зв’язок з певною структурою організму на певний період часу (тривалість дії препарату).
- Незворотна - міцна взаємодія речовини з рецептором, що зможе привести до токсичних проявів дії, тому для лікарських препаратів

Внаслідок тривалого застосування лікарських препаратів може відбутись посилення або послаблення їх ефекту.

Посилення ефекту препаратів пов'язаний з їх здатністю до комуляції (накопичення).

Толерантність (звикання) - зниження ефективності препарату при повторних застосування. Звикання може бути пов’язане із зменшенням швидкості всмоктування речовини, збільшенням швидкості метаболізму та посиленим виділенням з організму.

Різновидом звикання є тахіфілаксія - толерантність яка розвивається дуже швидко. В таких випадках збільшення дози чутливість препарату не відновить.

1.2 Спеціальна фармакологія

Спеціальна фармакологія поділяється на фармакокінетику та фармакодинаміку лікарських засобів. До фармакокінетики ми відносимо введення, всмоктування, розподіл, перетворення, виведення. До фармакодинаміки - механізм дії, фармакологічний ефект, показання до застосування, побічні явища, протипоказання.

Розглянемо деякі групи лікарських засобів які часто використовуються в медичній практиці.

1.2.1 Серцево-судинні засоби

Кардіотонічні засоби - застосовуються для лікування захворювання, що називається серцева недостатність.

Основною групою кардіотонічних засобів, що використовуються для лікування серцевої недостатності є серцеві глікозиди - препарати рослинного походження які є складноорганічними сполуками типу ефірів, що в процесі гідролізу розщеплюються на нецукристу частину (аглікон) який є носієм кардіотонічних властивостей та цукристу частину (глікон), що відповідає за фармакокінетику.

Серцеві глікозиди класифікують за видом рослин з яких вони одержані:

- Препарати наперстянки: дигоксин, дигітоксин, кордегіт.
- Препарати строфанту:строфантин К.
- Препарати конвалії: корглікон.
- Препарати горицвіту весняного: адонізид
- Препарати морської цибулі: кардіовален
- Комбіновані засоби кліфт

Механізм дії пов'язаний зі здатністю серцевих глікозидів утворювати комплексні сполуки з іонами кальцію та транспортувати їх у цитоплазму кардіоміцетів.

Фармакологіний ефект- каріотонічний:

- Позитивний інотропний
- Негативний хронотропний
- Негативний дромотропний
- Позитивний батнотропний

Протиаритмічні засоби - група препаратів, що використовується для зняття і профілактики серцевих артемій.

Аритмії поділяються на тахіаритмії і брадиаритмії.

Види тахіаритмії

- Екстрацистолія
- Пароксизмальна тахікардія
- Миготлива аритмія

Брадиаритмії виникають при порушенні проведення нервового імпульсу по провідній системі серця і проявляються у вигляді серцевих блокад різного ступеня, наприклад блокада проведення імпульсу між передсердями і шлуночками називається атріовентикулярна блокада.

Засоби, що відстороняють тахіаритмію:

- Мембаностабілізуючі засоби
- β - адреноблокатори
- Блокатори кальцієвих каналів
- Препарати що продовжують реполярізацію
- Препарати калію

Лікарські засоби, що використовуються при брадиаритміях:

- β - адреноміметики
- α,β - адреноміметики
- М – холіноблокатори

Антиангінальні засоби - лікарські засоби, що застосовується при недостатності коронарного кровообігу.

Патологічні стани пов’язані з недостатністю вінцевого кровообігу об’єднуються терміном ішимічна хвороба серця. Гострими формами

ішемічної хвороби серця є стенокардія та інфаркт міокарда.

Лікарські засоби, що використовуються при ішемічній хворобі серця:

- Нітровазоделятатори
- Блокатори кальцієвих каналів
- Засоби що продовжують реполяризацію
- β - адреноблокатори
- Засоби які поліпшують метаболізм міокарда
- Антиагреганти

Гіпотензивні засоби

В залежності від етиології гіпертензивна хвороба поділяється на первинну (есенціальну) та вторинну.

Класифікація лікарських засобів:

- Засоли нейротропної дії
- Периферичні судинно - розширюючі засоби
- Комбіновані гіпотензивні засоби

Антигіперліпопротеїнемічні засоби - група лікарських засобів, що застосовуються для профілактики та лікування атеросклерозу то його ускладнень.

Клінікобіохімічним процесом атеросклерозу є гіперліпопротеїнемія – підвищення вмісту певних класів ліпопротеїнів, тригліциридів та холістерину в плазмі крові.

Класифікація лікарських засобів:

- Лікарські засоби, що знижують синтез холестерину в печінці
- Лікарські засоби, що зв’язують в кишківнику жовчні кислоти та холестерин і виводять їх з організму
- Фібрати
- Інші засоби

1.2.2 Болезаспокійливі засоби

За походженням біль поділяють на:

- Вісцеральний – з боку внутрішніх органів
- Соматичний – в кістках, суглобах зв’язках

В цілому система, що проводить і сприймає больові відчуття називається ноцеципти вною. А система яка протидіє – антиноцеципти вною. Основним компонентом є опіоїдна система яка складається з опіоїдних рецепторів та складається з опіоїдних речовин.

Болезаспокійливі засоби поділяються на:

- Наркотичні анальгетики (опіоїди)
- Ненаркотичні

1.2.3 Психотропні лікарські засоби

Психотропними називають препарати під впливом яких змінюється психічний і емоційний стан людини. Серед психотропних засобів є засоби з пригнічуючим впливом на ЦНС і зі стимулюючим.

Групи, що чинять пригнічуючий вплив:

- Нейролептики
- Транквілізатори
- Седативні засоби

Стимулюючі:

- Антидепресанти
- Психостимулятори
- Загальнотонізуючі засоби
- Ноотропні
- Нормотиміки
- Аналептики

1.2.4 Лікарські засоби, що впливають на функцію органів дихання

Лікарські засоби, що впливають на функції органів дихання поділяються на:

- Стимулятори дихання
- Протикашльові
- Відхаркувальні
- Ті, що застосовуються при бронхіальній астмі
- Ті, що застосовуються при набряку легень

1.2.5 Антисептична та дезінфікуючі засоби; хіміотерапевтичні засоби

Антисептичні – це засоби, що використовуються для впливу на мікроорганізми, що знаходяться на поверхні шкіри, слизових оболонках, поранених поверхнях або порожнинах.

Дезінфікуючі – це засоби, що використовують для впливу на мікроорганізми оточуючого середовища, повітря, інструментів.

Протимікробні засоби можуть виявляти два види дії відносно мікроорганізмів:

- Бактеріостатична – здатність препаратів тимчасово припиняти ріст і розмноження мікроорганізмів
- Бактерицидна – здатність препарату викликати загибель мікроорганізмів

Часто один і той самий препарат в залежності від дози може проявляти обидва види дії.

Класифікація:

- Неорганічні сполуки
- Органічні сполуки
- Хіміотерапевтичні засоби – називаються лікарські засоби, що виявляють вибіркову протимікробну дію. Застосовують

хіміотерапевтичні засоби з лікувальною і профілактичною метою для припинення або попередження клінічних ознак інфекції, а також для знищення збудників інфекції в організмі здорових людей.

Групи:

- Антибіотики та фторхінологи
- Засоби вибіркової дії
- Синтетичні засоби

Для прикладу роботи системи пошуку медичних препаратів було прийнято рішення розробити модель системи пошуку медичних зображено на рис 1.1 .Де показано пошук від симптоматики та діагнозу до медичного препарату який найкраще підійде для вирішення проблем які були поставлені перед представником медичної установи. Нижче ви можете бачити цю модель про яку розповідалось вище.

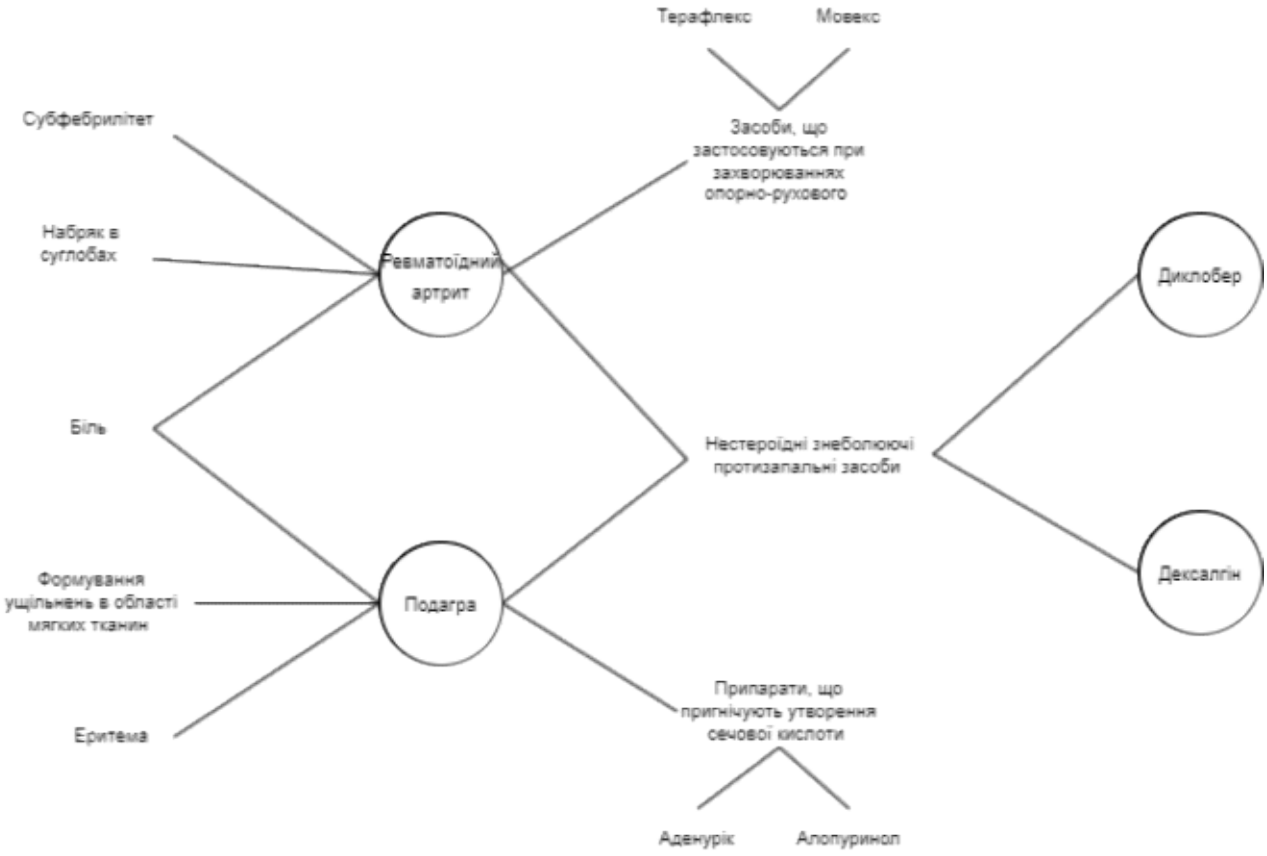


Рисунок 1.1 – Система пошуку медичних препаратів за клінічним діагнозом.

1.3 Лікарська справа

Їх можна назвати просто лікарями. Але більшість лікарів мають додатковий досвід у тому чи іншому виді медицини. Насправді існує кілька сотень медичних спеціальностей і субспеціальностей. Ось найпоширеніші типи лікарів, яких ви, ймовірно, зустрінете в повсякденному житті.

- Сімейні лікарі. Вони піклуються про всю сім'ю, включаючи дітей, дорослих і людей похилого віку. Вони проводять регулярні огляди та скринінгові тести, роблять вам щеплення від грипу та імунізацію, а також лікують діабет та інші поточні захворювання.
- Анестезіологи. Ці лікарі дають вам ліки, щоб заглушити ваш біль або ввести вас під час операції, пологів чи інших процедур. Вони контролюють ваші життєво-важливі функції, поки ви перебуваєте під наркозом.
- Кардіологи. Вони знають серце і судини. Ви можете бачити їх у разі серцевої недостатності, серцевого нападу, високого кров’яного тиску або нерегулярного серцебиття.
- Реаніматолог. Вони доглядають за важкохворими або пораненими людьми, часто очолюють відділення інтенсивної терапії лікарень. Ви можете побачити їх, якщо ваше серце чи інші органи не працюють або якщо ви потрапили в аварію.
- Дерматолог. Маєте проблеми зі шкірою, волоссям, нігтями? У вас є родимки, плями, прищі або шкірна алергія? Може допомогти дерматолог.
- Спеціаліст екстреної медичної допомоги. Ці лікарі приймають рішення щодо життя та смерті хворих і поранених, часто у відділенні невідкладної допомоги. Їхнє завдання — рятувати життя та уникати або зменшувати ймовірність інвалідності.
- Гематолог. Вони є спеціалістами із захворювань крові, селезінки та

лімфатичних вузлів, включаючи серповидно-клітинну анемію.

- Інфекціоніст. Діагностика та лікування інфекцій у будь-якому місці тіла, включаючи лихоманку, хворобу Лайма, пневмонію, туберкульоз, ВІЛ та СНІД. Деякі з них спеціалізуються на профілактичній медицині або медицині подорожей.
- Медичний генетик. Ми займаємося діагностикою та лікуванням генетичних захворювань, що передаються від батьків до дітей. Ці лікарі також можуть запропонувати генетичне консультування та скринінгові тести.
- Невролог. Вони є спеціалістами з нервової системи, включаючи мозок, спинний мозок і нерви. Лікує інсульт, пухлини головного мозку, пухлини спинного мозку, епілепсію, хвороби Паркінсона та Альцгеймера.
- Онколог. Ці терапевти є спеціалістами з раку. Вони проводять хіміотерапію та часто співпрацюють з радіаційними онкологами та хірургами, щоб доглядати за хворими на рак.
- Лікарі-офтальмологи. Вони можуть прописати окуляри або контактні лінзи, а також діагностувати та лікувати такі захворювання, як глаукома. На відміну від оптометристів, вони є лікарями, які можуть лікувати будь-які захворювання очей, а також оперувати очі.
- Отоларингологи. Вони лікують захворювання вух, носа, горла, носових пазух, голови, шиї та дихальної системи. Вони також можуть зробити реконструктивні та пластичні операції на голові та шиї.

Всі ці лікарі які перераховувались вище дуже важливі для нашого суспільства. Ця вдосконалена система пошуку підходить кожному з перелічувальних лікарів. Які за допомогою цього вдосконалення будуть виконувати свою роботу в рази якісніше та швидше.

2 ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

2.1 Характеристика структурно-функціональних особливостей

Медицина є невід’ємною частиною нашого життя, від якої залежить наш стан здоров’я. Навіть при звичайній простуді без медицини нікуди. Тому що, якщо не лікувати, то може перерости, в щось більш серйозне, ніж звичайна простуда.

На сьогоднішній день медицина врятувала не один мільйон людей від смерті та тяжких наслідків неприймання лікарських засобів.

Негативні наслідки приймання ліків також. Інколи буває, що в ліків більше побічних ефектів, ніж те, що вони лікують. Але це й не дивно, тому що медичні препарати це не так просто як здається на перший погляд - це дозоване змішування різних компонентів між собою. І в більшості випадків компоненти які змішуються між собою мають побічну дію.

Медицина - це сфера здоров'я та лікування. До нього входять медсестри, лікарі, різні спеціалісти. Він охоплює діагностику, лікування та профілактику захворювань, медичні дослідження та багато інших аспектів здоров’я.

Медицина спрямована на зміцнення та підтримку здоров’я та добробуту.

Традиційну сучасну медицину іноді називають алопатичною. Це передбачає використання ліків або хірургічне втручання, часто супроводжуване консультуванням і заходами щодо способу життя.

Існує велика кількість талановитих лікарів в різних галузях такі як, офтальмологи, лори, онколкологи, терапевти тощо. Вони мають запам ‘ятовувати та обробляти велику кількість інформації. В хороших лікарів завжди велика клієнтська база і кожен приходить зі своєю особливою хворобою, яка не схожа на попередню проблему пацієнта. Тому і було прийнято рішення проаналізувати системи пошуку які є на даний момент існують в метеріальній базі та обрати саме ту, яка підходить для вирішення завдання яке було поставлене перед вдосконаленням цієї системи пошуку.

Також доцільно зазначити, що запропонований підхід на основі штучної нейронної мережі відмінний, від раніше розроблених продуктів в цій області, оскільки може не тільки робити пошук медичних препаратів, а ще й робити пошук медичних препаратів за клінічним діагнозом. Система пошуку буде сама підбирати медичні препарати які будуть розбиті по класифікаціях та групах та низці інших факторів. Отже, лікареві вже не доведеться згадувати тей чи інший лікарський засіб його класифікацію чи побічні дії препарату і витрачати час на пошук цього перелічення , що зазначена вище. Система зробить це замість нього.

Очікується, що ця система пошуку буде мати великий попит, адже як ми всі знаємо, що для лікарів та пацієнтів важлива кожна хвилина витраченого часу. Ця хвилина може врятувати комусь життя.

2.2 Аналіз моделей, методів та засобів

Оскільки вдосконалювання системи пошуку буде з використанням штучних нейронних мереж, тому буде доцільно розібратись детальніше. Та проаналізувати деякі з цих видів та методів.

В інформаційних технологіях (ІТ) штучна нейронна мережа (ШНМ) — це система апаратного та/або програмного забезпечення, створена за принципом роботи нейронів у мозку людини. ШНМ, які також називають просто нейронними мережами, є різновидом технологій глибокого навчання, які також підпадають під парасольку штучного інтелекту, або ШІ.

Для того щоб зрозуміти, як працює нейронна мережа потрібно буде розібратись детальніше.

Нейронну мережу потрібно навчати та адаптовувати під свої потреби нижче розповідатиметься, як саме влаштоване навчання штучної нейронної мережі. Для початку я поясню тип штучного нейрона під назвою а персептрон. Персептрони були розроблені в 1950-1960-х роках вченим Франком Розенблата, натхненний попередніми роботами Воррена МакКалоха та Волтера Пітса. Сьогодні це більше зазвичай використовують інші моделі штучних нейронів – у цій книзі та багатьох

сучасних роботах у нейронних мережах використовується основна модель нейрона, яка називається сигмовидним нейроном. Ми отримаємо незабаром до сигмовидних нейронів. Але щоб зрозуміти, чому нейрони сигмовидної кишки визначаються так вони є, варто витратити час, щоб спочатку зрозуміти персептрони.

Отже, як працюють персептрони? Персептрон приймає кілька двійкових входів, x1, x2 , . . ., і створює один двійковий вихід:

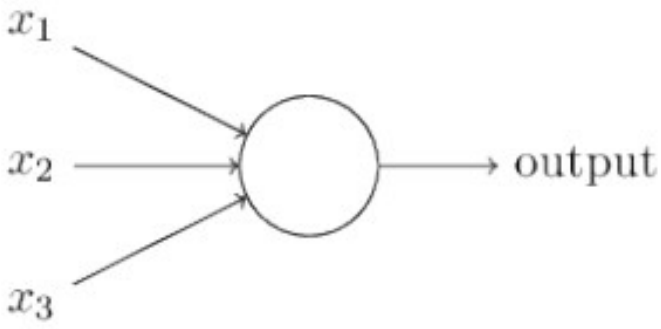


Рисунок 2.1 – Приклад двійкового виходу.

У наведеному прикладі персептрон має три входи x_1, x_2, x_3. Загалом, він може мати більше або менше входів. Розенблат запропонував просте правило для обчислення результату. Він вводить ваги w_1,w_2,..., дійсні числа, що представляють важливість відповіді введення-виведення. Те, чи дорівнює вихід нейрона 0 чи 1, визначається тим, чи є зважена сума $P_j w_j x_j$ меншою або більшою за певний поріг. Як і ваги, пороги є реальними числами і є параметром нейрона. У більш точних алгебраїчних термінах:

$$вихід = \begin{cases} 0, якщо \sum_j w_j x_j \leq поріг \\ 1, якщо \sum_j w_j x_j \leq поріг \end{cases} \quad (2.1)$$

Це все, що стосується роботи персептрона.

Це основна математична модель. Ось як ви можете думати про персептрон - це пристрій, який приймає рішення шляхом зважування доказів. Наведу приклад.

Його не дуже реалістичний приклад, але його легко зрозуміти, і незабаром ми перейдемо до більш реалістичного приклади. Припустімо, що наближаються вихідні, і ви чули, що вони будуть фестиваль сиру у вашому місті. Ви любите сир і намагаєтеся вирішити, йти чи ні фестиваль. Ви можете прийняти рішення, зваживши три фактори:

– Чи хороша погода?
– Ваш хлопець чи дівчина хоче вас супроводжувати?
– Чи фестиваль знаходиться поблизу громадського транспорту? (У вас немає автомобіля).

Ми можемо представити ці три фактори відповідними двійковими змінними x1, x2 і x3. Для наприклад, у нас буде x1 = 1, якщо погода хороша, і x1 = 0, якщо погода погана. Так само x2 = 1, якщо ваш хлопець або дівчина хоче піти, і x2 = 0, якщо ні. І так само знову для x3 і громадський транспорт.

Очевидно, що персептрон не є повною моделлю прийняття людських рішень! Але що приклад показує, як персептрон може зважувати різні види доказів по порядку приймати рішення. І це повинно здаватися правдоподібним, що складна мережа персептронів може приймати досить тонкі рішення:

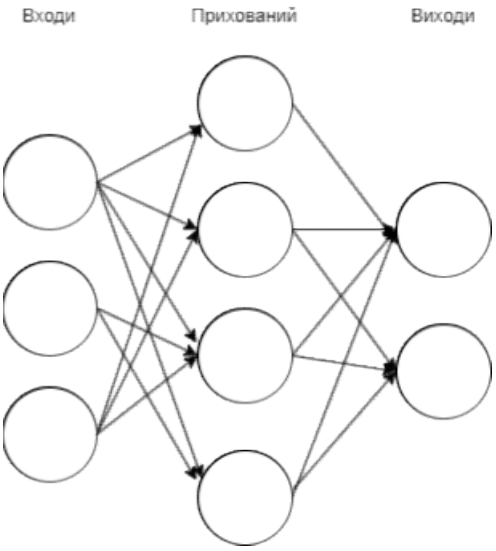


Рисунок 2.2 – Складна мережа персептронів

У цій мережі перший стовпець персептронів – те, що ми називатимемо першим шаром персептронів приймає три дуже прості рішення, зважуючи вхідні

докази. Що щодо персептронів у другому шарі? Кожен із цих персептронів приймає рішення шляхом зважування до результатів першого рівня прийняття рішень. Таким чином персептрон у другому рівень може приймати рішення на більш складному та більш абстрактному рівні, ніж персептрони перший шар. І навіть більш складні рішення може приймати персептрон в третьому шар. Таким чином, багатошарова мережа персептронів може приймати складні рішення виготовлення

Однак ситуація є кращою, ніж передбачає ця точка зору. Виявляється, ми можемо придумати алгоритми навчання, які можуть автоматично налаштовувати ваги та зміщення мережі штучних нейронів. Ця настройка відбувається у відповідь на зовнішні подразники, без прямого втручання програміста. Ці алгоритми навчання дозволяють нам використовувати штучні нейрони у спосіб, який радикально відрізняється від звичайних логічних воріт. Замість явного укладання із схеми NAND та інших вентилів, наші нейронні мережі можуть просто навчитися вирішувати проблеми, іноді проблеми, коли було б надзвичайно важко безпосередньо спроектувати звичайний.

Як правило, ANN спочатку навчається або подається великий обсяг даних. Навчання складається з надання вхідних даних і повідомлення мережі, яким має бути результат. Наприклад, щоб створити мережу, яка ідентифікує обличчя акторів, початковим навчанням може бути серія зображень, включаючи акторів, звичайних людей, маски, статуї та обличчя тварин. Кожне введення супроводжується відповідним ідентифікатором, наприклад іменами акторів або інформацією «не актор» або «не людина». Надання відповідей дозволяє моделі налаштувати свої внутрішні ваги, щоб дізнатися, як краще виконувати свою роботу.

У визначенні правил і прийнятті рішень — тобто рішення кожного вузла про те, що відправляти на наступний рівень на основі вхідних даних із попереднього рівня — нейронні мережі використовують кілька принципів. До них відносяться градієнтне навчання, нечітка логіка, генетичні алгоритми та методи

Байєса. Їм можуть бути дані деякі основні правила щодо зв’язків між об’єктами в даних, що моделюються.

Наприклад, системі розпізнавання обличчя можна вказати: «Брови знаходяться над очима» або «Вуса розташовані під носом. Вуса розташовані над та/або біля рота». Правила попереднього завантаження можуть пришвидшити навчання та швидше зробити модель потужнішою. Але це також створює припущення про природу проблеми, які можуть виявитися або нерелевантними та некорисними, або неправильними та контрпродуктивними, що робить дуже важливим рішення про те, які правила, якщо такі є, включити.

Крім того, припущення, які люди роблять під час навчання алгоритмів, змушують нейронні мережі посилювати культурні упередження. Упереджені набори даних є постійною проблемою в системах навчання, які знаходять відповіді самостійно, розпізнаючи закономірності в даних. Якщо дані, що передаються в алгоритм, не є нейтральними – і майже немає даних – машина поширює зміщення.

Отже, потрібно розібратись які ж саме типи штучних нейронних мереж існують на сьогодні.

Конкретні типи штучних нейронних мереж включають:

Нейронна мережа прямого зв’язку — це штучна нейронна мережа, в якій зв’язки між вузлами не утворюють петель. Протилежністю нейронних мереж прямого зв’язку є рекурентні нейронні мережі, де шляхи є циклічними. Моделі прямого зв’язку є найпростішою формою нейронної мережі, оскільки інформація обробляється лише в одному напрямку. Хоча дані можуть переміщатися через кілька прихованих вузлів, вони завжди переміщуються в один бік і ніколи не повертаються назад.

Нейронні мережі прямого зв’язку: один із найпростіших варіантів нейронних мереж. Вони передають інформацію в одному напрямку через різні вхідні вузли, поки вона не потрапить до вихідного вузла. Мережа може мати або не мати прихованих шарів вузлів, що робить їх функціонування більш зрозумілим. Він готовий до обробки великої кількості шуму. Цей тип обчислювальної моделі

ШНМ використовується в таких технологіях, як розпізнавання обличчя та комп’ютерний зір.

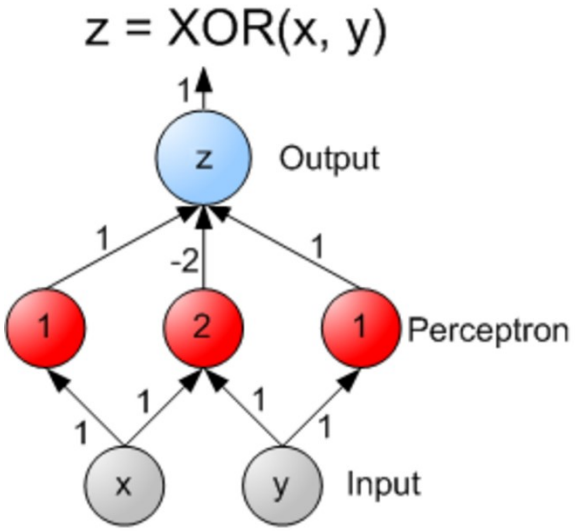


Рисунок 2.3 – Нейронна мережа прямого зв’язку

Нейронну мережу прямого зв’язку зазвичай розглядають у своїй найпростішій формі як одношаровий персептрон. У цій моделі серія вхідних даних входить до рівня та множиться на ваги. Потім кожне значення додається разом, щоб отримати суму зважених вхідних значень. Якщо сума значень перевищує певне порогове значення, яке зазвичай дорівнює нулю, отримане значення часто дорівнює 1, тоді як якщо сума опускається нижче порогового значення, вихідне значення дорівнює -1. Одношаровий персептрон є важливою моделлю прямої нейронної мережі та часто використовується в задачах класифікації. Крім того, одношарові персептрони можуть включати аспекти машинного навчання. Використовуючи властивість, відому як дельта-правило, нейронна мережа може порівнювати виходи своїх вузлів із запланованими значеннями, таким чином дозволяючи мережі коригувати свої ваги шляхом навчання, щоб виробляти більш точні вихідні значення. Цей процес навчання і навчання створює форму градієнтного спуску. У багатошарових персептронах процес оновлення ваг майже аналогічний, однак цей процес визначається більш

конкретно як зворотне поширення. У таких випадках кожен прихований шар у мережі коригується відповідно до вихідних значень, вироблених кінцевим шаром.

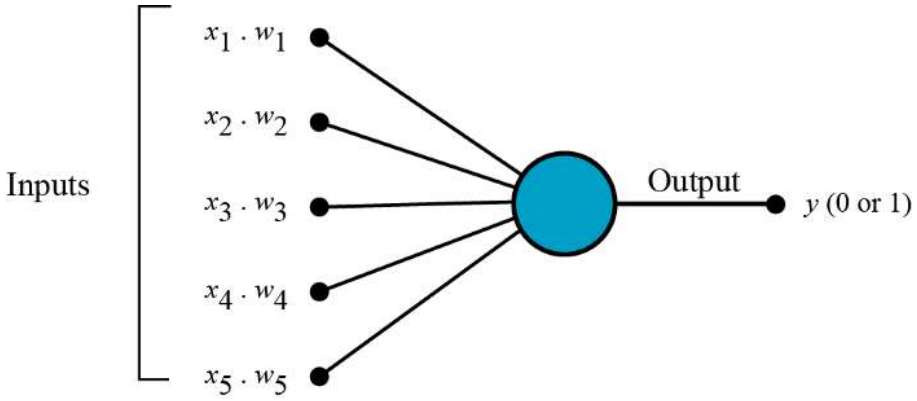


Рисунок 2.4 – Перемноження вхідних даних на ваги для 5 вхідних даних.

Хоча нейронні мережі прямої подачі є досить простими, їх спрощену архітектуру можна використовувати як перевагу в окремих програмах машинного навчання. Наприклад, можна створити ряд нейронних мереж прямого зв'язку з наміром запускати їх незалежно одна від одної, але з м'яким посередником для модерації. Як і людський мозок, цей процес покладається на багато окремих нейронів, щоб справлятися з більшими завданнями. Оскільки окремі мережі виконують свої завдання незалежно, результати можна об'єднати в кінці, щоб отримати синтезований і згуртований результат.

Архітектури варіюються від повністю взаємопов'язаних (рис.2.3) до частково з'єднані мережі (див рис. 2.4), включаючи багаторівневі мережі прямого зв'язку з різні вхідні та вихідні рівні. Повністю підключені мережі не мають відмінностей

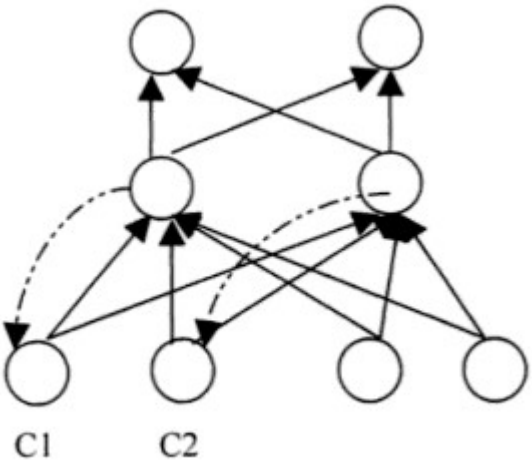


Рисунок 2.5 – Приклад повнозв’язної рекурентної мережі

Вхідні рівні вузлів, і кожен вузол має вхідні дані від усіх інших вузлів. Зворотній зв'язок до самого вузла можливо.

Описано мережеві реалізації фільтрації та контролю. Нарешті, досліджено застосування рекурентних нейронних мереж до хаотичних систем. Нотатки були вивчені за допомогою рекурентних нейронних мереж.

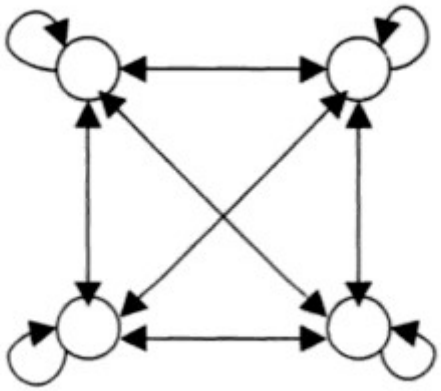


Рисунок. 2.6 – Приклад повнозв ‘язної рекурентної нейронної мережі

Для додавання зворотнього зв’язку до упередження можна використати два основних способи багатошарові нейронні мережі. Ввівши зворотній зв'язок з прихованим шаром до контекстної частини вхідного шару. Такий підхід приносить більше грошей привернути увагу на послідовність введення значень.

Рекурентні нейронні мережі використовують зворотний зв’язок від вихідного рівня до контекстних вузлів вхідний рівень і приділити більше уваги послідовності вихідних значень. Більш ефективні та дієві рекурентні нейронні мережі та їх приклади інших вузлів забезпечують послідовний контекст і отримують зворотний зв'язок від інших вузлів. Ваги з контекстних одиниць (C1 і C2) обробляються так само, як для блоки введення, наприклад, за допомогою зворотного поширення. Одиниці контексту отримують зворотний зв'язок із затримкою часу від, як у випадку (див рис. 2.4), блоків другого рівня.

Навчальні дані складаються з вхідних даних і бажаних наступних вихідних даних. Сітка може навчитися передбачати наступну літеру в рядку символів і перевіряти а рядок символів.

Рекурентні нейронні мережі: більш складні. Вони зберігають вихідні дані вузлів обробки та повертають результати в модель. Кажуть, що таким чином модель навчиться передбачати результат шарів. Кожен вузол у моделі RNN діє як одиниця зберігання, яка продовжує обчислювати та виконувати операції. Нейронна мережа починає з того самого фронту розповсюдження, що й мережа прямого зв’язку, але потім продовжує запам’ятовувати всю оброблену інформацію для подальшого повторного використання. Якщо мережа прогнозує неправильно, система навчається самостійно та продовжує робити правильні прогнози під час зворотного поширення. Цей тип ANN часто використовується для перетворення тексту в мову. Подібно до нейронних мереж прямого зв’язку та зворотних нейронних мереж (CNN), рекурентні нейронні мережі використовують навчальні дані для навчання. Вони відрізняються своєю «пам’яттю», оскільки вони беруть інформацію з попередніх вхідних даних, щоб впливати на поточні вхідні та вихідні дані. Хоча традиційні глибокі нейронні мережі припускають, що входи та виходи незалежні один від одного, вихід рекурентних нейронних мереж залежить від попередніх елементів у послідовності. Хоча майбутні події також були б корисними для визначення результату даної послідовності, односпрямовані рекурентні нейронні мережі не можуть врахувати ці події у своїх прогнозах.

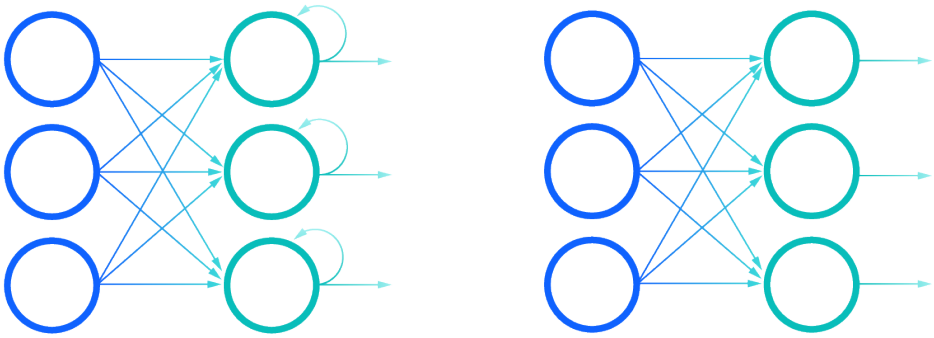


Рисунок 2.7 – Порівняння рекурентних нейронних мереж (ліворуч) і нейронних мереж прямого зв’язку (праворуч)

Деконволюційна нейронна мережа перетворює представлення латентного простору на високовимірні дані, подібні до навчального набору, застосовуючи послідовні операції деконволюції на кількох рівнях [NHH15]. Прихований простір містить приховані змінні низької розмірності, які забезпечують стисле («концептуальне») представлення можливих результатів (наприклад, зображення). Таким чином, латентна змінна може відповідати "стілець", а пов'язаним результатом є зображення стільця, "генероване" DCNN (див. рис. 1.3). На малюнку 2.1 показано 5-рівневу DCNN, розроблену в [RMC15] що складається з 4 деконволюційних шарів. Перший шар повністю пов'язаний і перетворює вхідний розмір 1x100 на вихідний розмір 1024x4x4; шари з 2 по 5 є шари деконволюції, які проектують карти ознак низької розмірності у відповідні високорозмірні через послідовні шари.

Деконволюційні нейронні мережі: використовують зворотний процес моделі CNN. Вони спрямовані на пошук втрачених функцій або сигналів, які спочатку вважалися неважливими для завдання системи CNN. Цю модель мережі можна використовувати для синтезу та аналізу зображень.

Модульні нейронні мережі: містять кілька нейронних мереж, що працюють окремо одна від одної. Мережі не спілкуються та не втручаються в діяльність одна одної під час процесу обчислень. Отже, складні або великі обчислювальні

процеси можуть виконуватися більш ефективно.

Згорткові нейронні мережі відрізняються від інших нейронних мереж своєю чудовою продуктивністю з зображенням, мовленням або аудіосигналом. Вони мають три основні типи шарів, а саме:

- Згортковий шар
- Шар об'єднання
- Повністю підключений (FC) рівень

Згортковий шар є першим шаром згорткової мережі. У той час як за згортковими шарами можуть слідувати додаткові згорткові шари або шари об’єднання, повністю зв’язаний рівень є останнім. З кожним шаром CNN зростає у своїй складності, ідентифікуючи більші частини зображення. Попередні шари зосереджені на простих функціях, таких як кольори та краї. Коли дані зображення просуваються між шарами CNN, вони починають розпізнавати більші елементи або форми об’єкта, поки нарешті не ідентифікують запланований об’єкт.

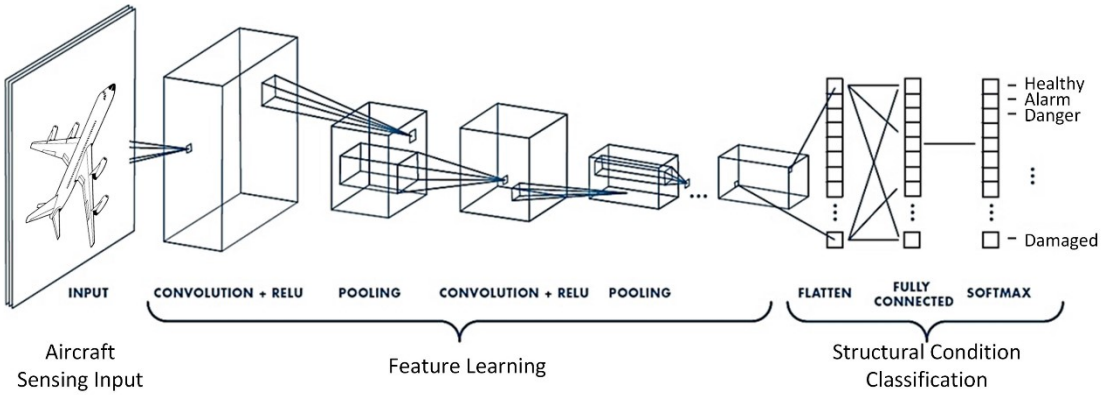


Рисунок 2.8 – Діаграма згорткової нейронної мережі

Згорткові нейронні мережі забезпечують розпізнавання зображень і задачі комп’ютерного зору. Комп’ютерний зір — це сфера штучного інтелекту (ШІ), яка дозволяє комп’ютерам і системам отримувати значущу інформацію з цифрових зображень, відео та інших візуальних вхідних даних, і на основі цих вхідних даних може вживати заходів. Ця здатність надавати рекомендації відрізняє його від завдань розпізнавання зображень. Деякі поширені застосування цього комп’ютерного зору сьогодні можна побачити в:

- Маркетинг: платформи соціальних медіа пропонують пропозиції щодо того, хто може бути на фотографії, опублікованій у профілі, що полегшує позначення друзів у фотоальбомах.
- Охорона здоров'я: комп'ютерний зір було включено в радіологічні технології, що дозволяє лікарям краще ідентифікувати ракові пухлини в здоровій анатомії.
- Роздрібна торгівля: візуальний пошук включено в деякі платформи електронної комерції, що дозволяє брендам рекомендувати речі, які доповнюватимуть наявний гардероб.
- Автомобільна промисловість. Хоча ера безпілотних автомобілів ще не настала, базова технологія почала проникати в автомобілі, покращуючи безпеку водія та пасажирів за допомогою таких функцій, як визначення смуги руху.

Згорткові нейронні мережі: одна з найпопулярніших моделей, що використовуються сьогодні. Ця обчислювальна модель нейронної мережі використовує різновид багатошарових персептронів і містить один або більше згорткових шарів, які можуть бути повністю з'єднані або об'єднані. Ці згорткові шари створюють карти функцій, які записують область зображення, яке зрештою розбивається на прямокутники та надсилається для нелінійного моделювання.

З описаного вище можна зробити висновок, що нейронні мережі є багатофункціональним та відрізняється суттєво одна від одної. Це досягається за допомогою використання моделей для окремих задач. Так, в сучасному світі нейронні мережі мають велику кількість архітектур, зокрема виділяють наступні:

- Канал розподілу — це ланцюжок підприємств або посередників, через які проходить товар або послуга, доки не досягне кінцевого покупця або кінцевого споживача. Канали збуту можуть включати оптовиків, роздрібних торговців, дистриб'юторів і навіть Інтернет.
- Повторювана нейронна мережа (RNN) — це клас штучних нейронних мереж, у яких зв'язки між вузлами можуть створювати цикл, що дозволяє виводу з деяких вузлів впливати на наступний вхід до тих

самих вузлів. Це дозволяє йому демонструвати тимчасову динамічну поведінку. Похідні від нейронних мереж прямого зв'язку, RNN можуть використовувати свій внутрішній стан (пам'ять) для обробки послідовностей вхідних даних змінної довжини. Це робить їх застосовними для таких завдань, як несегментоване розпізнавання пов'язаного рукописного тексту або розпізнавання мовлення. Рекурентні нейронні мережі теоретично повні за Тьюрингом і можуть запускати довільні програми для обробки довільних послідовностей вхідних даних;

- радіально-базисні функції: ця мережа досить компактна, швидко навчається і характеризується наявністю прихованого шару з радіальних елементів та вихідного шару з лінійних елементів;
- самоорганізаційна карта Кохонена – такий клас мереж, як правило, успішно виконується в задачах розпізнавання і має всього два шари, кожен з яких складений з радіальними елементами.

Далі ми розберемо нечітку систему. Вони базуються на нечітких множинах і теорії нечіткої логіки. Далі ми розберемо нечітку систему. Вони базуються на теорії нечітких множин і принципах нечіткої логіки. Можна помітити, що нечітка логіка — це вільний набір правил, у якому радикальні ідеї, інтуїтивні здогадки та накопичений досвід експертів у суміжних галузях можуть бути використані для досягнення поставлених цілей. Характерною рисою нечіткої логіки є відсутність суворого стандарту. Найчастіше використовується в експертних системах, нейронних мережах і системах штучного інтелекту. Замість «правди» в традиційному розумінні і «Брехня» в нечіткій логіці має ширше значення

Включіть «правда», «брехня», «можливо», «іноді», «не пам'ятаю» («все одно», «чому б і ні», «я ще не вирішив», «не скажу», тощо).

Нечітка логіка просто незамінна в ситуаціях, якщо на питання немає однозначної відповіді (так чи ні; «0» або «1») або коли всі можливості положення невідомі заздалегідь. Наприклад, у нечіткій логіці визначення у вигляді «X — це велике число» інтерпретуються як такі, що мають неточне значення, і характеризуються певним нечітким набором.

Алгоритми нечіткого висновку в основному розрізняються за типом використовуваних правил, типу Однак головна відмінність між цими двома методами полягає в тому, що в стратегії моменту фактор імпульсу фіксований і вгадується користувачем, тоді як у методі сполученого градієнта він автоматично коригується під час навчання.

Логічної операції та методом уточнення. Також варто відзначити, що нечітка логіка вплинула на інші парадигми штучного інтелекту. Таким чином, поєднання його принципів з методами інших напрямків потенційно до нових напрямків із такими відмінностями:

- адаптивні нечіткі системи;
- нечіткі асоціативні правила;
- нечіткі когнітивні карти;
- нечіткі запити;
- нечіткі нейронні мережі;
- нечітку кластеризацію.

Альтернативи штучному інтелекту використовуються в різних комбінаціях для створення гібридних інтелектуальних систем. Можна виділити основні переваги цієї нечіткої системи:

- можливість неоднозначної формалізації критеріїв оцінювання та порівняння: використовувати неоднозначні критерії («основний», «більшість» тощо);
- оцінити вірогідність вхідних даних і вихідних результатів;
- можливість роботи з неоднозначними вхідними даними;
- Можливість порівняльного аналізу складних динамічних систем із заданою точністю та можливість швидкого моделювання подібних систем.

У нашому остаточному підході до обчислювального ШІ ми розглянемо еволюційні обчислення. Тому частина еволюційних обчислень (або еволюційного моделювання) в основному базується на використанні еволюційних алгоритмів,

які часто використовуються для вирішення задач комбінаторної оптимізації, планування, розкладу, розрахунку маршруту, а також проблем розташування та транспортування. Еволюційні алгоритми, у свою чергу, є алгоритмами, заснованими на концепції біологічної еволюції, тобто на основі процесу відбору, мутації та розмноження. Яскравим прикладом такого алгоритму можна назвати генетичний алгоритм.

Суть цих алгоритмів полягає в наступному. Поясніти задачу, яку потрібно розв’язати, щоб її розв’язок можна було виразити у вигляді масиву. Далі кожному вектору присвоюється відповідне значення придатності шляхом випадкового генерування деяких початкових елементів у результуючому масиві, які згодом оцінюються за допомогою функції придатності виживання. Після цього елементи наступного покоління створюються на основі «генетичних операторів», які кросинговерують і мутують елементи попереднього покоління, і цим елементам дозволяється перейти на наступний етап (за допомогою операторів відбору). Елементи нового покоління також оцінюються, а потім до них застосовуються оператори відбору, кросинговеру, оператори мутації тощо. Це процес моделювання еволюційного процесу. Він триває кілька життєвих циклів (такий цикл називається генерацією), поки не будуть виконані умови для зупинки алгоритму.

Генетичні алгоритми в основному застосовуються для:

- складання розкладів;
- оптимізації функцій;
- налагодження і навчання нейронних мереж;
- завдання компонування;
- апроксимації функцій.

Основними перевагами генетичного алгоритму є:

- Не потрібно ніяких спеціальних знань про проблему, яку потрібно розв’язати;
- прозорість реалізації та концептуальна простота;

- Простота кодування вихідних і вхідних даних;
- Варіативність вибору типу досліджуваного параметра системи;
- можливість використання алгоритму для широкого кола завдань без істотних змін його структури;
- Можливість розпаралелювання .

Модель CNN особливо популярна у сфері розпізнавання зображень; він використовувався в багатьох найсучасніших програмах штучного інтелекту, включаючи розпізнавання облич, оцифрування тексту та обробку природної мови. Інші способи використання включають виявлення перефразування, обробку сигналів і класифікацію зображень.

MLP – це моделі нейронних мереж, які працюють як універсальні апроксиматори, тобто вони можуть апроксимувати будь-яку безперервну функцію [180]. Наприклад, їх можна використовувати як моделі SEE. MLP складаються з нейронів сприйняття. Отже, перш ніж пояснювати загальну структуру MLP, буде пояснено загальну структуру персептрона [372]. Як показано на малюнку 24.1, персептрон отримує n характеристик як вхідні дані (x = x1, x2,…, xn), і кожна з цих характеристик пов’язана з вагою. Вхідні елементи мають бути числовими. Таким чином, нечислові функції введення повинні бути перетворені в числові, щоб використовувати персептрон. Наприклад, категоріальна ознака з p можливими значеннями може бути перетворена на p вхідних ознак, що представляють наявність/відсутність цих значень. Вони називаються фіктивними змінними. Наприклад, якщо вхідна функція «тип розробки» може приймати значення «нова розробка», «покращення» або «повторна розробка», її можна замінити трьома фіктивними змінними «нова розробка», «покращення» та «перепланування», які приймають значення 1, якщо відповідне значення є, і 0, якщо воно відсутнє.

Багатошаровий персептрон (MLP) — це ще один процес штучної нейронної мережі, що містить кілька рівнів. В одному персептроні можна розв’язувати чітко лінійні проблеми, але він не дуже підходить для нелінійних випадків. Для вирішення цих складних проблем можна розглянути MLP. З архітектури мережі,

представленої на рис. 2.9, можна помітити, що MLP є прямою нейронною мережею, поєднаною з кількома рівнями. Як правило, він використовується для різних цілей машинного навчання.

Багатошаровий перцептрон (MLP) є доповненням до прямої нейронної мережі. Він складається з трьох типів шарів — вхідного, вихідного та прихованого, як показано на рис. 3. Вхідний рівень отримує вхідний сигнал для обробки. Потрібне завдання, таке як прогнозування та класифікація, виконує вихідний рівень. Довільна кількість прихованих шарів, розміщених між вхідним і вихідним шарами, є справжнім обчислювальним механізмом MLP. Подібно до прямої мережі в MLP, дані проходять у прямому напрямку від вхідного до вихідного рівня. Нейрони в MLP навчаються за допомогою алгоритму навчання зворотного поширення. MLP розроблені для наближення будь-якої безперервної функції та можуть вирішувати проблеми, які не є лінійно роздільними. Основними випадками використання MLP є класифікація шаблонів, розпізнавання, прогнозування та наближення.

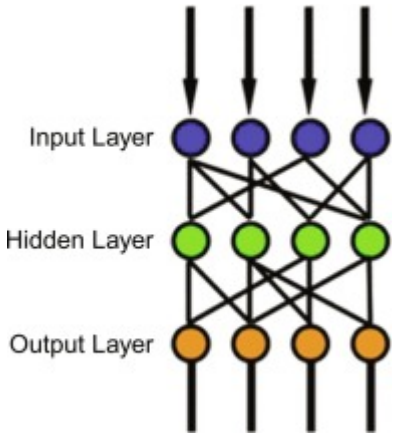


Рисунок 2.9 – Схематичне зображення MLP з одним прихованим шаром.

Обчислення, які відбуваються на кожному нейроні вихідного та прихованого шару, такі:

$$o(x) = G\big(b(2) + W(2)h(x)\big) \tag{2.2}$$

$$h(x) = \Phi(x) = s(b(1) + W(1)x)$$

з векторами зміщення b(1), b(2); вагові матриці W(1), W(2) і функції активації G і s. Набір параметрів для вивчення — це набір θ = {W(1), b(1), W(2), b(2)}. Типові варіанти для s включають функцію tanh з $\tanh(a) = (e^a - e^{-a})/(e^a + e^{-a})$ або логістичну сигмоїдну функцію з $\text{sigmoid}(a) = 1/(1 + e^{-3})$.

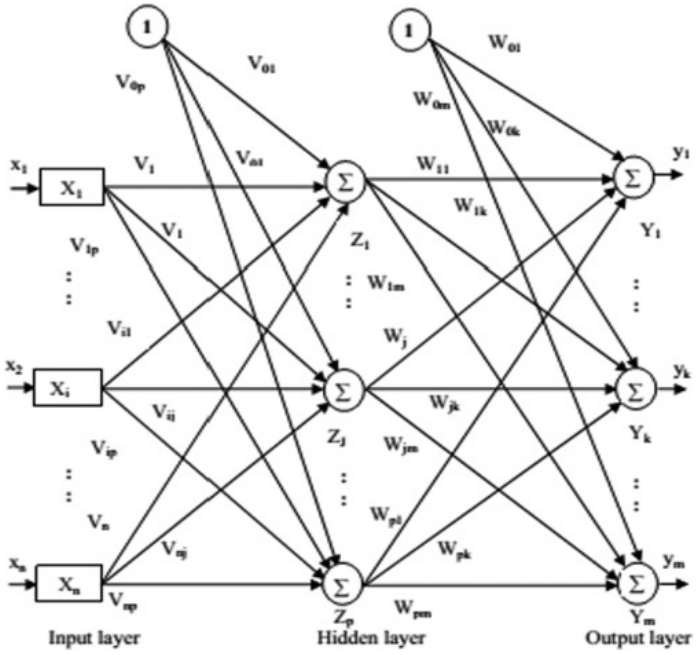


Рисунок 2.10 – Архітектура MLP.

Ваги мережі встановлюються у випадковому порядку перед початком навчання. Після завершення етапу навчання за допомогою навчальних даних(x_1,x_2,y) модель перевіряється. У навчальному наборі дані x_1 і x_2 є вхідними, а у відповідним очікуваним виходом вхідних даних. Результат залежить від нейронів і ваги нейронної мережі. Його можна представити

$$y = \omega x + b \tag{2.3}$$

де w — вага, b — зсув.

Зважений вектор і вектор нахилу основного шару створюють зважену мережу і називаються вектором нахилу наступного шару для нелінійного

компонента . Вихід пов'язаний із внесками різних нейронів у покритому шарі і не помітний у виході. прибутковість присвоюється як 0 і. 0 означає відсутність діабету, а 1 означає діабет.

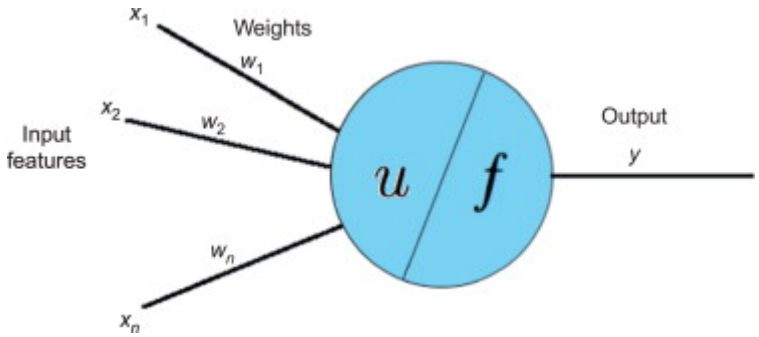


Рисунок 2.11 – Схема персептрона з n вхідними ознаками.

Вхідні ознаки передаються до вхідної функції u, яка обчислює зважену суму вхідних ознак:

$$u(x) = \sum_{i=1}^{n} \omega_i \, x_i \tag{2.4}$$

Результат цього обчислення потім передається на функцію активації f, яка вироблятиме вихідні дані персептрона. В оригінальному персептроні функція активації є ступінчастою функцією:

$$y = f\big(u(x)\big) = \begin{cases} 1, \textit{якщо}\, u(x) > \emptyset \\ 0, \textit{інакше}, \end{cases} \tag{2.5}$$

де θ – пороговий параметр. Приклад ступінчастої функції з θ = 0 показано на рис 2.10 . Таким чином, ми бачимо, що персептрон визначає, чи є $w_1x_1 + w_2x_2 + \cdots$ + wnxn − θ > 0 істинним чи хибним. Рівняння $w_1x_1 + w_2x_2 + \cdots$ + wnxn − θ = 0 є рівнянням гіперплощини. Персептрон виводить 1 для будь-якої вхідної точки над гіперплощиною та виводить 0 для будь-якого входу на або під гіперплощиною. З

цієї причини перцептрон називають лінійним класифікатором, тобто він добре працює для даних, які можна розділити лінійно. Навчання персептронів полягає в регулюванні ваг таким чином, щоб визначити гіперплощину, яка добре розділяє навчальні дані. Ми відсилаємо читача до Gurney для отримання додаткової інформації про алгоритм навчання персептрона.

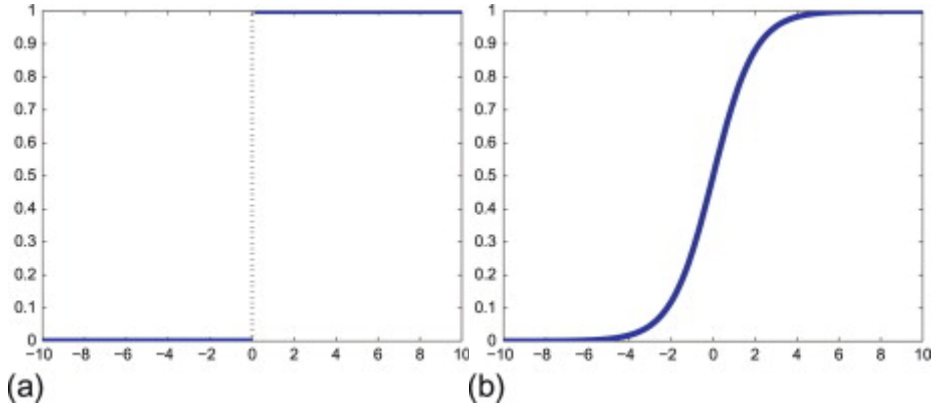


Рисунок 2.12 – Приклади функцій активації. (a) Приклад ступінчастої функції та (b) Приклад сигмоїдної функції.

MLP здатні апроксимувати будь-яку неперервну функцію, а не лише лінійні функції. Вони роблять це шляхом поєднання кількох нейронів, які організовані принаймні в три шари:

- Один вхідний шар, який просто розподіляє вхідні функції на перший прихований шар.
- Один або кілька прихованих шарів персептронів. Перший прихований шар отримує як вхідні дані функції, розподілені вхідним шаром. Інші приховані шари отримують як вхідні дані кожного персептрона з попереднього шару.
- Один вихідний рівень персептронів, які отримують як вхідні дані кожного персептрона останнього прихованого шару.

На рис 2.12 показана схема MLP з трьома шарами. Персептрони, що використовуються MLP, часто використовують інші типи функцій активації, ніж функція кроку. Для нейронів прихованого шару часто використовуються

сигмоподібні функції. Приклад сигмоїдної функції показано на малюнку 24.2b. Сигмоїдні функції призведуть до плавних переходів замість жорстких меж рішень, як при використанні ступінчастих функцій. Функція активації нейронів вихідного рівня зазвичай є сигмоподібною для проблем класифікації та функцією ідентичності для задач регресії.

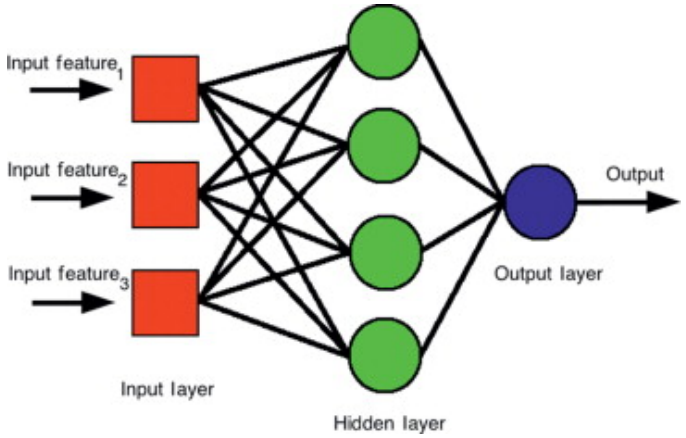


Рисунок 2.13 – Схема тришарового MLP з трьома вхідними функціями, чотирма прихованими нейронами та одним виходом.

Навчання в MLP також полягає в регулюванні ваг персептронів таким чином, щоб забезпечити низьку похибку даних навчання. Традиційно це робиться за допомогою алгоритму зворотного поширення, який намагається мінімізувати MSE. Однак можна використовувати й інші алгоритми. У цьому розділі ми покажемо, як використовувати MOEA для навчання MLP на основі кількох різних показників ефективності. Стратегії також можуть бути використані, щоб уникнути переобладнання навчальних даних, тобто уникнути створення моделей, які мають низьку прогностичну продуктивність через випадкову помилку моделювання або шум, присутній у навчальних даних. Уникнення переобладнання часто передбачає допускання вищої похибки в навчальних даних, ніж та, яка може бути досягнута за допомогою моделі.

Отже, з вже вивчених детально нейронних мереж. Можна сміливо сказати яка нейронна мережа підходить саме в нашому випадку. Це саме багатошаровий персептрон. Тому що саме він має змогу обробляти велику кількість інформації. В

нашому випадку з медичними препаратами так і є.

Подібним чином розберемо існуючі засоби в предметній області. Так, в Інтернеті є багато сервісів для пошуку медичних препаратів (проте вони мають багато недоліків).

По-перше, пошуки здійснюються не за допомогою штучних нейронних мереж, а за допомогою звичайних пошуків в Інтернеті.

По-друге, всі наявні в інтернеті аналоги не дозволяють шукати ліки за клінічним діагнозом.

Отже, виходячи з вищевикладеного, можна зробити висновок про актуальність даної роботи, оскільки переважна більшість сучасних засобів пошуку медичних виробів мають велику кількість недоліків, головним з яких є низька якість результатів.

2.3 Постановка задачі

З огляду на все вище сказане, можна зробити висновок, що медицина це дуже важливо на сьогоднішній час і про здоров’я потрібно дбати. Своєчасно звертатись до лікаря за допомогою. Багато людей недооцінюють роботу технологій та їхній прогрес в сучасному світі. Можна сміливо сказати, що технологій на сьогодення рятують людям життя не тільки в медицині а й в багатьох інших галузях.

Крім того в сучасних умовах потрібно берегти своє здоров’я.

На основі опрацювання предметної області та дослідження методів, моделей та засобів прийняття рішень при виборі системи пошуку медичних препаратів, котрі в процесі роботи повинні бути вирішені, щоб вдосконалити штучну нейронну мережу яка підвищує продуктивність. Це буде робитись шляхом поповнення ваг і рівнів упередженості мережі. Правило навчання є одним з факторів, який визначає, як швидко або наскільки точні можуть бути розроблені штучні мережі.

З усіх парадигм навчання, а їх існує всього три це навчання з вчителем, без вчителя та змішане . Було прийнято рішення обрати саме змішане навчання тому що цей метод навчання найкраще підходить для пошуку великої кількості інформації та вибору найважливішого із знайденого в нашому випадку це медичні препарати . Пошук буде базуватись на класифікації медичних препаратів.

Для вдосконалення системи пошуку потрібно багатошаровий персептрон навчити навчальної вибірки для вирішення питань які перед ним поставлені. В нашому випадку це пошуку медичних препаратів за клінічним діагнозом. Що саме має ця система пошуку вона має при веденні клінічного діагнозу . З використанням багатошарового персептрона він виконує пошуку медичних препаратів які саме підходять під цей клінічний діагноз. Представлені приклади вище пояснюють, як це робить саме багатошаровий персептрон.

2.4 Висновки

Отже, підсумуючи вищесказане можна зробити висновки, що нейронні мережі дуже різноманітні та має попит в різних сферах. Полегшує роботу багатьох систем зокрема і система пошуку медичних препаратів. Було проведено дослідження типів нейронних мереж прийнято рішення обрати багатошарову нейронну мережу. Бо ця нейронна мережа має змогу обробляти великі кількості інформації та серед багатьох класифікацій обирає саме те що потрібно. В розділі було детально досліджено всі типи нейронних мереж. Для вдосоконалення системи пошуку було прийнято рішення використати навчання нейронних мереж як предмет дослідження. В наступному розділі буде розповідатись про навчання нейронних мереж. Та вибір методу навчання нейронних мереж та вдосканелння системи пошуку яка буде обрана.

3 НАВЧАННЯ НЕЙРОННИХ МЕРЕЖ

Штучні нейронні мережі призначені для функціонування як біологічні нейронні мережі. У біологічному процесі дендрит спочатку отримує якийсь імпульс. Потім цей імпульс передається через аксон і, нарешті, вивільняється на терміналі аксона. Ці три етапи подібні до трьох шарів, які складають штучний нейрон. Перший рівень — це вхідний рівень, у який ми вводимо значення, які будемо використовувати для прогнозування. Другий рівень називається прихованим, у якому ми маємо деяку функцію активації, яка трансформує наші вхідні дані на додаток до їх сумування. Третій рівень — наш вихідний рівень, на якому ми бачимо спостережуваний результат нашого нейрона в очікуваному форматі. Ця остання частина означає, що якщо ми очікуємо отримати результат «так» або «ні», ми не отримаємо жодного іншого виду результату, наприклад числа. Справжня сила нейронних мереж, як біологічних, так і штучних, походить від взаємодії нейронів.

Найважливішою властивістю нейронних мереж є їхня здатність навчатися з даних у предметній області та підвищувати продуктивність шляхом навчання. Продуктивність зростає з часом відповідно до певних правил. Процес навчання можна розглядати як визначення архітектури мережі та ітераційне коригування ваг синаптичних з’єднань для ефективного виконання конкретного завдання.

3.1 Дослідження алгоритму

FFANN складається з набору простих нейронів (або блоків обробки). Нейрон описується двома сутностями: його функцією активації та його зміщенням або порогом. Він приймає вхідні дані від інших нейронів або із зовнішніх джерел. Набір нейронів у FFANN розділений на кілька непересічних підмножин. Усі нейрони в одному розділі віднесені до одного шару. Наприклад, у двошаровій FFANN нейрони розділені на дві непересічні підгрупи, по одній для кожного шару. Кожен нейрон одного шару з’єднаний з усіма нейронами

наступного шару зв’язками з регульованою силою/вагою. Як показано пізніше, вага з’єднання може змінюватися під час процесу навчання. Нехай о(— і-й нейрон у С-му шарі. Вхідний рівень називається Oth-шаром, а вихідний — oth-шаром. Нехай n — кількість нейронів у f-му шар. Вага ланки між нейрон u^l у шарі 1 і нейрон u “ ' в шар 1 + 1 позначається mJj. Дозволяє 1›’2 означає, що X_p — це набір вхідних даних, які має мати мережа навчиться класифікувати та множити (d,, dg, ... , d p] відповідні бажані вихідні шаблони. Слід зазначити, що xp є ri_0-вимірним вектором (x, p , i d p є n-вимірним вектором (x, p, x_{2p} ,…,x_{noP} і dp є n-вимірним вектором (d $_{ld\ p2}$, d, p). Відро (x p, d p) називається шаблоном навчання.

Де Q визначає крутизну функція активації. Менше значення Q дає більш плавний перехід від нижчого до більш високої активації Iesel як emp змінює своє значення; більш високе значення для 9 викличе ступінчастий перехід рівня активації. У решті статті ми припускаємо, що значення Q = 1.

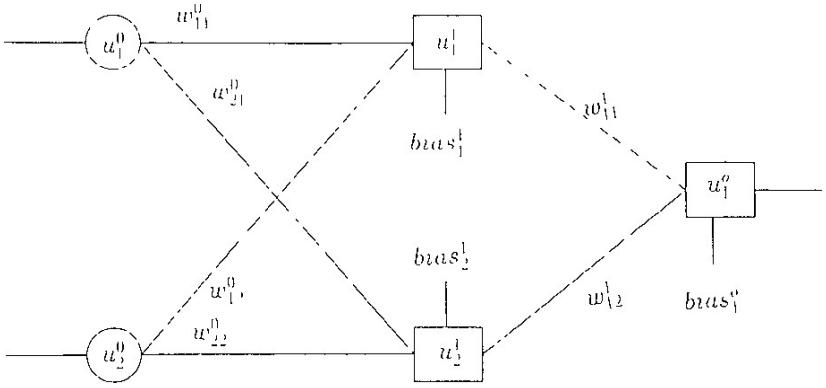


Рисунок 3.1 – Трьох шарова нейронна мережа.

Алгоритм EBP для FFANN працює шляхом представлення вхідного шаблону на вхідний (або Oth) рівень, після чого мережа створює вихідні дані. Цей вихід порівнюється з бажаним або цільовим результатом. Різниця між цільовим виходом і фактичною мережею називається помилкою. Формально помилка *ep* для і-го нейрона u,° вихідного рівня о для вхідної тренувальної пари (*x p, dp1* обчислюється як

$$\varepsilon_{pi} = d_{pi} - out_{pi}^{o}. \tag{3.1}$$

Метою алгоритму навчання є використовуйте цю помилку, щоб налаштувати вагові коефіцієнти таким чином, що помилка поступово зменшується. Тренувальний процес припиняється, коли помилка кожного нейрона для кожної тренувальної пари знижується до прийнятного рівня, або коли не досягається подальше покращення. В останньому випадку мережа знову ініціалізується малими вагами, і процес навчання починається заново.

Використовуючи ці значення для ваг зв'язку та зміщення, можна обчислити чисті входи та відповідні виходи нейронів для кожного вхідного шаблону. наприклад, розглядаючи шаблон x1 як вхідні дані для мережі, ми отримуємо чисті вхідні дані два нейрона в 1-му шарі як

$$net_{11} =^{2j} E_2! \ \omega_0 x + bias1 = 0.4, i \, neft_2 = \sum_{j}^{2} = 1 + djs = -0.2 \tag{3.2}$$

Тепер, застосовуючи ці чисті входи до функції активації, ми отримуємо вихід нейронів у 1-му шарі як outtt - y net') —— 1/(1+e)$^{-net1/11}$ 0,599, і OU12 ffnet 12) = 1/(1 e)$^{‘\ e''!^{\circ}}$ - 0 450.

Ці виходи 1-го шару діють як вхідні дані для наступного рівня (який є вихідним рівнем у цьому прикладі), і для нейрона у вихідному шарі ми отримуємо net = 0,124 і out, 1 = 0,531. Оскільки бажаний вихід із вхідним шаблоном x є d = [0], помилка — 0,469. , = d — вихід —0.469.

Для вимірювання продуктивності алгоритм навчання цільова функція визначається таким чином, що зі зменшенням помилки зменшується і значення мети. Таким чином, навчальний алгоритм вирішує зміну вагових коефіцієнтів за допомогою певної процедури, яка гарантує відсутність збільшення значення цільової функції. Цільові функції відомі як функції енергії (від назви подібних функцій у фізиці). EBP використовували число квадратів помилки як функцію енергії.

$$E = \frac{1}{2}\sum_{p=1}\sum_{l=1}\left(\varepsilon_{pl}\right)^2 \tag{3.3}$$

Енергетичну функцію можна визначити для лише одна пара шаблонів навчання (*px, dp*) як представлено на прикладі нижче.

$$E_p = \frac{1}{2}\sum_{l=1}^{n_0}\left(\varepsilon_{pl}\right)^2 \tag{3.4}$$

Таблиця 3.1 Версії алгоритму EBP

Версії алгоритму EBP	Дослідження версій	Формули
Онлайн версія	В онлайн-алгоритмі EBP ваги оновлюються з використанням помилки, що відповідає кожному шаблону навчання. Цей метод використовує енергетичну функцію, визначену рівнянням	$E = \frac{1}{2}\sum_{p=1}\sum_{t=1}\left(\varepsilon_{p1}\right)^2$
Пакетна версія	Проте в пакетному алгоритмі EBP ваги оновлюються після накопичення помилок, що відповідають усім вхідним шаблонам, і, таким чином, використовується функція енергії.	$net_{11} =^{2j} E_2^{'} w_0 + bias\,1 = 0{,}4, i\,nef = \sum_j 2 = v^0 i + djs =- 0.2$

Існує дві версії алгоритму ЕВР: онлайн і пакетна. В онлайн-алгоритмі ЕВР ваги оновлюються з використанням помилки, що відповідає кожному шаблону навчання. Цей метод використовує енергетичну функцію, визначену рівнянням (3.5). Проте в пакетному алгоритмі ЕВР ваги оновлюються після накопичення помилок, що відповідають усім вхідним шаблонам, і, таким чином, використовується функція енергії, визначена рівнянням (3.4). Правило оновлення ваги для пакетного режиму задано

$$G_{ij}^{l}[s] = -\frac{\partial E}{\partial \omega_{ij}^{l}[s]'} \tag{3.5}$$

Індекс s позначає номер ітерації або кроку в процесі навчання, і є розмір кроку або коефіцієнт швидкості навчання. Для онлайн-алгоритму ЕВР правило оновлення ваги отримуємо, якщо енергію А в рівнянні (3.8) замінити на енергію Ер для обчислення градієнта ваги енергії.

Вирази для *6E p/ d netp;“ '* відрізняються для нейронів прихованого шару та нейронів вихідного шару. Таким чином, зручно позначати *dEp/d net p* δ_{pi}^{l-} ' як для нейронів прихованого, так і для вихідного шарів, щоб отримати правило оновлення ваги для всіх нейронів як

$$\omega_{ij}^{l}[s+1] = \omega_{ij}^{l}[s] + n\delta_{pj}^{l+1} out_{pj}^{l}[s] \tag{3.6}$$

Для обчислення виразів для *bp,“ ’* використовується ланцюгове правило диференціювання. Ми пропускаємо похідні, але пишемо для них вирази. Передбачається, що рівні активації нейронів обчислюються за допомогою стандартної сигмоїдної функції Рівняння (3.2) з Q — 1:

$$\delta_{pi}^{0} = out_{pl}^{0}\left(1 - out_{pi}^{0}\right)\varepsilon_{pi} \tag{3.7}$$

Для нейрона і у прихованих шарах. З рівняння (3.12) ясно, що сигнал помилки b обчислюється безпосередньо з відповідною помилкою *′ op; dp; — out p;,* але сигнал помилки для нейронів у прихованих шарах отримується шляхом розповсюдження сигналів помилки нейронів у шарі відразу після нього: тому назва Помилка Баха Пропагування (рис 3.5) ілюструє обчислення сигналів помилок *bp;* для нейрон і в шарі *l + 1*.

Поєднуючи рівняння (3.12) з рівнянням (3.13), можна обчислити поправки ваги для нейронів вихідного рівня. Подібним чином корекцію ваги для нейронів прихованого шару можна обчислити шляхом поєднання рівняння (3.11) з рівнянням (3.13). Процес навчання FFANN є ітеративним процесом. Кожна ітерація складається з наступних кроків:

- Виберіть навчальний шаблон (x p, dp).
- Forward Pass: презентація введення шаблон px на вхід FFANN і визначити його вихід. Рівняння (3.l) і (3.2) необхідні для виконання обчислень для цього проходу.
- Зворотний перехід: обчисліть сигнал помилки для кожного нейрона. Слід зазначити, що обчислення сигналу помилки починається з вихідного шару та продовжується у зворотному напрямку до прихованих шарів. Наприклад, якщо FFANN має чотири рівні — один вхід.
-

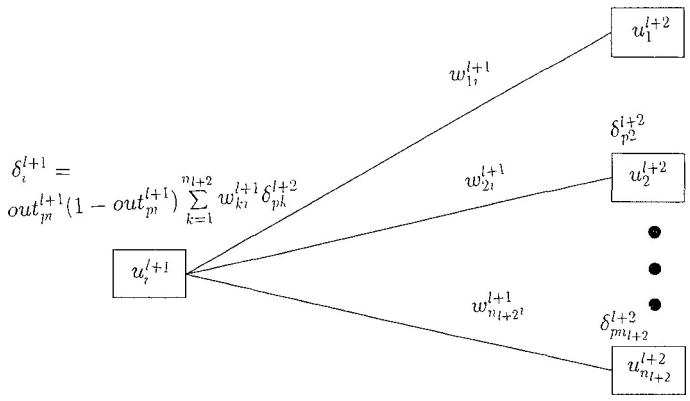


Рисунок 3.2 – Обчислення сигналу помилки для нейрона і прихованого шару в шарі 1 — — 1.

Шар, два прихованих шару та один вихідний шар — потім (а) спочатку обчислюються сигнали помилок для нейронів вихідного шару, потім (б) обчислюються сигнали помилок для нейронів другого прихованого шару (поруч із вихідним шаром), і нарешті (с) обчислюються сигнали помилок для нейронів першого прихованого шару (найближчих до вхідного шару).

(3.4) Коригування ваги: відрегулюйте ваги за допомогою рівняння (3.12).

На останньому кроці за допомогою рівняння (3.11) вагові коефіцієнти коригуються, щоб отримати w^0_{11}—0,3, in'2 = — 0,6 і зсув = 0,377759 для нейрона u1 у прихованому шарі; m2 0,2, 0 1 = 0,8, bias $^{1/2}$ — — 0,229459 для нейрона u2 у прихованому шарі; in= 0,620827,'12 = 0,840468 і bias° = — 0,832244, для нейрона u t° у вихідному шарі. Для коригування ваги ми використовували коефіцієнт швидкості навчання — 1. (Для узгодженості ми використовуємо коефіцієнт швидкості навчання = 1 у всіх наших наступних прикладах.) Завершує одну ітерацію з вхідним шаблоном x. Оскільки ваги були оновлені після представлення лише одного шаблону, це онлайн-версія алгоритму навчання.

Ми надаємо шаблони введення $x_{,,}$ x_2, z_3 та x_4 у такому порядку в мережу в кожному циклі для легкого відтворення результатів. (Однак вони можуть бути представлені в будь-якому порядку, але, можливо, впливаючи на кінцевий результат.) Передбачалося, що мережа вивчила всі шаблони, якщо абсолютне значення помилки було менше 0,3 для кожного шаблону. Онлайн-алгоритму знадобилося 420 циклів, щоб вивчити всі шаблони. Остаточні ваги підключення мережі були такими: w^0_{11} = 3,30961, 2 = — 3,74105, і *bias*1 = — 2,06024 для нейрона 1 ' прихований шар; 0 - — 3,32466, *bias* 1^{21} 2,99838 і зміщення 1 = — 1,81799 '22 для нейрон о} у прихованому шарі; w^0_{11}— 3,75645, t_2 - 3,73334, і *bias* —— 1,74346, для нейрона о{ у вихідному шарі.

Пакетна версія. Тепер ми проілюструємо пакетну версію алгоритму навчання EBP за допомогою того самого зразка FFANN. Можна нагадати, що в пакетній версії: кожна модель представлена один раз і розраховується корекція ваги, але ваги фактично не коригуються; і розраховані поправки ваги для кожної ваги додаються разом для всіх шаблонів, а потім коригуються лише ваги один раз

за допомогою кумулятивної корекції. Що стосується онлайн-версії, ми представляємо шаблони введення *x, z t, z 3 i x4* у такому порядку в мережу в кожному циклі. Однак слід зазначити, що в пакетній версії алгоритму EBP порядок представлення шаблонів у мережі не має значення, оскільки ми регулюємо ваги лише один раз після того, як усі шаблони представлені та накопичені помилки для ваг. обчислюється.

Після першого циклу скориговані ваги мережі становлять: w^0_{11} **0,292659,** $^0_{12}$— 0,605081 і 0,389771 для шару нейронів; m_2t = 0,197833, прихований w^0_{22} 0,796052 і òias^1— — 0,208627 для нейрона u^1_1 у прихованому шарі; > ^{1}ii = 0,671218, $_{12}$ = 0,871077 і òias{ = — 0,750384, для нейрона u{ у вихідному шарі. Ця версія алгоритму вимагала 491 циклу, щоб досягти бажаного рівня помилки (меншого за абсолютне значення 0,3) для всіх шаблонів. Далі наведені ваги мережі після 491 циклу. Вона повинна була зазначити, що отримані ваги сильно відрізняються від тих, що отримані онлайн-версією. Остаточні ваги підключення мережі були такими:'ll^0 = —2,45067, $_{,12}^0$ = — 2,47898 і òias* = 3,51763 для нейрона u, у прихованому шарі; $_{,21}^0$ — 5,16276, $_{,22}^0$ 5,33647, створені для прискорення початкового EBP, можна розділити на групи: (1) Динамічні та *òios2* = — 1,86421 для нейрона e_2 iI1 прихований шар; m tt = 4,12527, w} = 4,30574, і зміщення° - — 5,9717, для нейрона u,° у вихідному шарі.

Для реалізації алгоритму EBP важливі значення кількох параметрів. Початкове значення ваг має бути невеликим і вибраним випадковим чином щоб уникнути проблеми симетрії» використовували рівномірно розподілені випадкові числа від -0,5 до 0,5 як вагові коефіцієнти зсуву та від -0,5/*ii*) до (0,5/*nt*) як початкові вагові коефіцієнти для зв'язків між рівнями I та *l + 1*. Примітка що ділення на n, кількість входів, усуває вплив кількості входів на нейрон. Цінність відіграє дуже важливу роль. Менше значення робить навчання повільним, але занадто велике значення викликає коливання, заважаючи мережі вивчати завдання . На практиці найбільший ефективне значення залежить від проблеми. Відомо, що 0,9 як коефіцієнт швидкості навчання для однієї проблеми, відомо про 0,002 як коефіцієнт швидкості навчання для іншої задачі. Ця зміна значення

коефіцієнта швидкості навчання для швидшого навчання FFANNs привернула увагу багатьох дослідників. Були запропоновані різні методи для прискорення алгоритму EBP. У наступному розділі подано короткий огляд цих методів.

Таблиця 3.2 Дослідження алгоритмів EBP та FFANN

Дослідження	Результати дослідження	Використані формули для дослідження
Алгоритм EBP	Для FFANN працює шляхом представлення вхідного шаблону на вхідний (або Oth) рівень, після чого мережа створює вихідні дані. Формально помилка *ep* для і-го нейрона u,° вихідного рівня о для вхідної тренувальної пари (*x p, dp1* обчислюється як	$\varepsilon_{pl} = d_{pl} - out_{pl}^{0}$
Алгоритм FFANN	Складається з набору простих нейронів (або блоків обробки). Нейрон описується двома сутностями: його функцією активації та його зміщенням або порогом. Він приймає вхідні дані від інших нейронів або із зовнішніх джерел. Набір нейронів у FFANN розділений на кілька непересічних підмножин	$netp\}^{'n_l}\ \omega\, out^{'} + bras,^{l^{''}}\ j = 1$

3.2 Огляд різних модифікацій

Було зрозуміло, що оригінальний алгоритм EBP надто повільний для більшості практичних застосувань, і було запропоновано багато модифікацій для його прискорення. Перш ніж описувати різні модифікації оригінального алгоритму EBP у розділі 4, ми коротко обговоримо їх тут, щоб побачити, як вони пов’язані між собою.

Дерево класифікації показано на малюнку 3, щоб допомогти нашому обговоренню. Як видно з малюнка, методи, запропоновані для прискорення оригінального EBP, можна розділити на групи: (1) Динамічне коригування швидкості навчання, де значення

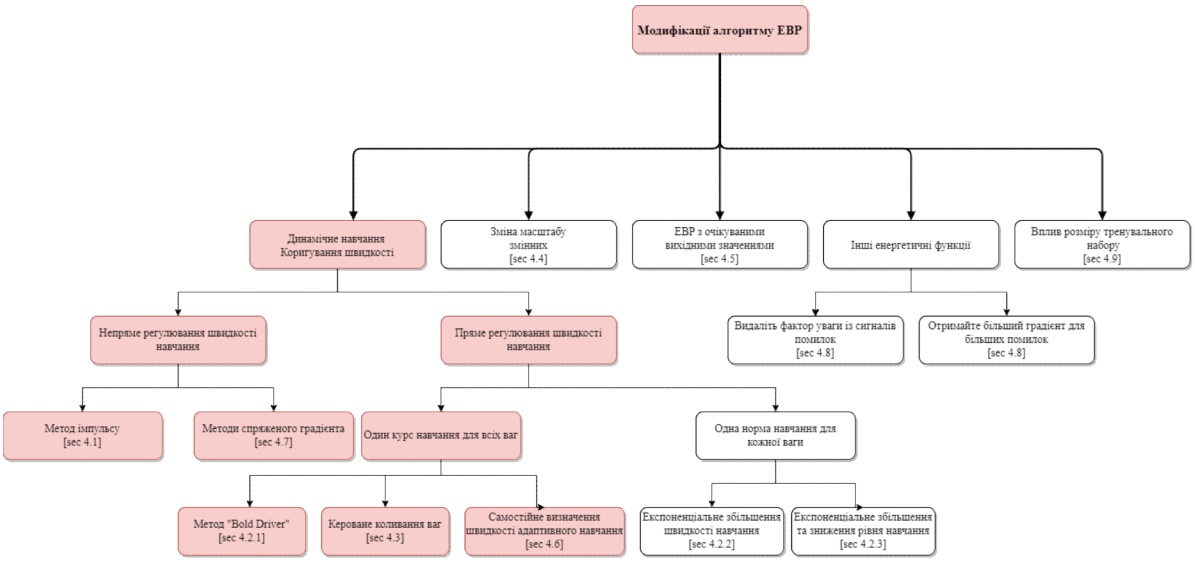


Рисунок 3.3 – Класифікація різних методів прискорення алгоритмів навчання EBP.

Коефіцієнтів швидкості навчання коригуються в міру продовження навчання. Коригування коефіцієнтів швидкості навчання може здійснюватися опосередковано або прямо. Прикладами непрямого регулювання швидкості навчання є метод імпульсу і методи сполучених градієнтів. Методи непрямого регулювання швидкості навчання, особливо імпульсний метод, у деяких випадках пришвидшили алгоритм EBP. Однак у багатьох випадках вони були недостатньо

задовільними, і тому були запропоновані методи прямого регулювання коефіцієнта швидкості навчання: один коефіцієнт швидкості навчання для всіх вагових коефіцієнтів

Методи, окрім налаштування коефіцієнтів швидкості навчання, показали помітне прискорення для деяких програм.

Перемасштабування змінних (сигналу помилки) базується на спостереженні, що сигнали помилки для нейронів прихованого шару дуже швидко слабшають. Рівень ослаблення зростає експоненціально, оскільки відстань прихованого шару від вихідного шару збільшується лінійно. Таким чином, щоб компенсувати згасання, сигнали помилок масштабуються Іншим джерелом покращення є використання очікуваного виходу нейрона замість поточного виходу. Оригінальний алгоритм EBP використовує суму квадратів помилок як функцію енергії. Проте будь-яка функція похибки, яка зменшується зі зменшенням похибки, може бути використана як функція енергії. Декілька інших енергетичних функцій було використано для прискорення навчання EBP. Аналіз та експерименти показали, що розмір навчального набору є параметром, який можна використовувати для визначення швидкості навчання Коли розміри навчальних наборів для різних класів різні, використання різних коефіцієнтів швидкості навчання для них може прискорити навчання EBP.

У наступних підрозділах представлені методи модифікації оригінального алгоритму EBP з метою прискорення процесу навчання.

коефіцієнт багаторазово зменшується, доки він не дає розмір кроку, який зменшує значення енергетичної функції. Приклад продовжується, щоб проілюструвати основні ідеї, що лежать в основі методу.

Приклад (продовження). Ми зберегли початкові умови, як і раніше. Коефіцієнт підвищення коефіцієнта швидкості навчання p було встановлено рівним 1,1, а коефіцієнт зменшення коефіцієнта швидкості навчання a був встановлений рівним 0,5. На рисунку 5 зображено графіки коефіцієнта швидкості навчання та значення енергетичної функції щодо кількості кроків навчання. Для кращої ілюстрації значення енергетичної функції було помножено на 30.

Зрозуміло, що коефіцієнт швидкості навчання поступово зростає (на коефіцієнт 1,1), доки ми не досягнемо кроку навчання, де значення енергетичної функції зростає. При цьому коефіцієнт швидкості навчання знижується (у 0,5 разу).

На рисунку 6 порівнюються енергетичні значення методів імпульсу та методу «сміливого драйвера». На початку обидва мають однакове значення енергетичної функції. Але незабаром значення енергетичної функції швидко зменшується за рахунок збільшення коефіцієнта швидкості навчання (див. рис. 5). Однак поступово вони досягають відносно високого коефіцієнта швидкості навчання, і значення енергетичної функції зростає, а не зменшується. У цей момент алгоритм починає зменшувати значення коефіцієнта швидкості навчання, поки не отримає значення, яке зменшить значення енергетичної функції. Як і у випадку методу імпульсу, значення енергетичної функції повільно спадає без будь-яких коливань.

Потрібний рівень помилки був досягнутий після 199 ітерацій. Кінцеві ваги мережі були дуже схожі на ті, що були отримані методом імпульсу, але мали невеликі відмінності. Метод «Bold driver» був значно швидшим для цього прикладу, як ми і хотіли. Однак слід мати на увазі, що ситуація не завжди може бути такою хорошою.

Метод «Bold driver» є нелокальним, оскільки він зберігає лише один коефіцієнт швидкості навчання для всіх ваг.

3.3 Модифікація алгоритму EBP

У цьому розділі представлено огляд модифікацій оригінального алгоритму EBP, які були запропоновані для прискорення навчання FFANN. У міру того, як ми продовжимо, стане зрозуміло, що деякі з цих методів можна поєднати, щоб отримати алгоритм, який не тільки забезпечує швидше навчання, але й допомагає автоматизувати вибір певних параметрів, чутливих до проблеми.

3.3.1 Стратегія імпульсу

Стратегія навчання, яка використовується в оригінальному алгоритмі ЕВР, насправді є градієнтним спуском на багатовимірній енергетичній поверхні у ваговому просторі. Більш уважний погляд на вплив значення коефіцієнта швидкості навчання показує, що в області енергетичної поверхні, де градієнт не змінює знак, більше значення зменшує енергетичну функцію швидше; з іншого боку, поблизу області, де знак градієнта швидко змінюється, менше значення зберігає спуск уздовж енергетичної поверхні. Ця різнорідна потреба в значенні пропонує метод, який динамічно адаптує значення залежно від характеристик енергетичної поверхні. Стратегія імпульсу реалізує такий змінний коефіцієнт швидкості навчання неявно шляхом додавання частки останньої зміни ваги до поточного напрямку руху в просторі ваг, і це є незначною зміною правила оновлення ваги початкового ЕВР [Рівняння (7))].

Нове рівняння для зміни ваги задається формулою

$$\Delta\omega_{ij}^{l}[s+1] = nG_{ij}^{l}[s] + \alpha\Delta\omega_{ij}^{l}[s] \tag{3.8}$$

де а — коефіцієнт імпульсу, який може мати значення від нуля до одиниці.

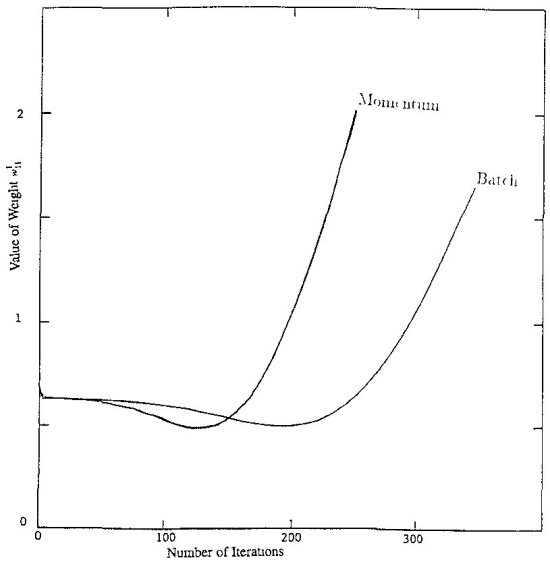


Рисунок 3.4 – Порівняння двох значень ваг з’єднання для партійної версії ЕВР з і без терміну імпульсу.

Зверніть увагу, що ми отримуємо рівняння (3.8) із наступного рівняння:

$$\omega_{ij}^{l}[s+1] = \omega_{ij}^{l}[s] + nG_{ij}^{l}[s] + \alpha\left(\omega_{ij}^{l}[s] - \omega_{ij}^{l}[s-1]\right) \tag{3.9}$$

3.3.2 Коефіцієнт адаптивної швидкості навчання

У попередньому розділі ми виявили, що коефіцієнт імпульсу неявно регулює ефективний коефіцієнт швидкості навчання динамічно залежно від природи енергетичної поверхні. Покращення часу навчання відбувається завдяки цій динамічній поведінці ефективного коефіцієнта навчання. Існують методи, де коефіцієнт навчання явно регулюється для отримання покращеної швидкості конвергенції. Далі описано три методи для явної адаптації коефіцієнта навчання.

Метод «Bold driver». Цей метод вносить дві тривіальні зміни в оригінальний алгоритм EBP: він відстежує значення енергетичної функції E, заданої рівнянням (3.4), і динамічно регулює значення коефіцієнта швидкості навчання p.

Навчання починається з деякого довільного значення коефіцієнта швидкості навчання y. Якщо значення A зменшується, швидкість навчання збільшується на коефіцієнт p> 1). Це допомагає зробити довший крок III наступної ітерації. Крім того, значення швидкості навчання експоненціально зростає в області постійного градієнта енергетичної функції. З іншого боку, якщо значення енергетичної функції A зростає, припускається, що розмір останнього кроку був занадто великим і (l) остання корекція ваги для кожної ваги скасовується, (2) значення коефіцієнта швидкості навчання зменшується на коефіцієнт w (cr < 1), і (3) виконується нове випробування. Якщо нове випробування показує зниження значення енергетичної функції, зменшений коефіцієнт швидкості навчання приймається як наступна швидкість навчання; інакше швидкість навчання.

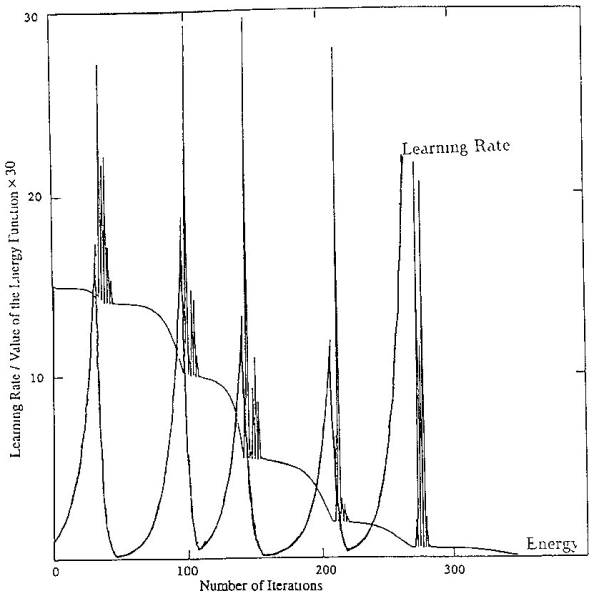


Рисунок 3.5 – Зв’язок між швидкістю навчання та енергетичною цінністю у версії EBP навчання «Bold driver».

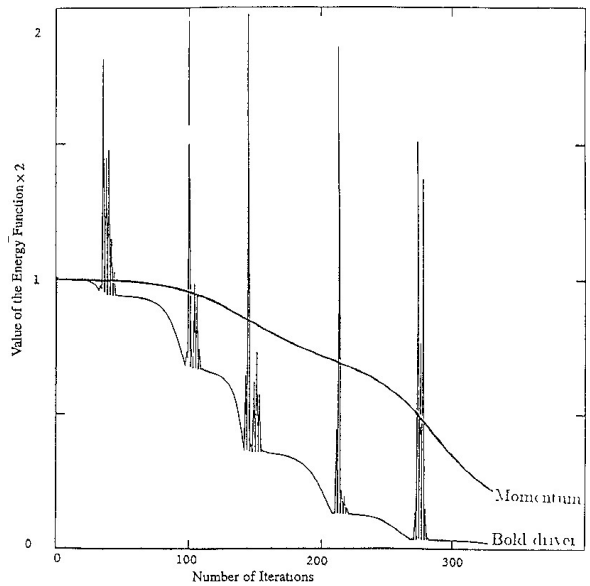


Рисунок 3.6 – Порівняння енергетичних значень для «Bold driver» та імпульсної версії навчання EBP.

Стратегія імпульсу регулює коефіцієнт ефективного навчання локально для кожної ваги через термін імпульсу. Стало відомо що p — 1,1 і cr = 0,5 є хорошим вибором. Оскільки цей метод контролює коливання значення функції енергії, його можна розглядати як алгоритм EBP з контрольованим коливанням функції енергії.

Самоадаптивне зворотне поширення. Цей метод був розроблений незалежно Джейкобсом і Девосом і Орбаном. Це локальне прискорення стратегії, і це підтверджується наступними спостереженнями: (1) кожна вага повинна мати свій власний коефіцієнт швидкості навчання, тому що часткова похідна енергетичної функції А щодо кожної ваги дає градієнт лише для цієї ваги. Крім того, коефіцієнт швидкості навчання для однієї ваги не обов'язково повинен бути відповідним для інших ваг. (2) Коефіцієнт швидкості навчання має змінюватися залежно від природи поверхні енергетичної функції вздовж розміру, що розглядається. (Незвично мати різні властивості градієнта вздовж одного виміру.) (3) Коефіцієнт швидкості навчання для ваги слід збільшити, якщо послідовні кроки мають однаковий знак. (4) Коефіцієнт швидкості навчання має бути малим, коли похідна енергетичної функції щодо ваги змінює знак.

Нехай коефіцієнт швидкості навчання для ваги m{; на кроці часу s бути q.

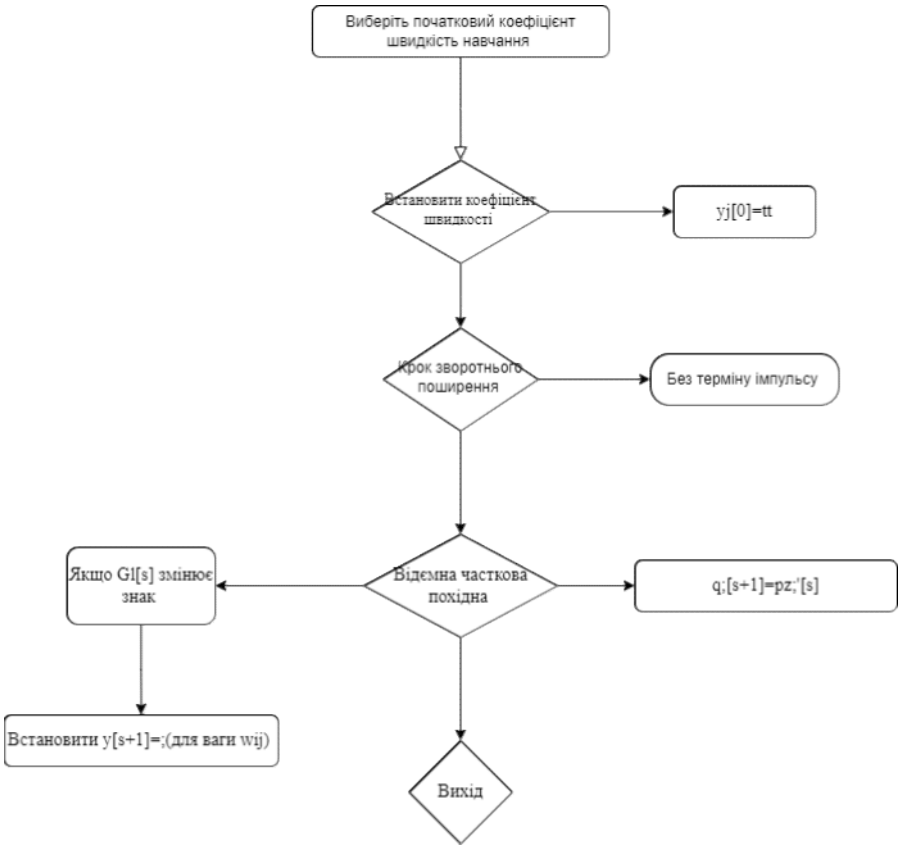


Рисунок 3.7 – Алгоритм підвищення швидкості навчання

«оцініть «хорошу» вагу мДж,[s + 1] шляхом інтерполяції (на основі попередніх значень AmJq і припускаючи, що простір ваг є квадратичним у кожному вимірі);
« виконайте кілька кроків зворотного розповсюдження rift momentum term.
Перезапустіть алгоритм з кроку.

Приклад (продовження). Метод SAB принципово відрізняється від усіх інших методи, розглянуті раніше. У цьому методі для кожної ваги зберігається один коефіцієнт швидкості навчання. На рис 3.10 зображено графіки

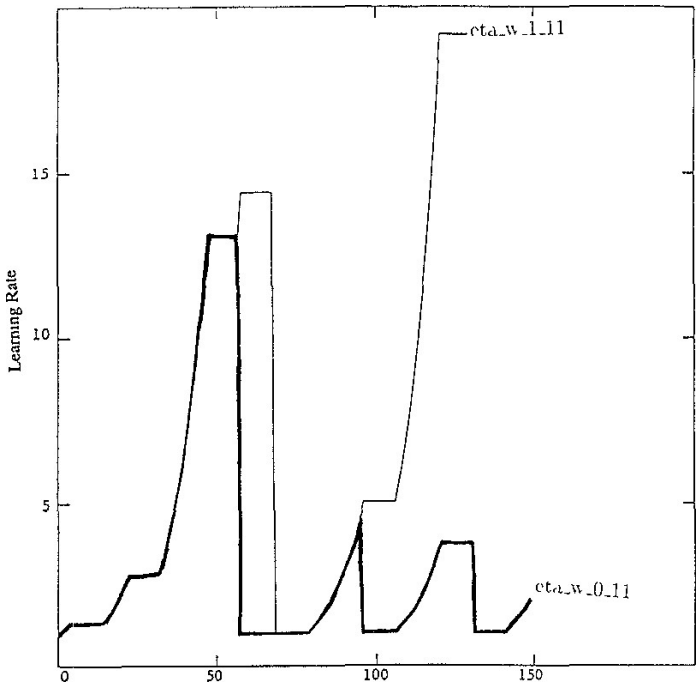


Рисунок 3.8 – Окремі коефіцієнти швидкості навчання для
дві ваги у версії SAB навчання EBP.

Коефіцієнти швидкості навчання для двох ваг прикладу мережі. Це показує, що вони починають з однакового значення коефіцієнта швидкості навчання і зберігають свої значення протягом деякого часу, збільшуючи їх експоненціально. Для цього прикладу ми використали коефіцієнт приросту p —— 1,1. Але після кількох ітерацій коефіцієнт швидкості навчання для m 0t падає до початкового значення, тоді як для in$_l$ t – ні. Це пояснюється тим, що градієнт енергетичної функції в напрямку w $^0{}_{11}$змінює знак, але для in*t не змінюється. Бажаний рівень помилки був досягнутий після 171 ітерації. Кінцеві ваги мережі відрізнялися від

отриманих іншими методами. Кінцеві ваги та зміщення мережі

Цей алгоритм працює краще, ніж оригінальний EBP, оскільки він може регулювати коефіцієнт швидкості навчання в широкому діапазоні, якщо початковий коефіцієнт швидкості навчання є «малим». Одна проблема, притаманна оригінальному алгоритму EBP, вибір «хорошого» коефіцієнта швидкості навчання користувачем, залишається незмінною в цьому алгоритмі. Крім того, цей алгоритм починається з початкового коефіцієнта швидкості навчання, якщо відбувається зміна знаку градієнта. Долає обидві проблеми. тут його називають SuperSAB.

SuperSAB нехай p буде коефіцієнтом, на який коефіцієнт швидкості навчання збільшується, якщо необхідно, і нехай cr буде коефіцієнтом, на який швидкість навчання зменшується, якщо необхідно. Експериментальні дослідження показують, що зниження коефіцієнта швидкості навчання має бути швидшим, ніж збільшення. Нагадаємо, що коефіцієнт швидкості навчання на кроці часу s для ваги позначається [s].

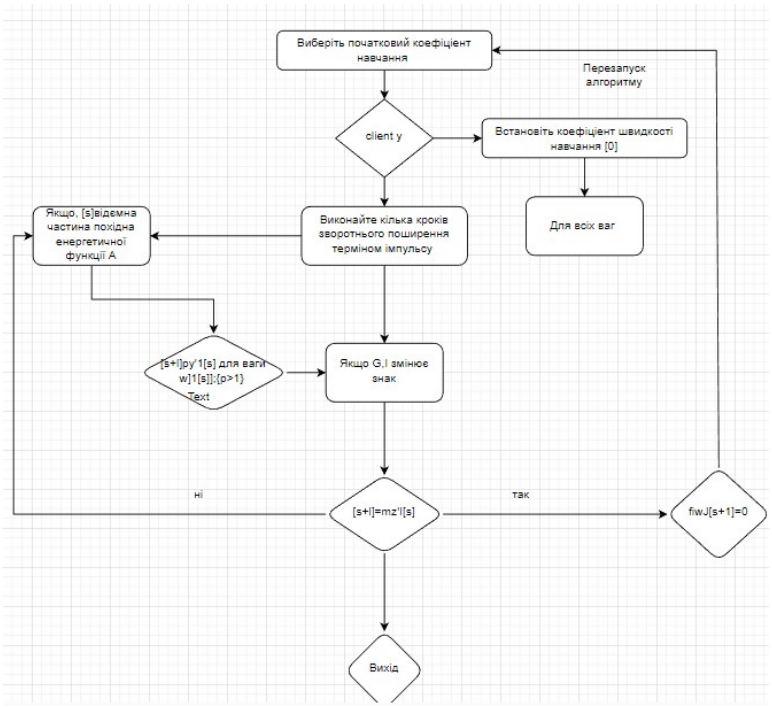


Рисунок 3.9 – Алгоритм зниження коефіцієнту швидкості навчання

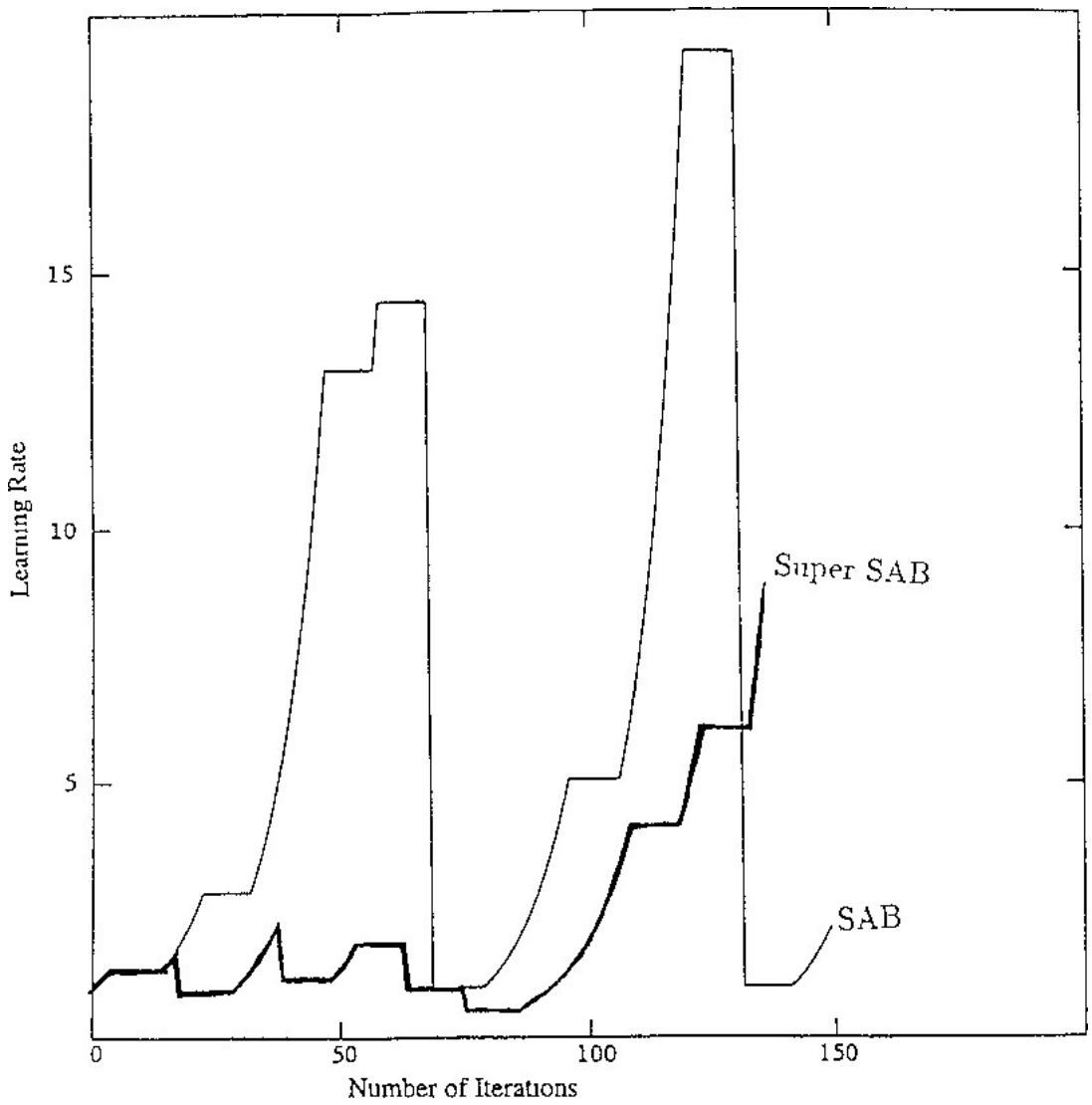


Рисунок 3.10 – Порівняння коефіцієнтів швидкості навчання для ваги у версіях навчання EBP SAB і SuperSAB.

Приклад (продовження). На малюнку 8 показано графіки швидкості навчання ваги in1 для алгоритмів навчання SAB і SuperSAB. Ці два графіки показують найважливішу різницю між методами навчання SAB і SuperSAB: коли відбувається зміна знаку градієнта енергетичної функції в напрямку ваги ’11›

(3.1) у методі навчання SAB коефіцієнт швидкості навчання встановлюється на початкове значення, але (2) у методі навчання SuperSAB значення коефіцієнта швидкості навчання зменшується на постійний коефіцієнт, який становить 0,5 у нашому прикладі.

Таблиця 3.3 Порівняння методів SAB та SuperSAB

Методи	Порівняння	Формули
Метод SAB	Локальне прискорення стратегії, і це підтверджується наступними спостереженнями: (1) кожна вага повинна мати свій власний коефіцієнт швидкості навчання, тому що часткова похідна енергетичної функції. Метод SAB принципово відрізняється від усіх інших методи, розглянуті раніше. У цьому методі для кожної ваги зберігається один коефіцієнт швидкості навчання. У методі навчання SAB коефіцієнт швидкості навчання встановлюється на початкове значення	$\Delta\omega_{ij}(t+1) = -\frac{n\partial E(t+1)}{\partial\omega_{ij}(t)} + \alpha\omega_{ij}(t)$
Метод SuperSAB	У методі навчання SuperSAB значення коефіцієнта швидкості навчання зменшується на постійний коефіцієнт. Стверджується, що в середньому метод SuperSAB швидший за метод SAB, у цьому прикладі нам непощастило: мережа не навчилася навіть після 500 ітерацій.	$\Delta\omega ij(t+1) = -\frac{n\partial E(t+1)}{\partial\omega ij(t)} + \alpha\omega ij(t)$

3.4.3 Зміна масштабу змінних

Модифікації, описані раніше, не враховують вплив кількості рівнів у мережі, хоча вважається, що зі збільшенням кількості рівнів зростає і час навчання в оригінальному алгоритмі EBP. Заснований на двох простих, але взаємопов'язаних фактах. По-перше, сигмоїдна функція див. Рівняння (3.3), коли вона використовується як функція активації в оригінальному алгоритмі EBP (функція активації), узгоджується з логістичним диференціальним рівнянням

$$y' = y(1 + y) \qquad (3.10)$$

Оскільки значення сигмоїдальної функції f(x) обмежене нулем і одиницею, значення у' —— у(1 — у) обмежене нулем і 0,25. Похідна у' досягає максимального значення 0,25, коли у = 0,5. По-друге, термін у(1 — у) походить від ланцюгового правила для обчислення часткових похідних енергетичної функції, заданої рівнянням (3.5) або (3.6). Кількість разів, коли цей термін зустрічається, рівно стільки шарів, на яких ваги віддалені від вихідного шару. Іншими словами, член у(1 — у) зустрічається рівно один раз у часткових похідних енергетичної функції щодо ваг вихідного шару, рівно двічі в частковій похідній енергетичної функції щодо ваг у прихованому шарі відразу за вихідним шаром і так далі.

Через ці два факти значення часткової похідної на різних шарах не може перевищувати 1/4, 1/16, 1/64, ..., і, отже, величина градієнта зменшується з експоненціальною швидкістю, коли ми віддаляємося від вихідного шару. Одним із способів вирішення цієї проблеми є зміна масштабу значення градієнта в кожному шарі. Ріглер та ін. }1991] пропонують використовувати степені очікуваного значення у(1 — у), А[у(1 — у)]', припускаючи, що значення у є однозначним. формально розподілені між нулем і одиницею. Ці значення становлять 1/6, 1/36, 1/216, а відповідні значення зміни масштабу — 6, 36, 216, ... для шарів один, два, три і так далі, коли шари підраховуються назад, починаючи з вихідний шар як перший. Про покращення ефективності навчання за допомогою

процедури змінного масштабу повідомлялося в Ріглер. Було виявлено, що в середньому це значно зменшує кількість необхідних ітерацій.

3.4.4 Самостійне визначення швидкості адаптивного навчання

Проблема з алгоритмом ЕВР полягає в тому, що для оновлення ваг необхідний коефіцієнт швидкості навчання. Два підходи, які обговорювалися досі, або використовують обґрунтоване припущення, щоб зафіксувати коефіцієнт швидкості навчання на постійному значенні, або коригують його динамічно за допомогою деяких евристичних методів. Але вони не можуть усунути ризик «занадто великих» кроків і перевищення мети. Метод, розглянутий у цьому розділі, обчислює локальний, найбільш екстремальний, найкрутіший градієнт, досяжний для обчислення оптимального розміру кроку. Як тільки помилка (E p) або поточна висота та екстремальний градієнт ваги енергії (екстремальний (6Ep/ 6 w [s j)) відомі, $\Delta_{\text{opt}} w_{ij}^{l}[s]$можна обчислити, як можна обчислити як

$$\Delta opt\omega_{ij}^{l}[s] = \frac{E}{extreme\ \dfrac{\partial E_p}{\partial \omega_{ij}^{l}[s]}} \tag{3.11}$$

Отже, використовуючи рівняння (3.20), оптимальний коефіцієнт швидкості навчання можна виразити через

$$copt = \frac{E}{extreme\dfrac{\partial E_p}{\partial \omega_{ij}^{l}[s]}\Big)\dfrac{\partial E_P}{\partial \omega_{ij}^{l}[s]}} \tag{3.12}$$

Методи обчислення енергетично-вагового градієнта та енергії відомі описано метод обчислення екстремального енергетично-вагового градієнта.

Екстремальний градієнт. Метод для обчислення екстремального градієнта енергетичної ваги, розглядаються нейрони у вихідному шарі та у прихованих шарах окремо. По-перше, описано обчислення екстремального енергетично-

вагового градієнта для одиниць вихідного шару. Виконуйте позначення, введені раніше ланцюгове правило диференціювання, градієнт енергетичної функції по відношенню до ваги мДж, для вхідного шаблону навчання рх можна записати як

$$\frac{\partial E_p}{\partial \omega_{ij}^{l-1}[s]} = \frac{\partial E_p \, \partial out_i^l \, \partial net_i^l}{\partial out_i^{l^-} \, \partial net_i^l \, \partial \omega_{ij}^{l-1}[s]} \tag{3.13}$$

Встановлюючи значення ou/ ' до можливого максимуму, що дорівнює одиниці, межі енергетично-вагового градієнта можна обчислити шляхом розв'язання кубічної in out* [отриманої з рівнянь (3.23), (3.24) і (3.25)] за інтервал (0, l). Нехай знайдені максимальне і мінімальне значення дорівнюють m і m2 відповідно. Визначимо b як

$$\delta_{pk}^0 = \frac{\partial E_p \, \partial out_k^0}{\partial out_k^0 \, \partial net_{pk}^0} \tag{3.14}$$

Приховані шари. Для нейрона u'I у прихованому шарі значення градієнта ваги енергії наступного зовнішнього шару / + l використовується для обчислення його екстремальних градієнтів ваги енергії:

$$\frac{\partial E_p}{b0 \; якщо \, t \; 1} = \sum_{i=1}^{n_l+1} \frac{\partial E_p}{d \; out,'} + \frac{\partial out^{l+1}}{3net''1} \frac{\partial out^{l+1}}{1} \tag{3.15}$$

Використовуючи рівняння (3.26) і зауважуючи, що це рівняння що представлене нижче.

$$\frac{d \; net,''''}{\partial out_j^l} = \omega_{ij}^l, \tag{3.16}$$

$$\frac{nEp}{d\ out\ j'} \sum_{l=1}^{n_l+1} \left(\delta_{pl}^{l+1} \omega_{ij}^{l}\right) \tag{3.17}$$

Тепер з останнього рівняння b^l для нейрона l^pу прихованому шарі може визначається як рівнянням представленим нижче.

Екстремальні ваги. Обчислення екстремального енергетично-вагового градієнта для вузлів у прихованих шарах вимагає значень екстремальних ваг. Тепер описано метод обчислення екстремального градієнта. Гранична вага для зв'язків між нейроном або w^l_{ij} s] або з нейроном1 u^{l+1} i8[s + 1]. У момент часу s значення першого відомого: однак обчислення значення *w* [s + 1] за допомогою рівнянь (3.6) і (3.8) вимагає значення y,p, і обчислюється крадіжка. Ця проблема круговості може уникати, виражаючи крайність градієнту з точки зору і обчислень.

Навчання зворотного поширення помилок безпосередньо. Обчислення можна продовжити після перегляду екстремального градієнта як

$$\text{extreme}\frac{\partial E_p}{\partial \omega_{ij}^{l}[s]}\Big) = \alpha n_{opt} + \ddot{0} \tag{3.18}$$

$$a = \frac{1^{n_l+1}}{4}, extreme(Op, !'fi) \tag{3.19}$$

де fi, = 0 або — якщо *Ep/ à w s*] залежно від того, екстремум (m , [s]) — m , [s] або mJ, [/ + 1] відповідно, і

$$b --\frac{1}{4}\sum_{i=1}^{nl+1} extreme\left(\delta_{pl}^{l+1}\omega_{ij}^{l}[s]\right) \tag{3.20}$$

Підстановка екстремального значення градієнта з рівняння (3.33) у рівняння (26) призводить до квадратного рівняння вигляду

$$\alpha n_{opt}^2 + bn_{opt} + c = 0 \tag{3.21}$$

Можете переглянути на формулу зображену нижче.

$$c = \frac{-E_p}{\partial E_p / \partial \omega_{ij}^{l}[s]} \tag{3.22}$$

Квадратне рівняння в sept має єдиний позитивний дійсний розв’язок. Таким чином, за допомогою техніки, описаної в цьому розділі, можна легко обчислити безпечний розмір кроку на кожній ітерації алгоритму EBP. Модифікація збільшує вимоги до обчислень. Крім того, оскільки він завжди має консервативні розміри кроку, він, швидше за все, буде слідкувати за енергетичною поверхнею дуже уважно, і, отже, може застрягти в локальних мінімумах, перш ніж вивчати всі приклади. Емпіричне дослідження показало деяку покращену продуктивність цього алгоритму порівняно з тими, які використовують термін імпульсу [Weir 19911. Найбільш привабливою особливістю цього модифікованого алгоритму є те, що для використання не потрібно «вгадувати» коефіцієнт швидкості навчання.

3.4.5 Методи спряженого градієнта

В алгоритмі EBP енергетична функція E мінімізується після найкрутішого спуску. На кожному кроці обчислюється напрямок, який мінімізує помилку

Де р транспонування вектора р;. Основна ідея полягає в наступному: якщо новий напрямок мінімізації, р, перпендикулярний попередньому напрямку р;,

мінімізація з попереднім напрямком не буде залежати від мінімізації в поточному напрямку.

Перш ніж продовжити далі, введемо деякі позначення, які використовуються лише в цьому розділі. Нехай ваги мережі на кроці часу s вважаються вектором W[s]. Нехай G[s] буде вектором градієнта енергетичної ваги, що відповідає елементам W[s] на кроці часу s. Нехай F[s] = G[s + 1] — G[s]; перший напрямок p[l] задається p[l] — — G[1]. Починаючи з початкової вагової матриці W[l] наступні вагові матриці обчислюються наступним чином:

$$IV[s+1] = W[s] + y[s\,p[s] \tag{3.23}$$

$$\emptyset[s] = \frac{Y[s].G[s+1]}{Y[s].p[s]} \tag{3.24}$$

Де A. B двох векторів однакових розмірів — скалярний добуток, визначений як сума добутку відповідних елементів. Формально,

$$A.B = \sum_{\cdot} a_l \times b_j \tag{3.25}$$

Існують різні версії методу спряженого градієнта в залежності від вибору обчислення 8 c].

Дві проблеми викликають занепокоєння щодо використання методів сполучених градієнтів для ЕВР. По-перше, обчислення 8[s] за допомогою рівняння (3.43) дороге, оскільки кожен крок лінійного пошуку оптимального значення включає оцінку енергетичної функції, а це вимагає обчислення похибки. По-друге, цей метод гарантував би збіжність після N + 1 ітерацій, якби цільова функція мала квадратичні ваги. Крім того, існує ймовірність чисельної нестабільності. Незважаючи на це, було показано, що метод спряженого градієнта

працює краще, ніж стандартний алгоритм EBP. Інші припустили та емпірично показали, що в деяких випадках можна використовувати метод спряженого градієнта з деяким неточним лінійним пошуком замість точного лінійного пошуку (у рівнянні (3.43) . Напрямок пошуку обчислювався за допомогою наступного рівняння:

$$p[s+1] = G[s+1] + A\ s\ MW[s] + f > [s]Y[s) \tag{3.26}$$

Де

$$AW[s] = W[s] - W[s-1] \tag{3.27}$$

Після кожних N кроків, де n – кількість ваг у ваговій матриці W[s], пошук перезапускається в напрямку від'ємного енергетично-вагового градієнта.

3.4.6 Різні енергетичні функції

Методи, описані в інших розділах, зберігають суму квадратів помилки як енергетичну функцію [див. Рівняння (3.4) і (3.5)], тоді як вони змінюють вихідний алгоритм EBP шляхом коригування коефіцієнта навчання, додавання члена імпульсу, використовуючи елегантне числове обчислення методи тощо. У цьому розділі описується набір нових енергетичних функцій, які визначаються та використовуються для прискорення навчання.

Вираз для обчислення сигналу помилки, коли використовується сигмоподібна активація [Рівняння (3.2)] і сума квадратів помилок, містить коефіцієнт форми out,°(1 — ont,'), whore значення out,' знаходиться між нулем і одиницею. Значення цього фактору наближається до нуля, як owl, наближається до одного зі своїх крайніх значень, нуля або одиниці. Таким чином, якщо вихідний сигнал вихідного нейрона близький до будь-якого екстремуму і далекий від цільового вихідного сигналу, то високе значення його сигналу помилки послаблюється дуже близько до нуля, і майже не вноситься корекція його ваг. З

цим спостереженням ван Оойен і Ніенхейс [1992] запропонували нову енергетичну функцію

Містить фактори, який послаблює сигнал помилки, коли вихід нейрона у вихідному шарі близький до будь-якої крайності. Емпіричне дослідження кількох проблем показало покращену продуктивність. Ця логарифмічна функція правдоподібності також була запропонована Софією та ін. [l9h8] використовувати як міру похибок у багатошарових нейронних мережах, і за допомогою чисельного моделювання вони виявили помітне скорочення часу навчання. Це вдосконалення розглядалося як характеристика функції. Холт і Семнані також рекомендували цю енергетичну функцію і спостерігали за допомогою моделювання, що вона може прискорити навчання зворотного поширення.

Оскільки стандартний алгоритм EBP спускається вздовж градієнта поверхні помилки, використання будь-якої енергетичної функції, яка має більший градієнт, ніж градієнт суми квадратів помилки при вищих значеннях енергії, сприятиме швидшому навчанню. Ця точка зору, можливо, спонукала Ахмада та Далама запропонувати та використати енергетичну функцію Коші та поліноміальну енергетичну функцію. Енергетична функція Коші E, визначена рівнянням (3.28), продемонструвала більш швидку збіжність:

$$E_c = \frac{1}{2}\prod_{p--1}^{P}\prod_{i-1}^{n_0}\left(1+\left(\varepsilon_{pl}\right)^2\right) \tag{3.28}$$

Поліноміальна функція енергії, задана як

$$E_{pol} = a\sum_{p=1}\sum_{i=1}\left(\left|\varepsilon_{pl}\right|+b\right)^r \tag{3.29}$$

Де o і ò є константами (або будь-якими іншими енергетичними функціями, які допомагають прискорити процес навчання). Повідомлялося, що більш швидка

збіжність досягається за допомогою поліноміальної функції енергії. Третя функція енергії, використана Ахмадом і Саламом, називається експоненціальною функцією енергії, з’являється в практичних ситуаціях, щоб допомогти уникнути локальних мінімумів на вищих рівнях енергії. Він визначається в термінах суми квадратів функції енергії помилки E as

3.4.7 Вплив розміру навчальної групи

Оригінальний EBP і модифікації, які тут обговорюються, не враховують ефект розміру навчального набору. Емпіричне дослідження показало залежність від розміру навчального набору зосереджено на складній проблемі, функції парності, яка вимагає запам'ятовування вхідних шаблонів. Якщо мережа не надто мала для запам'ятовування, то час навчання експоненціально зростає з кількістю бітів, для яких потрібно обчислити парність. Були обмежені складною проблемою, їхні результати можуть не застосовуватися до таких проблем, як класифікація шаблонів і регресія.

Що довжина енергетично-вагового градієнта Cry може бути пропорційна L,

$$L - \sqrt{N_1^2 + N_2^2 + \ldots + N_m^2} \tag{3.30}$$

Де N — кількість шаблонів типу i, а m — кількість різних типів шаблонів. Інтуїція, яка лежить в основі їх припущення щодо довжини градієнта, полягає в тому, що «розмірність вагового простору зазвичай більша, ніж кількість унікальних типів шаблонів, тому градієнт кожного типу шаблону може бути ортогональним до всіх інші типи». Щоб компенсувати цей ефект, вони запропонували коефіцієнт швидкості навчання, обернено пропорційний L. Їхній експеримент із декількома проблемами припускає, що якщо вибрано за допомогою наступного рівняння та коефіцієнт імпульсу a = 0,9, то досягається швидке навчання:

$$n = \frac{1.5}{\sqrt{N_1^2 + N_2^2 + \ldots + N_m^2}} \quad (3.31)$$

Був запропонований метод прискорення навчання класифікації, коли розміри навчальних наборів для кожного класу різні. Вони показали, що в задачах двох класів, якщо розмір одного навчального набору набагато менший, ніж розмір іншого, перша ітерація вихідного алгоритму EBP збільшує похибку для класу з меншою кількістю прикладів, одночасно зменшуючи похибку для іншого. Це в середньому збільшує кількість ітерацій, необхідних для вивчення класифікації. Вони запропонували метод, який зменшує помилку для обох класів одночасно. Вони досягають цього шляхом вибору напрямку корекції ваги, який сприяє навчанню обох класів. Дуже простий спосіб знайти такий напрямок — вибрати напрямок, який ділить навпіл напрямки негативних градієнтів кожного класу .

У наступному розділі порівнюються та порівнюються модифікації алгоритму зворотного поширення, представлені в цьому розділі.

Підведення підсумків вищесказаного в розділі.

Модифікації вихідного алгоритму EBP, описані в попередньому розділі, можна класифікувати різними способами. Одна класифікація може базуватися на тому, чи модифікація застосовується до онлайн-алгоритму EBP, до пакетних алгоритмів EBP або до обох. Стратегія імпульсу, масштабування змінних, EBP із очікуваними вихідними значеннями та різні енергетичні функції можуть використовуватися як для онлайнового, так і для пакетного алгоритмів EBP. Самостійне визначення швидкості адаптивного навчання може бути застосовано лише до онлайн-алгоритму EBP. Усі інші методи, описані тут, корисні лише для пакетних алгоритмів EBP.

Інша класифікація може ґрунтуватися на тому, чи існує лише один (ефективний) коефіцієнт рівня заробітку для всіх ваг чи один коефіцієнт рівня навчання для кожної ваги. Метод «Blod driver», перемасштабування змінних, самовизначення адаптивної швидкості навчання, метод спряженого градієнта та

ефект навчання можуть використовувати лише один коефіцієнт швидкості навчання для всіх ваг. Усі інші методи можуть використовувати один коефіцієнт швидкості навчання для кожної ваги.

Деякі з цих методів можна комбінувати, щоб отримати швидше навчання, ніж це можливо отримані з будь-якого окремого. Наприклад, і SAB, і SuperSAB використовують стратегію імпульсу. Вважається, що стратегію імпульсу , керування коливанням ваг, перемасштабування змінних, EBP із очікуваним вихідним значенням!„ та вплив розміру навчального набору можна поєднати разом для швидшої конвергенції.

Проблема самостійного визначення швидкості адаптивного навчання полягає в тому, що, роблячи консервативні кроки, щоб уникнути перевищення мети, це може призвести до того, що мережа застрягне в локальному мінімумі до вивчення всіх прикладів навчального набору. Будь-яка модифікація, окрім самостійного визначення швидкості адаптивного навчання, яка застосовується до онлайн-алгоритму EBP, потребує «хорошого» коефіцієнта швидкості навчання, який надає користувач, ефективний коефіцієнт швидкості навчання для двох задач може бути досить різним.

Таким чином, необхідно знайти модифікацію алгоритму EBP для онлайн-навчання, яка не потребує ефективного коефіцієнта швидкості навчання.

3.3 Оцінка навчання

Нейронні мережі є однією з найкрасивіших парадигм програмування, коли-небудь винайдених. В звичайний підхід до програмування, ми кажемо комп’ютеру, що робити, на ура проблеми перетворюються на безліч невеликих, точно визначених завдань, які комп’ютер може легко виконати. Навпаки, у нейронній мережі ми не говоримо комп’ютеру, як вирішити нашу проблему. Натомість він вчиться на даних спостережень, знаходячи власне рішення проблеми.

Отже, в попередньому розділі ми розібрались чому підходить саме змішаний метод навчання

Процедура навчання MLP виглядає наступним чином:

- Починаючи з вхідного рівня, передайте дані на вихідний рівень. Цей крок є прямим поширенням.
- На основі результату обчисліть похибку (різницю між прогнозованим і відомим результатом). Помилку потрібно звести до мінімуму.
- Зворотне поширення помилки. Знайдіть його похідну відносно кожної ваги в мережі та оновіть модель.
- Повторіть три кроки, наведені вище, протягом кількох епох, щоб дізнатися ідеальну вагу.

Нарешті, вихідні дані беруться за допомогою порогової функції для отримання прогнозованих міток класу.

Зворотне розповсюдження помилок (EBP) зараз є найбільш використовуваним алгоритмом навчання для штучних нейронних мереж прямого зв’язку (FFANN). Однак зазвичай вважається, що він дуже повільний, якщо він збігається, особливо якщо розмір мережі не надто великий у порівнянні з поточною проблемою. Основна проблема з алгоритмом EBP полягає в тому, що він має постійний коефіцієнт швидкості навчання, а різні області поверхні помилок можуть мати різні градієнти характеристик, які можуть вимагати динамічної зміни коефіцієнта швидкості навчання на основі природи поверхні. Крім того, характеристика поверхні помилок може бути унікальною в кожному вимірі, що може вимагати одного коефіцієнта швидкості навчання для кожної ваги. Щоб подолати ці проблеми, було запропоновано кілька модифікацій. Це опитування є спробою представити їх разом і порівняти. Першою модифікацією була стратегія імпульсу, де частка останньої корекції ваги додається до поточної запропонованої корекції ваги. Він має як прискорюючий, так і сповільнюючий ефект, де це необхідно. Однак цей метод може дати лише відносно малий динамічний діапазон для коефіцієнта швидкості навчання. Для збільшення динамічного діапазону коефіцієнта швидкості навчання запропоновано такі

методи як «Bold driver» і SAB (самоадаптивне зворотне поширення). Модифікація SAB, яка усуває вимогу вибору «хорошого» коефіцієнта швидкості навчання користувачем, дала SuperSAB. Невелика модифікація стратегії імпульсу створила новий метод, який контролює коливання ваг для прискорення навчання. Модифікація алгоритму EBP, у якій градієнти змінюються на кожному шарі, допомогла підвищити продуктивність. Використання «очікуваного результату» нейрона замість фактичного результату для коригування ваг покращило продуктивність стратегії імпульсу. Метод сполученого градієнта та «самовизначення адаптивної швидкості навчання» не вимагають від користувача коефіцієнта швидкості навчання. Використання енергетичних функцій, відмінних від суми квадратів помилки, продемонструвало кращу швидкість збіжності. Вибір ефективного коефіцієнта швидкості навчання повинен враховувати розмір навчального набору. Усі ці методи покращення продуктивності алгоритму EBP представлені тут.

Таблиця 3.4 Підсумок досліджень методів вдосконалення

Методи вдосконалення	Короткий опис
Модифікація алгоритму EBP	Градієнти змінюються на кожному шарі, допомогла підвищити продуктивність. Використання «очікуваного результату» нейрона замість фактичного результату для коригування ваг покращило продуктивність стратегії імпульсу.
Алгоримт EBP	Зараз є найбільш використовуваним алгоритмом навчання для штучних нейронних мереж прямого зв’язку (FFANN)
Метод сполученого градієнта	Не вимагають від користувача коефіцієнта швидкості навчання. Використання енергетичних функцій, відмінних від суми

	квадратів помилки, продемонструвало кращу швидкість збіжності. Вибір ефективного коефіцієнта швидкості навчання повинен

Продовження таблиці 3.4 Підсумок досліджень методів вдосконалення

Методи вдосконалення	Короткий опис
	враховувати розмір навчального набору.
Метод «Bold driver»	Для збільшення динамічного діапазону коефіцієнта швидкості навчання запропоновано такий метод.
Метод SuperSAB	Невелика модифікація стратегії імпульсу створила новий метод, який контролює коливання ваг для прискорення навчання.

4 ВДОСКОНАЛЕННЯ САМОАДАПТИВНОГО ЗВОРОТНЬОГО ПОШИРЕННЯ

Для досягнення кращої продуктивності системам вкрай важливо адаптувати активну поведінку з ключовим моментом пошуку медичних препаратів. Однак через відмінні індивідуальності поведінки користувача шаблони відрізняються один від одного. Тому ціль цієї роботи вдосконалити ефективніший спосіб отримання потрібної інформації.

У цьому розділі представлено інтелектуальне проміжне програмне забезпечення для вдосконалення пошуку та розширений механізм навчання SABPA (Self-adaptive BP Algorithm) та інших методів які будуть представлені нижче.

4.1 Проміжне програмне забезпечення для навчання

Ми пропонуємо нове проміжне програмне забезпечення, яке використовує механізм навчання для надання корисного контексту в системі усвідомлення контексту. Як показано на рис. 4.1, ми загалом поділяємо наше проміжне програмне забезпечення на три рівні, а саме рівень джерела контексту, рівень обробки контексту та рівень доставки контексту. Системний кеш. Дозволяє системі розпізнавання контексту реагувати швидше. Користувачі не змінили б своїх звичок раптово; збереження останнього контексту введення та його опцій значно покращить час відповіді. Ці записані параметри можуть бути надані швидко без обчислення, коли користувачі повторюють свої запити. Крім того,

Кеш зберігає параметри навчальної мережі для різних користувачів (користувачі використовують навчальні мережі відповідно, тому що кожен користувач індивідуальний), що робить можливим підтримку кількох користувачів і відповідь за короткий час. Навчальну мережу, фоновий процес,

який вивчає шаблони користувачів, потрібно навчити перед використанням.

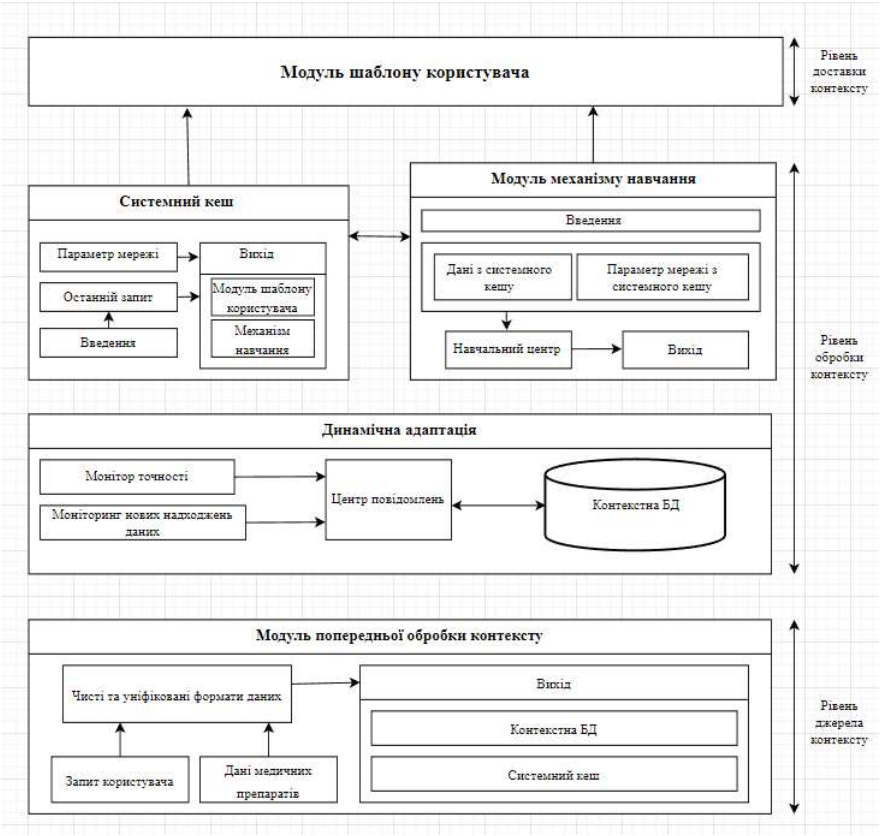


Рисунок 4.1 – Проміжне програмне забезпечення для механізму навчання

Модуль шаблону користувача. Зберігає останній результат запиту. Коли останній запит знайде відповідність у системному кеші, відповідну відповідь можна надати безпосередньо звідси. Механізм навчання — це алгоритм, обраний для навчання. У нашому дослідженні представлено вдосконалений алгоритм BP SABPA, який буде обговорено в наступному розділі

4.2 Самоадаптивний алгоритм BP

Мотивацією використання алгоритму BP є його здатність вивчати зв’язки в складних наборах даних, які не можуть бути легко сприйняті людьми. Він має можливість модифікувати вихідні дані відповідно до зміни контексту користувача. Це не вимагає перебудови всієї мережі, коли з’являється новий контекст.

Однак стандартний алгоритм BP має два недоліки. По-перше, швидкість навчання повільна. Традиційна мережа BP не може забезпечити швидку відповідь. По-друге, під час навчання виникають коливання. Ці проблеми зумовлюють його низьку ефективність. Ми маємо вдосконалити оригінальний алгоритм BP, щоб він міг бути придатним для запропонованого нами проміжного ПЗ. Формула для ваги оновлення в BP є

$$\Delta\omega ij(t+1) = -\frac{n\partial E(t+1)}{\partial\omega ij(t)} + \alpha\omega ij(t) \tag{4.1}$$

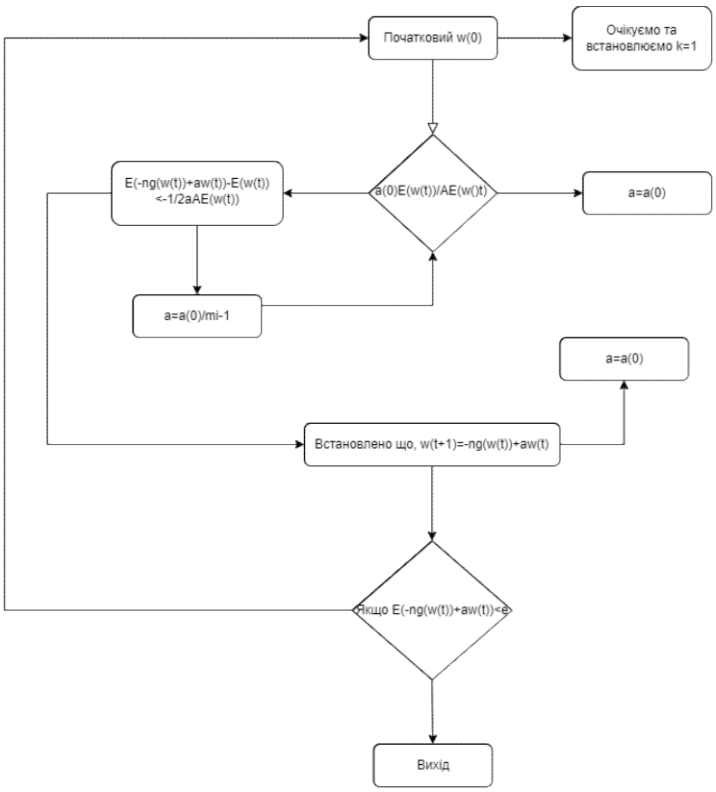


Рисунок 4.2 – Алгоритм SABPA

Де постійна швидкість навчання, яка зазвичай вибирається 0<<1, щоб гарантувати, що простір ваг не перевищить мінімум простору помилок. є імпульсом, який суттєво впливає на швидкість навчання.

E — це міра пакетної помилки, визначена як функція помилки суми квадратів різниць для всього навчального набору

$$E = \sum_{k-1}^{m} E_k = \sum_{k}^{m} \sum_{t}^{q} \left(y^k - c^k\right)^2 / 2 \tag{4.2}$$

Крім того, g( ) визначає градієнт ∇E(ω) функції похибок E суми квадратів різниць при .

Великі значення імпульсу можуть прискорити процес навчання, але це недолік результатом є коливання під час навчання. З іншого боку, менші значення імпульсу сповільнюють процес навчання. Зазвичай імпульс фіксується перед тренуванням. Тут ми пропонуємо стратегію динамічного коригування імпульсу, яка називається SABP (самоадаптивний алгоритм BP). Ця стратегія робить мережу BP швидшою та дозволяє уникнути коливань. Основна ідея полягає в тому, щоб перевірити E(t + 1) кожен раз. Якщо E(t + 1) > E(t) , яка означає, що навчання йде надто повільно, ми повинні прискорити процес навчання, збільшуючи . Якщо E(t + 1) < E(t) , що означає надто швидке навчання, можуть виникнути коливання, ми повинні скоротити процес навчання, зменшивши . Пропонований алгоритм SABPA проілюстровано в таблиці 1.

$$\begin{aligned} E\left(-ng(\omega(t)) + \alpha\omega(t)\right) - E(\omega(t)) \\ \leq -\frac{1}{2}\alpha\left\|\nabla E(\omega(t)\right\|^2 \end{aligned} \tag{4.3}$$

Це означає, що навчання йде трохи швидше, вигідно уповільнити процес навчання. Через $\frac{\mathrm{E(t+1)}}{\mathrm{E(t)}} > 1$ тоді α (t + 1) < α (t) . Процес навчання гальмується.

Коли E(t + 1) > E(t) , це означає, що навчання надто повільне. Необхідно підвищити швидкість навчання. Через $\frac{1}{\mathrm{mk}-1} > 1$ тоді α (t + 1) > α (t) процес навчання

Завдяки щоразу динамічно регулюючи імпульс, SABPA уникає коливань під час навчання, а також прискорює процес навчання.

4.3 Навчальні кроки

Через різноманітність необроблених даних перед навчанням потрібно виконати певну підготовчу роботу, яка полягає в зборі та очищенні контексту кожного користувача та збереженні їх у контекстній БД для навчання мережі ВР. Необроблені дані спочатку надсилаються до моделі попередньої обробки контексту, яка очищає дані та генерує їх до відповідного формату або вищого рівня, щоб зробити їх доступними для запропонованого механізму навчання. Попередньо оброблені дані будуть надіслані до DB, де зберігаються всі необхідні дані для навчання та оновлення наших мережі. Відповідно до іншої ситуації буде побудована різноманітна мережа ВР (різниця в кількості прихованих шарів і вузлів кожного шару). Навчання мережі ВР для кожного користувача має тривати до досягнення бажаного рівня помилок. При цьому параметри мережі ВР для кожного користувача зберігаються в системному кеші відповідно.

Коли користувач вводить запит або датчики виявляють зміну середовища, контекст надсилається до модуля обробки контексту, де контексти перетворюються на нормалізовані форми, а деякий контекст агрегується на вищому рівні. Основні контексти вибираються для збереження в контекстній БД і надсилаються в системний кеш. Якщо контекстна інформація вже існує в системному кеші, вихідні дані можуть бути надані безпосередньо системним кешем, оскільки він зберігає останні запити та відповіді. В іншому випадку кеш вибирає параметри мережі ВР відповідно до конкретного користувача та надсилає в мережу ВР модель механізму навчання. У моделі механізму навчання мережа ВР вивчає ці контексти, класифікує їх за вже відомою класифікацією, щоб отримати шаблон користувача, і надсилає результати як у вихідний, так і в системний кеш для повторних запитів. На малюнку 3 показано етапи навчання.

Модуль динамічної адаптації відповідає за оновлення мережі SABPA. Коли точність мережі SABPA знижується або новий контекст займає певний відсоток від загального контексту, цей модуль надсилає повідомлення як до БД контексту,

так і до моделі механізму навчання, щоб керувати ними, переходить до навчання мережі SABPA, доки вона не відповідатиме очікуваній частоті помилок. Таким чином, точність гарантована.

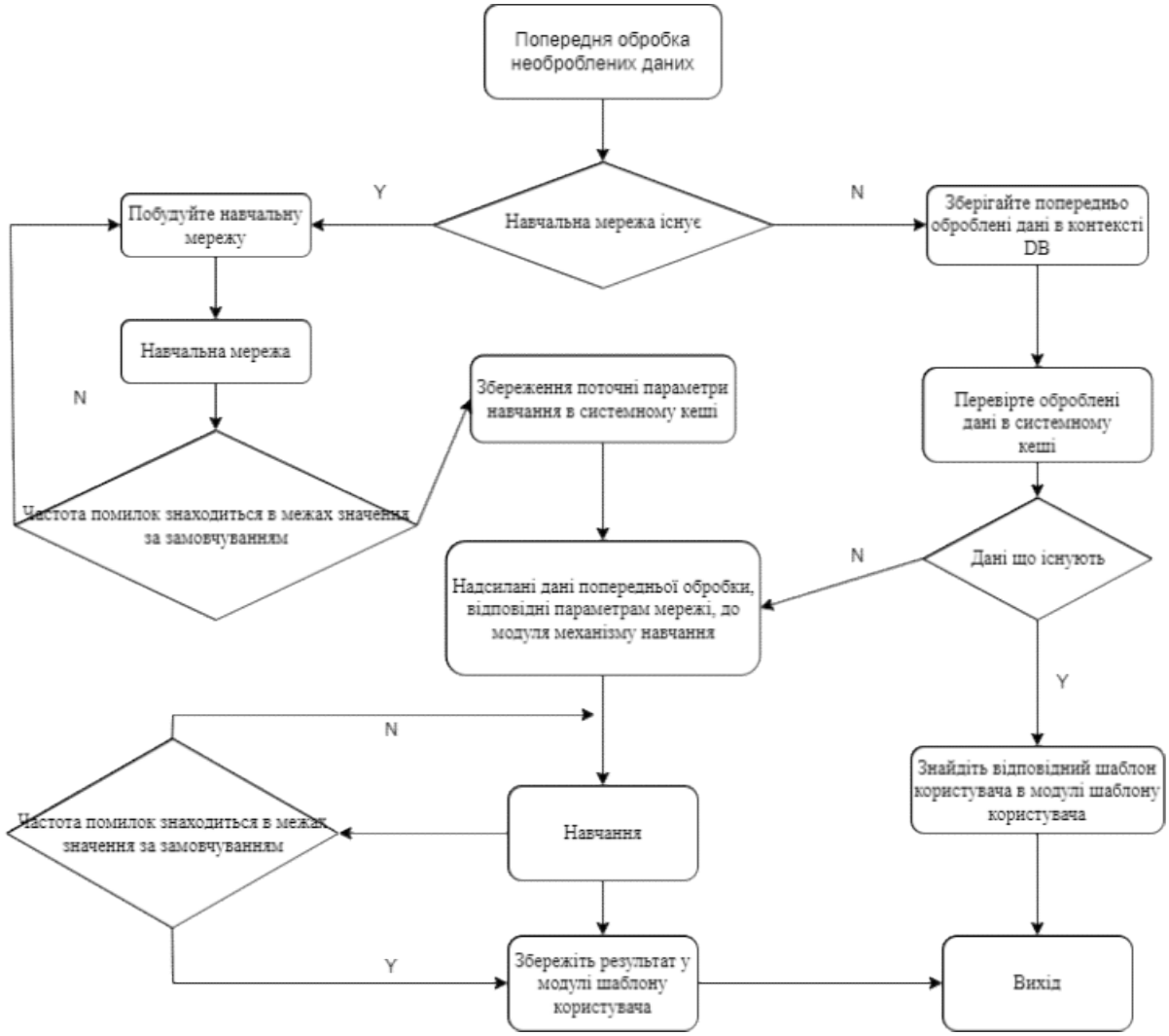


Рисунок 4.3 – Етапи навчання запропонованого проміжного ПЗ

У цьому розділі ми досліджуємо проміжне програмне забезпечення для інтелектуального навчання систем усвідомлення контексту та використовуємо SABPA (самоадаптивний алгоритм ВР) як механізм навчання для контекстних рекомендацій у повсюдному обчислювальному середовищі. На відміну від стандартного ВР, ми покращили його, щоб стати придатним для запропонованого проміжного програмного забезпечення, незалежно від його повільності. Використовуючи це проміжне програмне забезпечення в нашому сценарії,

системи усвідомлення контексту, очевидно, зменшують робоче навантаження та забезпечують кращий сервіс. За допомогою тестів ми довели, що наше проміжне програмне забезпечення та SABPA працюють далеко за межі прийнятного.

Майбутня робота в цьому полягає в тому, як вибрати основний контекст для навчання. Навчання мережі BP — це обчислення з витратами.

4.4 Реалізація самоадаптивного зворотнього поширення

В цьому розділі представлено обслуговування та позиціювання різних показників пошуку медичних препаратів які застосовуються в навчанні нейронної мережі. Використаний підхід SuperSAB Цей підхід включено в алгоритм навчання Левенберга-Марквардта, тому також представлено його нову версію, що призводить до самоадаптивного динамічного алгоритму навчання. Цей динамізм позитивно впливає не лише на точність отриманої моделі, а й на сам процес навчання. Описані комплексні тести алгоритму довели, що запропонований новий алгоритм динамічно інтегрує два великі світи статистики та теорії інформації,

В огляді пов’язаних історичних наукових і найсучасніших результатів досліджень, наведених у попередніх розділах, SuperSAB та інші вимірювання теорії інформації були проаналізовані на основі їх впливу на алгоритми навчання та поведінку отриманих моделей. Трьома ключовими та найбільш часто застосовуваними аспектами оцінювання є точність, швидкість і стабільність/надійність, тому ці точки зору застосовуються для позиціонування різних вимірювань помилок навчання. У цьому розділі структуровано описані переваги (або аналогічні) позиції вищезгаданих заходів.

4.4.1 Перевага показників помилок навчання

Застосований захід моделювання також має значний вплив на швидкість навчання, отже, різні показники також потрібно порівнювати за цим критерієм.

4.4.2 Перевага на основі стабільності

Методи, запроваджені для підвищення швидкості навчання, природно, включають вищий ризик розбіжностей, отже, вплив на стабільність під час навчання з використанням різних заходів також має бути проаналізовано та порівняно. Це пов’язано з надійністю алгоритму навчання проти багатьох інших факторів, таких як початкові значення ваги, незбалансовані тренувальні дані, викиди, шум тощо. Отже, стабільність і надійність розглядаються разом, цікаво, що лише менший коефіцієнт представлений такий аналіз. Тим не менш, можлива спеціальна тема майбутнього дослідження для аналізу впливу на розділену стабільність і надійність на набагато глибшому рівні.

4.5 Умови тестування запропонованого алгоритму

Визначивши кожен крок нового алгоритму, необхідно виконати його комплексне тестування, як описано в наступних розділах. Новий алгоритм реалізовано на Python використано Google Colab Google (2022). Під час тестування частково використовувався Google Colab.

«Імпульс» На рис 4.3 показано типовий результуючий характер кривих τ під час навчання за новим алгоритмом на тому ж наборі даних. Це означає, що незалежно від початкового значення кінцеві значення τ-s (майже) однакові в кінці процесів навчання на тому самому наборі даних (наприклад, близько 0,2 на наборі даних Iris).

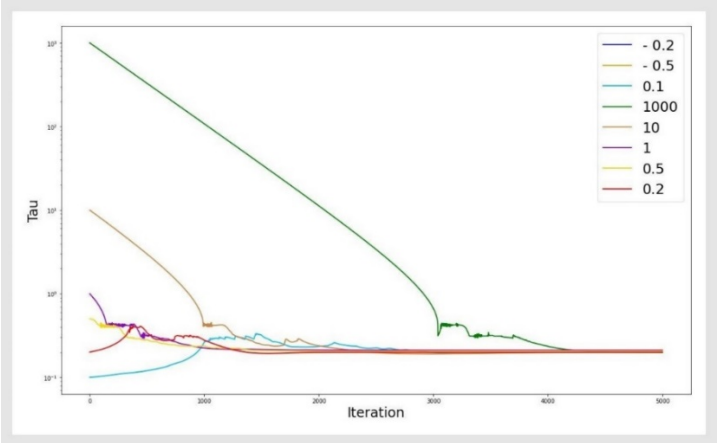


Рисунок 4.4 – Зміна значень τ від різних початкових значень під час

навчання нейронної мережі.

На цій діаграмі вертикальна вісь масштабована як логарифмічна.

На рис 4.3 показано, що довгі логарифмічні (лінійні на цій логарифмічній шкалі) ділянки задані на початку тау-оптимізації, і вона довша у випадку вищих τ-s. Це типовий випадок для введення розширення «Momentum» Rumelhart et al. (1986) як:

$$\tau_{k+1} = \tau_k + \alpha\Delta\tau^{previous} \tag{4.4}$$

«SuperSAB» — це вже традиційна техніка прискорення нейронної мережі, яку можна використовувати в будь-якій техніці, заснованій на grandient. Основна ідея складається з трьох компонентів, як це було введено Tollenaere (1990):

- Впорядкування параметра, який називається η, як множника до похідної ваг моделі, отже, величина кроку навчання залежить від добутку множника та самої похідної.
- Індивідуальний η застосовується для всіх ваг. Це означає, що кожен параметр тренування має свій множник.
- Значення цього нового множника η динамічно адаптується на основ співвідношення між напрямком попереднього кроку навчання та напрямком похідної. Якщо ці напрямки однакові (це означає, що під час навчання кроки виконуються безперервно в одному напрямку), множник η трохи збільшується, але коли ці напрямки стають протилежними (це означає, що навчання змінює свій напрямок), тоді множник η значно зменшується. Співвідношення невеликого збільшення, але значного зменшення є важливим у цій техніці прискорення. Таким чином, коли тренування реалізує безперервне оновлення ваги в тому самому напрямку (безперервне збільшення або зменшення ваги), множення на η збільшує розмір кроку, який було обчислено на основі чистої похідної. Коли напрямки оновлень ваги коливаються, то через зменшення множника η навчання виконує маленькі, обережні кроки поблизу фактичної точки навчання.

Подробиці цього трюку прискорення описано в Tollenaere (1990): як основний ефект, він може збільшити швидкість тренування принаймні на одну або дві величини, ніж чисте рішення «Імпульс».

4.6 Нова комбінація Momentum і SuperSAB

Алгоритм Левенберга-Маркварда (LM) — це метод мінімізації нелінійної функції, який досягає синтезу переваг інших алгоритмів оптимізації (таких як Найкрутіший спуск, Ньютон, Гаусс-Ньютон). Отже, можна сказати, що LM є швидким і стабільним алгоритмом порівняно з іншими згаданими. LM обробляє як два основні елементи матрицю Якобі та вектор похідної застосованої міри помилки, як описано в наступних двох підрозділах.

Формула цього алгоритму представлена нижче:

$$\beta \in argmin_{\beta} \equiv argmin_{\beta} \sum_{i=1}^{m} [y_i - f(x_i, \beta)]^2 \tag{4.5}$$

Враховуючи ці дві методики прискорення оновлення, не так просто включити їх у навчання Левенберга-Марквардта (LM), оскільки сам алгоритм навчання LM вже є рішенням прискорення (він об’єднує градієнт і Гаусса-Ньютона динамічно) і існує значний ризик того, що інтегровані рішення гасять ефекти прискорення одне одного.

Як згадувалося раніше, прискорення «Імпульс» застосовується до параметра τ і в той же час також до методу «SuperSAB». Звичайно, ці методи позитивно впливають не лише на «прискорення», але й на конвергенцію, стабільність тощо, але для простого посилання лише «прискорення» згадується під час застосування цих додаткових методів для модифікацій τ.

У підсумковому алгоритмі застосовано такий порядок прийомів:

– Левенберга-Марквардта (LM) для всіх ваг мережі, а також для τ.

– Прискорення «SuperSAB» тільки для τ.

– Прискорення «імпульсу» лише для τ.

На рис 4.5 представлені різні методи прискорення на тій же структурі мережі з різними початковими параметрами τ. Можна побачити, що різні випадки навчання призвели до подібних характеристик кривої навчання, але значно коротших (швидших) у разі меншого початкового τ. Однак метод з комбінованими техніками (LM і «Імпульс» і «SuperSAB») призвів до більш стабільного та набагато швидшого процесу тренування з меншими коливаннями, ніж версії лише з «Імпульс».

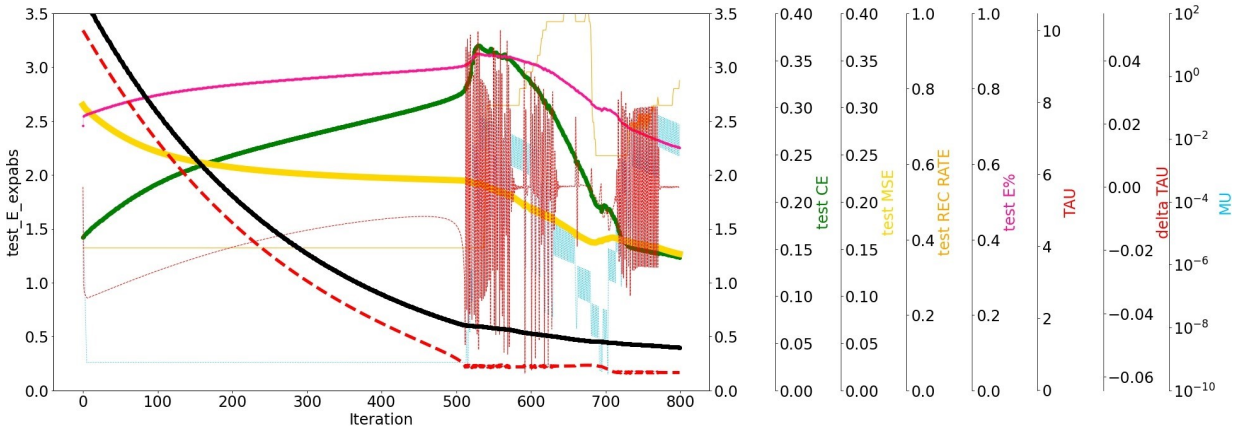


Рисунок 4.5 – Графік представляє процес навчання на тій же структурі мережі з методом «Імпульс» з різними початковими значеннями τ (10, 3, 1).

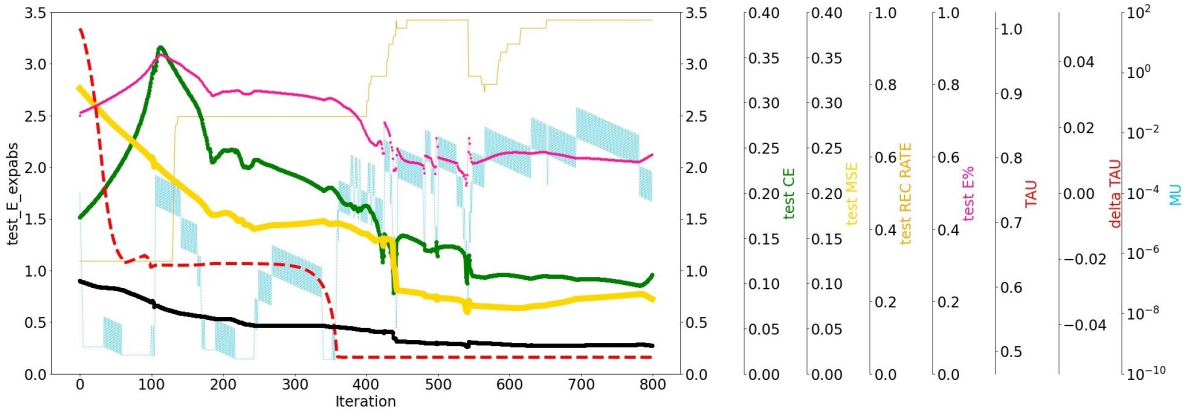


Рисунок 4.6 – Останній графік унизу відображає алгоритм, прискорений методом «SuperSAB» з тим самим початковим τ1, що й на останньому «Momentum»

4.6 Параметри тестування вдосконалення навколо нейронної мережі

Алгоритм тестувався для більшої кількості вихідних значень τ: 0,05, 0,1, 0,2, 0,5, 1, 2, 3, 4, 5, 10, 50, 100, 300, 1000, 5000, 10000. Частину цих значень досліджено в дослідження Silva та ін. (2008), але для доведення переваги запропонованого методу було обрано більше інших значень. Невеликі значення (0,05, 0,1, 0,2) представляють здебільшого напрям підготовки CE, але значення вище 100 вважаються майже лише MSE. Усі початкові значення між цими діапазонами починають навчання із застосуванням змішаного показника CE-MSE. Додатковий параметр α методу «Імпульс» був обраний рівним 0,1. Параметри прискорення «SuperSAB» були: η+ = 1,05, η− = 0,5 подібно до Tollenaere (1990).

4.7 Результати

Початкові значення τ близькі до кінцевих (оптимальних), збільшення швидкості є меншим, однак комбіноване застосування «SuperSAB» і «Momentum» забезпечує величезне збільшення швидкості тренування при старті далеко від (раніше невідомого) фіналу значення τ (рис. 4.6). Це збільшення швидкості є послідовним у всьому діапазоні початкових значень τ (рис. 4.7).

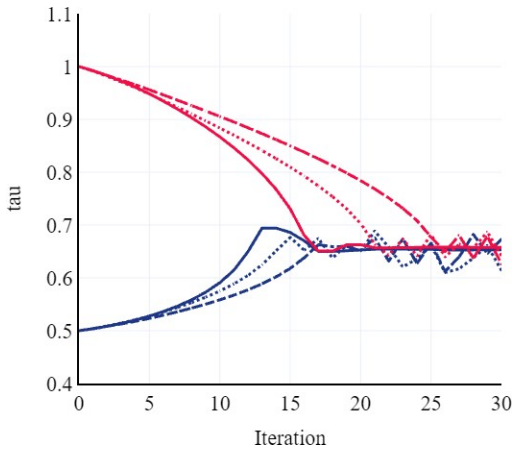


Рисунок 4.7 – Різні методи прискорення для динамічної версії τ суцільні

лінії представляють метод прискорення з «Momentum» і «SuperSAB».

На діаграмі зображено по осі ординат значення tau - функція дільників, на осі абсцис зображена ітерація.

Пунктирні лінії показують алгоритм з технікою «Momentum», і пунктирна лінія малює простий динамічний алгоритм. Значення τ видно на осі y. Прискорення та чистий динамічний алгоритм можуть знайти те саме, майже оптимальне значення τ, але з іншими, меншими швидкостями.

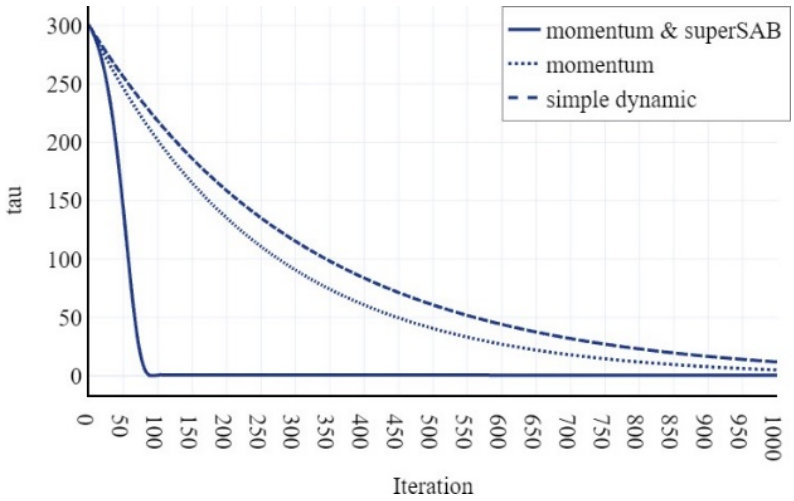


Рисунок 4.8 – Графіки порівняння прискорення методів

Помічене величезне прискорення спостерігається для вищих значень τ.

На діаграмі зображено по осі ординат значення tau-функція дільників, на осі абсцис зображена ітерація.

Найкращий метод прискорення знайшов відповідне значення τ менш ніж за 100 ітерацій, однак інші методи можуть навчатися значно повільніше.

4.7.1 Успішне збільшення швидкості навчання

Вимірювалася успішність підвищення швидкості навчання та перевага динамічного алгоритму над фіксованим.

4.7.2 Новий динамічний алгоритм

Процес навчання був обмежений (відносно високою) максимальною кількістю ітерацій (5000 кроків ітерації), але в більшості випадків застосована техніка ранньої зупинки була активною (попередньо вибране число ітерацій терпіння становило 200). Під час тестових експериментів лише приблизно в 5-6% усіх прогонів було зупинено за допомогою максимальної кількості ітерацій, і в більш ніж 90% співвідношенні була активна рання зупинка, отже, ці контрольні випадки можна навчити з набагато меншими кроками навчання (більше 5000 ітерацій). Це означає, що досягнення ліміту кроку ітерації (5000) є ознакою того, що навчання відчуло локальний, не оптимальний мінімум. На рис. 17 показано, скільки разів було досягнуто ліміту в 5000 кроків, використовуючи однакову кількість тренувань для фіксованого та нового динамічного алгоритму (загальна кількість тренувань для одного набору даних для однієї версії становила 480 (= 30 повторень x 16 різних/початкових τ)).

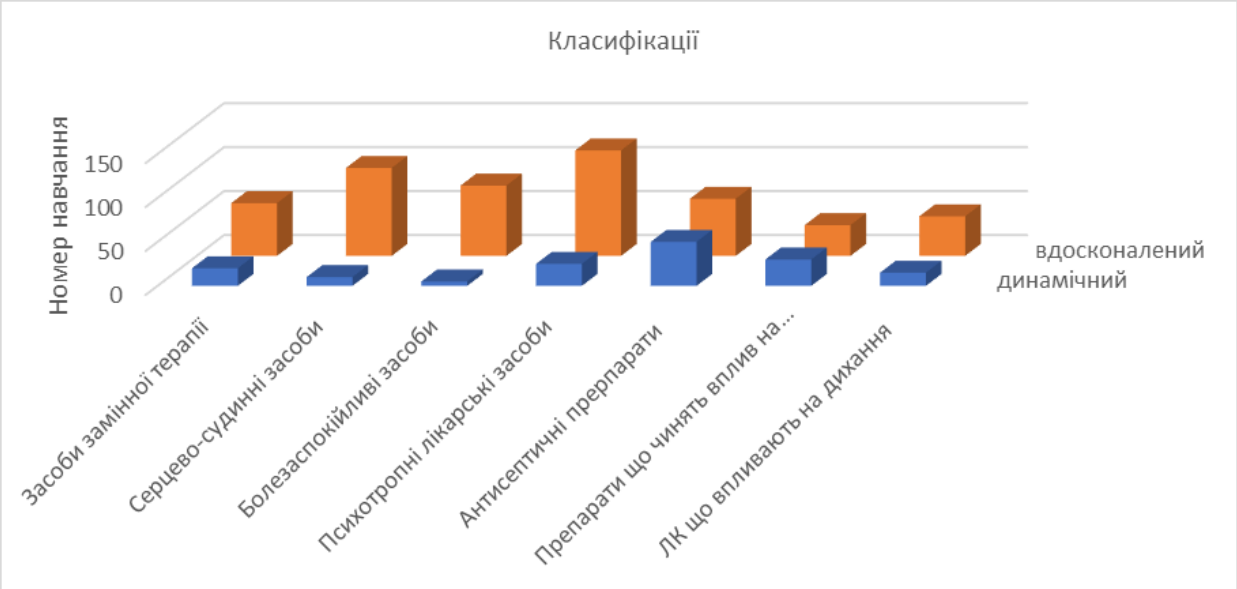


Рисунок 4.9 – Запропонована нова динамічна версія τ навчання значно швидша, ніж фіксована версія.

Оскільки ліміт кроку навчання застосовувався набагато частіше (у 2-10

разів частіше) у фіксованій версії, ніж у новому запропонованому алгоритмі динамічної версії τ, доведено, що запропонований алгоритм падає з набагато меншою ймовірністю до локального, а не оптимального мінімумів, отже, новий алгоритм набагато надійніший з огляду на процес навчання та його результат.

Слід зазначити, що ці додаткові результати стійкості отримані тому, що параметр τ налаштовується динамічно під час навчання, що неможливо з оригінальною, фіксованою версією τ, оскільки там τ є постійним.

4.8 Висновки

Загальновідомо, що моделювання на основі штучних нейронних мереж є перспективною сферою штучного інтелекту, яка постійно розвивається. Різноманітність застосованих вимірювань помилок у процесі навчання/перевірки спонукало до дослідження, а відповідний всебічний огляд літератури встановив, що не існує показника «найкращого», хоча існує набір показників зі змінними перевагами в різних навчальних ситуаціях.

Експерименти з класифікації показали, що SSE є найкращим для попереднього навчання та функції CE у тонкій настройці, отже, найкращою парою попереднього навчання/тонкої настройки є SSE/CE.

Основна ідея статті полягає в тому, щоб вийти далеко за рамки та інтегрувати відносну важливість, введену Сільвою та його колегами, в алгоритм(и) навчання нейронної мережі, реалізований за допомогою нового вимірювання помилок під назвою SuperSAB крім того, запропоноване рішення є в в той же час динамічний алгоритм навчання, тому не потрібні послідовні окремі етапи навчання. Цей підхід включено до (прискореного) алгоритму навчання Левенберга-Марквардта, тому також була представлена нова версія навчання Левенберга-Марквардта, що призвело до самоадаптивного динамічного алгоритму навчання. Цей динамізм позитивно впливає не лише на точність отриманої моделі, а й на сам процес навчання. Описані комплексні тести алгоритму довели, що він динамічно інтегрує два великі світи статистики та теорії інформації, що є

ключовою новинкою цієї дипломної роботи.

Запроваджена техніка збільшення швидкості (об’єднані кроки Левенберга-Марквардта, «Імпульс» і «SuperSAB») призвела до конвергентного рішення, крім того, такий вбудований динамічний алгоритм у кілька разів швидший за оригінальну версію з фіксованим τ. Будь ласка, зверніть увагу, що версія з фіксованим τ не допускає прискорення, оскільки цей параметр не можна змінити під час навчання.

ВИСНОВКИ

В результаті виконання магістерської роботи було удосконалення системи пошуку медичних препаратів за клінічним діагноз на основі штучних нейронних мереж з використанням на основі розробки нового метода для вдосконалення швидкості навчання .

При виконанні роботи було виконано наступні задачі

1. Проаналізовано існуючі методи навчання на основі штучної нейронної мережі для системи пошуку медичних препаратів.
2. Розроблено класифікацію різних методів прискорення алгоритмів навчання.
3. Модифіковано динамічний алгоритм навчання
4. Встановлено доцільність використання технік збільшення швидкості Левенберга-Марквардта, «Імпульс» і «SuperSAB»

Отже, вдосконалення системи пошуку медичних препаратів було вдосконалено за допомогою методів SUPERSAB та стартегія «імпульсу», а також за допомоги алгоритму Левенберга-Маркварда. Завдяки цим дослідженням було досягнуто підвищення значень менш ніж на 100 ітерацій, однак інші методи можуть навчатись значно повільніше.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. What is medicine? [Електронний ресурс] // Healthline Media. – 2004. – Режим доступу до ресурсу: https://www.medicalnewstoday.com/articles/323679.
2. Dany Paul Baby, MD. What Are the Different Types of Doctors? [Електронний ресурс] / Dany Paul Baby, MD // Medically Reviewed – Режим доступу до ресурсу: https://www.webmd.com/health-insurance/insurance-doctor-types.
3.
4. Neural network [Електронний ресурс] // TechTarget. – 2018. – Режим доступу до ресурсу: https://www.techtarget.com/searchenterpriseai/definition/neural-network.
5. Feed Forward Neural Network [Електронний ресурс] // DeepAI. – 2020. – Режим доступу до ресурсу: https://deepai.org/machine-learning-glossary-and-terms/feed-forward-neural-network.
6. Recurrent Neural Networks [Електронний ресурс] // IBM. – 2020. – Режим доступу до ресурсу: https://www.ibm.com/cloud/learn/recurrent-neural-networks.
7. Medsker L.R. Recurrent Neural Networks / Medsker L.R, Jain L.C. – Washington D.C: International Standard Book, 2000. – 2 с. – (CRC Press LLC). – (G8CA-199-2H1H).
8. Xinyu Zhang. A Design Methodology for Efficient Implementation of Deconvolutional Neural Networks on an FPGA / Xinyu Zhang. – UNIVERSITY OF CALIFORNIA, SAN DIEGO: East Eisenhower Parkway, 2017. – 1 с. – (ProQuest LLC). – (MI 48106 - 1346; кн. 10255385).
9. K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. In Proc. BMVC, 2014
10. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In Proc. CVPR, 2009
11. S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. CoRR, 2015

12. Kennedy Vieira Batista. Neural Networks and Deep Learning / Kennedy Vieira Batista, Michael Nielsen., 2019. – 260 с.
13. Element Materials Technology. A system for searching for medicinal products [Електронний ресурс] / Element Materials Technology // Element Materials Technology. – 2021. – Режим доступу до ресурсу: https://www.element.com/life-sciences/pharmaceutical/extractables-and-leachablestudies?utm_campaign.
14. A knowledge-based system to find over-the-counter medicines for self-medication [Електронний ресурс] // Elsevier Inc.. – 2020. – Режим доступу до ресурсу: https://www.sciencedirect.com/science/article/pii/S1532046420301325.
15. Elsevier B.V. The Digital Twin Paradigm for Smarter Systems and Environments: The Industry Use Cases [Електронний ресурс] / Elsevier B.V. // Copyright. – 2022. – Режим доступу до ресурсу: https://www.sciencedirect.com/topics/computer-science/multilayer-perceptron.
16. Madan Gupta. Soft Computing and Intelligent Systems / Madan Gupta. – McMaster University, Hamilton, Ontario, Canada: Academic Press, 1999. – 639 с. – (Academic Press). – (9780126464900; кн. 2147483647).
17. Advanced Data Mining Tools and Methods for Social Computing / Sourav De, Sandip Dey, Siddhartha Bhattacharyya, Surbhi Bhatia.. – 292 с. – (Academic Press). – (9780323857093; кн. 2147483647).
18. Pethuru Raj. The Digital Twin Paradigm for Smarter Systems and Environments: The Industry Use Cases / Pethuru Raj, Preetha Evangeline.. – 384 с. – (Academic Press). – (9780128187579; кн. 2147483647).
19. Sharing Data and Models in Software Engineering / Tim Menzies, Ekrem Kocaguneli, Burak Turhan, Leandro Minku.. – 406 с. – (Morgan Kaufmann). – (9780124172951; № 2147483647).
20. Kayli Leung. Learning Paradigms in Neural Networks [Електронний ресурс] / Kayli Leung – Режим доступу до ресурсу: https://medium.com/swlh/learning-paradigms-in-neural-networks-30854975aa8d.

21. Philip D. Wasserman. Neural Computing: Theory and Practice. Coriolis Group. June 1, 1989. 230 pages. ISBN-13 : 978-0442207434. ISBN-10 : 0442207433
22. Simon S. Haykin. Neural Networks: A Comprehensive Foundation.Prentice Hall, 1999 - 842 pages. ISBN-13: 978-0132733502 ISBN-10: 0132733501
23. Wasserman P. D. Experiments in translating Chinese characters using backpropagation. Proceedings of the Thirty-Third IEEE Computer Society International Conference.. — Washington: D. C.: Computer Society Press of the IEEE, 1988.
24. Werbos P. J., Beyond regression: New tools for prediction and analysis in the behavioral sciences. Ph.D. thesis, Harvard University, Cambridge, MA, 1974.
25. Rumelhart, David E.; Williams, Ronald J. (1986). Learning Internal Representations by Error Propagation. In: Parallel Distributed Processing (англ.) (Cambridge, MA: MIT Press) 1: 318—362.
26. Dey, A.K., Abowd, G.D.: Towards a better understanding of context and context-awareness. Technical Report GIT-GVU-99-22, Georgia Institute of Technology, College of Computing (June 1999)
27. Ranganathan, A., Campbell, R.H.: A Middleware for Context-Aware Agents in Ubiquitous Computing Environments. In: ACM/IFIP/USENIX International Middleware Conference (2003)
28. Byun, H.E., Cheverst, K.: Harnessing context to support proactive behaviours. In: ECAI2002 Workshop on AI in Mobile Systems, Lyon (2002)
29. Schmidt, A.: Potentials and Challenges of Context-Awareness for Learning Solutions. In: LWA 2005, pp. 63–68 (2005)
30. Dey, A.K., Abowd, G.D.: The context toolkit: Aiding the development of context-aware applications. In: Workshop on Software Engineering for Wearable andPervasive Comput- ing, Limerick, Ireland (June 2000)
31. Roman, Manuel, et al.: A Middleware Infrastructure to Enable Active Spaces. In: IEEE Pervasive Computing, pp. 74–83 (October-December 2002)

32. Shehzad, A., Ngo, H.Q., Pham, K.A., Lee, S.Y.: Formal Modeling in Context Aware Sys- tems. In: International Workshop on Modeling and Retrieval of Context (2004)
33. Yau, S.S., Karim, F.: An adaptive middleware for context-sensitive communications for real-time applications in ubiquitous computing environments. Journal of RealTime Sys- tems 26(1), 29–61 (2004)
34. Park, H.: A middleware of context-awareness for ubiquitous computing middlewares. In: International Conference on Information Systems, vol. 00, pp. 369–374 (2005)
35. Kumar, S.: neural networks. McGraw-Hill Education, New York (2005)
36. Dowson, N., & Bowden, R. (2007). Mutual information for lucas-kanade tracking (milk): An inverse compositional formulation. IEEE transactions on pattern analysis and machine intelligence, 30, 180– 185.
37. Fontes, T., Silva, L., Silva, M., Barros, N., & Carvalho, A. (2014). Can artificial neural networks be used to predict the origin of ozone episodes? *Science of the total environment*, *488*, 197–207.
38. Heravi, A. R., & Hodtani, G. A. (2016). A new robust correntropy based levenberg-marquardt algorithm. In 2016 Iran Workshop on Communication and Information Theory (IWCIT) (pp. 1–6). IEEE.
39. Heravi, A. R., & Hodtani, G. A. (2018c). Where does minimum error entropy outperform minimum mean square error? a new and closer look. *IEEE Access*, *6*, 5856–5864.
40. Heravi, A. R., & Hodtani, G. A. (2019). A new and fast correntropy-based method for system identifi- cation with exemplifications in low-snr communications regime. Neural Computing and Applications, 31, 4407–4422.
41. Kline, D. M., & Berardi, V. L. (2005). Revisiting squared-error and cross-entropy functions for training neural network classifiers. Neural Computing & Applications, 14, 310–318.

42. Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. Journal of the society for Industrial and Applied Mathematics, 11, 431–441.
43. Nilsaz-Dezfouli, H., Abu-Bakar, M., & Pourhoseingholi, M. (2016). An artificial neural network model for outcome prediction in gastric cancer patients. Journal of Fundamental and Applied Sciences, 8, 1687–1698.
44. Panin, G., & Knoll, A. (2008). Mutual information-based 3d object tracking. International Journal of Computer Vision, 78, 107–118.
45. Rady, H. A. K. (2011b). Shannon entropy and mean square errors for speeding the convergence of multilayer neural networks: A comparative approach. Egyptian Informatics Journal, 12, 197–209.
46. Rimer, M., & Martinez, T. (2006). Cb3: an adaptive error function for backpropagation training. Neural processing letters, 24, 81–92.
47. Silva, L. M., Santos, J. M., & Marques de Sa´, J. (2014). Classification performance of multilayer per- ceptrons with different risk functionals. International Journal of Pattern Recognition and Artificial Intelligence, 28, 1450013.
48. Tollenaere, T. (1990). SuperSAB: Fast adaptive back propagation with good scaling properties. Neu- ral Networks, 3, 561–573. URL: https://www.sciencedirect.com/science/article/pii/0893608090900067. doi:10.1016/0893-6080(90)90006-7.
49. Viharos, Z. J., Monostori, L., & Vincze, T. (2002). Training and application of artificial neural networks with incomplete data. Lecture Notes in Computer Science, 2358, 649–659.
50. Wen, D., Jia, P., Hsu, S. H., Zhou, Y., Lan, X., Cui, D., Li, G., Yin, S., & Wang, L. (2019). Estimating coupling strength between multivariate neural series with multivariate permutation conditional mutual information. Neural Networks, 110, 159–169. URL: https://doi.org/10.1016/j.neunet.2018.11.006. doi:10.1016/j.neunet.2018.11.006.

51. Yang, J., Cao, J., Wang, T., Xue, A., & Chen, B. (2020). Regularized correntropy criterion based semi- supervised ELM. Neural Networks, 122, 117–129. URL: https://doi.org/10.1016/j.neunet. 2019.09.030. doi:10.1016/j.neunet.2019.09.030.
52. Information Systems, Information Assurance, and Computer Science Resources Guide [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://library.dsu.edu/c.php?g=22496&p=133198.
53. Neural Networks [Електронний ресурс] // Refsnes Data. – Режим доступу до ресурсу: w3schools.com/ai/ai_neural_networks.asp.
54. Types of search systems [Електронний ресурс] // Neeva Inc. – 2022. – Режим доступу до ресурсу: https://neeva.com/learn/what-is-a-search-engine.
55. Different Types Of Search Engines [Електронний ресурс] // Marketing In Asia. – 2022. – Режим доступу до ресурсу: https://www.marketinginasia.com/different-types-of-search-engines/.
56. Інформаційно-пошукові системи [Електронний ресурс]. – 2013. – Режим доступу до ресурсу: http://megalib.com.ua/content/6369_53_Informaciino-poshykovi_sistemi_y_vidavnictvi.html.
57. about HELSI [Електронний ресурс] // helsi.me. – 2016. – Режим доступу до ресурсу: https://helsi.me/.
58. Класифікація нейронної мережі [Електронний ресурс] // uk_education. – 2022. – Режим доступу до ресурсу: https://uk.education-wiki.com/4761295-classification-of-neural-network.
59. Про таблетки юа [Електронний ресурс] // tabletki.ua – Режим доступу до ресурсу: https://tabletki.ua/uk/.
60. КЛАСИФІКАЦІЙНІ СИСТЕМИ ЛІКАРСЬКИХ ПРЕПАРАТІВ [Електронний ресурс] // Національний фарамацевтичний університет. – 2022. – Режим доступу до ресурсу: https://www.pharmencyclopedia.com.ua/article/3557/klasifikacijni-sistemi-likarskix-preparativ.
61. Основні положення теорії штучних нейронних мереж. // ПЗЕОМ. – 2020. – №1. – С. 4–7.

62. ANAND, R., MEHROTTRA, K. /.,MOHAN, C. K.,D in, S. 1993. An improved algorithm for neural network classification of imbalanced training sets. IEEE Trans. Neural Nets. 4.
63. Arim, M. AND So, F. M. A. 1992a. lDynamic learning using exponential energy function. In Proceedings o[the International storm Confer- ence on Neural Hetu›orbs.
64. Amino, M.D SALAM, F. M. A. 1992b. Error back-propagation learning using the polyno- mial energy function. In Proceedings of the [ourth International Conference on !S ysteni En- gineering.
65. BRYISON, A. E. wD Ho, Y. C. 1969. Applied Optical Control. Waltham MA.
66. DEVOS, M.R ORBAN, G.A.1988.Self learning backpropagation. In Proceedings the Neuro Nimes.
67. Yu, S., & Pr´ıncipe, J. C. (2019). Understanding autoencoders with information theoretic concepts. Neural Networks, 117, 104–123. URL: https://doi.org/10.1016/j.neunet.2019.05.003. doi:10. 1016/j.neunet.2019.05.003. arXiv:1804.00057.
68. Magoulas, G.D., Plagianakos, V.P., & Vrahatis, M.N. Globally Convergent Algorithms with Local Learning Rates. In IEEE Transactions on Neural Networks, Vol 13, No 3, 774 – 779, 2002.

ДОДАТОК А

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра інженерії програмного забезпечення

МАГІСТЕРСЬКА РОБОТА
«ВДОСКОНАЛЕННЯ СИСТЕМИ ПОШУКУ МЕДИЧНИХ ПРЕПАРАТІВ ЗА КЛІНІЧНИМ ДІАГНОЗОМ НА ОСНОВІ ШТУЧНОЇ НЕЙРОННОЇ МЕРЕЖІ »

Виконав: студент групи ПДМ–61, Маринскас Вадим Миколайович

Керівник: к.т.н., доц. кафедри ІПЗ Трінтіна НаталіяАльбертівна

Київ - 2022

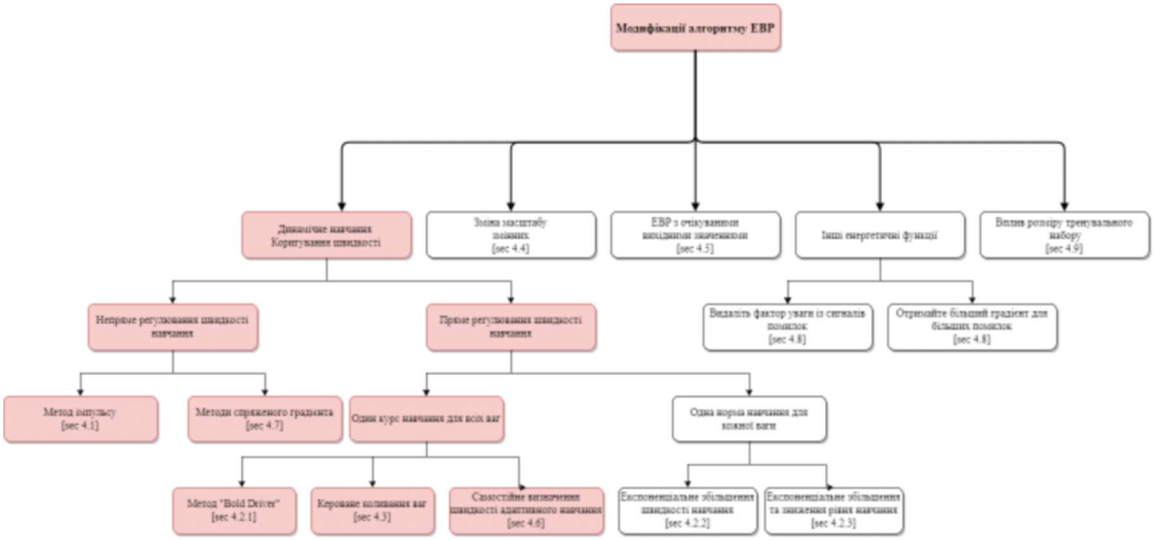
МЕТА, ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи підвищення ефективності пошуку медичних препаратів за допомогою штучної нейронної мережі

Об’єкт дослідження процес вдосконалення навчання нейронних мереж

Предмет дослідження методи вдосконалення нейронних мереж

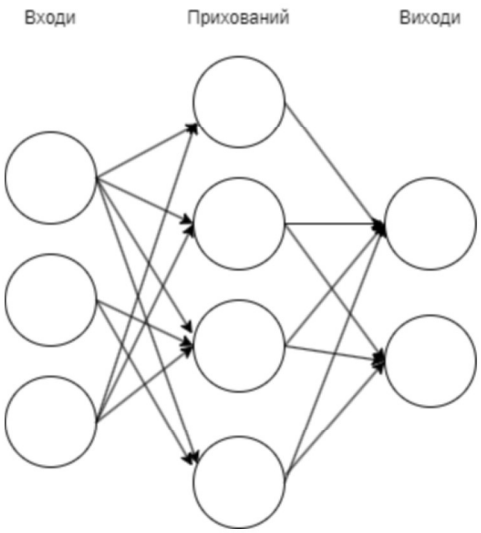
КЛАСИФІКАЦІЯ РІЗНИХ МЕТОДІВ ПРИСКОРЕННЯ АЛГОРИТМІВ НАВЧАННЯ



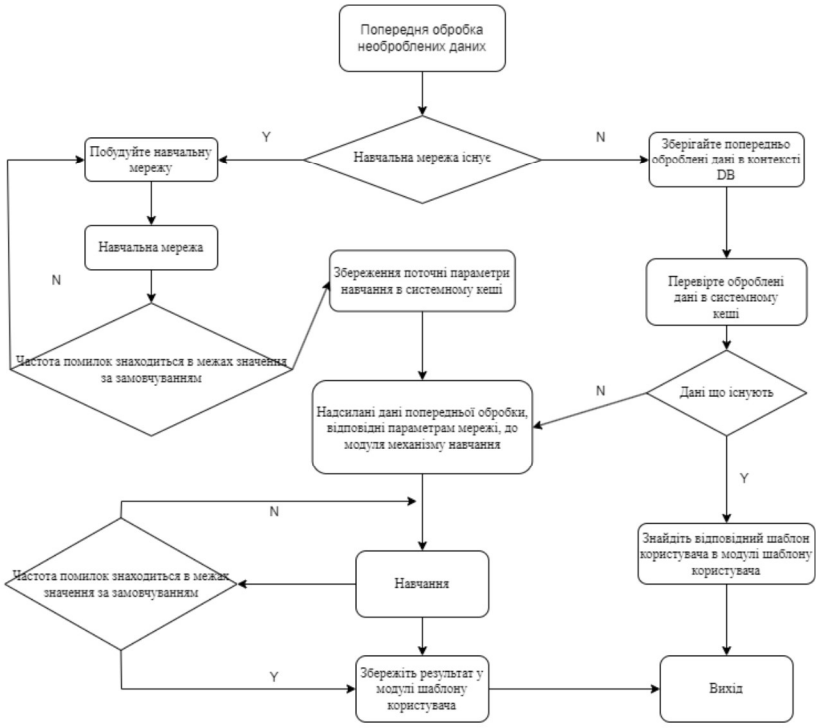
МАТЕМАТИЧНА МОДЕЛЬ БАГАТОШАРОВОГО ПЕРСЕПТРОНА

$o(x) = G\big(b(2) + W(2)h(x)\big)$
$h(x) = \Phi(x) = s(b(1) + W(1)x)$

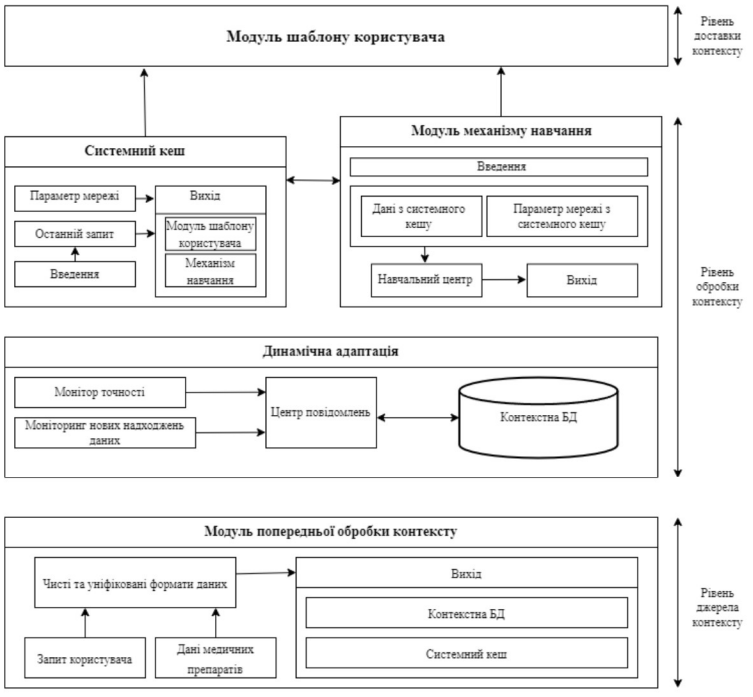
o-функція вихідного шару
h-функція прихованого шару
b(1),b(2)- вектори зміщення
W(1), W(2)- вагові матриці
G і s - функції активації
{W(1), b(1), W(2), b(2)}-набір параметрів для вивчення
Ф-функція
θ- порожня множина



АЛГОРИТМ НАВЧАННЯ ЗАПРОПОНОВАНОГО ПРОМІЖНОГО ПЗ



МОДУЛЬ ШАБЛОНУ КОРСТУВАЧА



АЛГОРИТМ ЛЕВЕНБЕРГА-МАРКВАРДА

Основним застосуванням алгоритму є задача вдосконалення та пришвидшення методів мінімізації нелінійної функції та отримання переваг інших алгоритмів оптимізації

$$argmin_{\beta} S(\cdot) = argmin_{\beta} = \sum_{i=1}^{m} [y_i - (x_i + \beta)]^2$$

m - емпіричні пари (x_i, y_i)

x_i - залежні змінні

Y_i - незалежні змінні

$(x, \cdot)$ - модельна крива

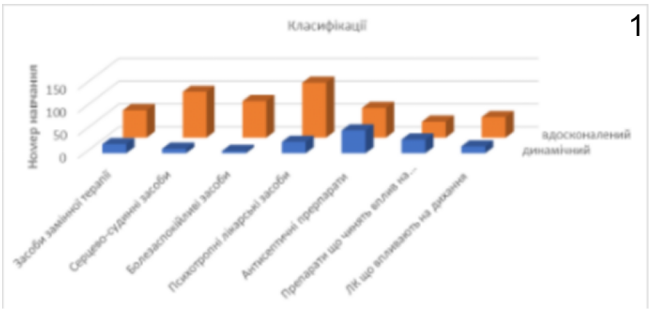
Σ - сума

$S(\cdot)$ - квадратні відхилення

$\cdot$ - параметри моделі

$argmin_{\beta}$ - мінімальне значення вздовж заданої осі

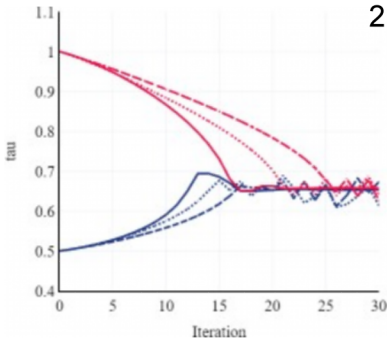
РЕЗУЛЬТАТИ ДОСЛІДЖЕНЬ



1

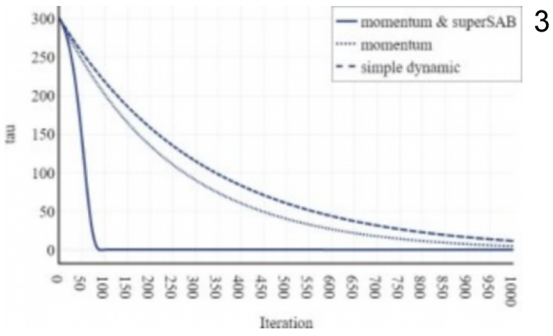
tau - функція дільників
iteration - номер ітерацій

Графік динамічної версії навчання



2

Методи прискорення для динамічної версії



3

Графік порівняння прискорення методів

ВИСНОВКИ

1. Проаналізовано існуючі методи навчання на основі штучної нейронної мережі для системи пошуку медичних препаратів
2. Розроблено класифікацію різних методів прискорення алгоритмів навчання
3. Модифіковано динамічний алгоритм навчання
4. Встановлено доцільність використання технік збільшення швидкості Левенберга-Марквардта, «Імпульс» і «SuperSAB»

ПУБЛІКАЦІЇ ТА АПРОБАЦІЯ РОБОТИ

Тези доповідей на конференціях

35
17 Трінтіна Н.А., Маринскас В.М. Вдосконалення системи пошуку медичних препаратів за клінічним діагнозом на основі штучної нейронної мережі // XV Науково-технічна конференція «Сучасні інфокомунікаційні технології» – Київ: ДУТ, 2022

ДЯКУЮ ЗА УВАГУ!