**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**
**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»**
**(КПІ ім. Ігоря Сікорського )**

ФАКУЛЬТЕТ БІОМЕДИЧНОЇ ІНЖЕНЕРІЇ
кафедра БІОМЕДИЧНОЇ КІБЕРНЕТИКИ

**«До захисту допущено»**

В.о. завідувач кафедри БМК

**_____ Євген НАСТЕНКО**

“___”_____2023р.

# Дипломна робота

**на здобуття ступеня бакалавра**
**за освітньо-професійною програмою**
***«Комп’ютерні технології в біології та медицині»***
**спеціальності *122 «Комп’ютерні науки»***

**на тему:** **Система аналізу біологічних сигналів**

Виконав: студент **IV** курсу, групи БС-91

**КУЛАКІВСЬКИЙ СТАНІСЛАВ МАКСИМОВИЧ** _____
(прізвище, ім’я, по батькові) (підпис)

Керівник: *д.б.н., к.т.н., проф., в.о.зав. каф. БМК*
**НАСТЕНКО ЄВГЕН АРНОЛЬДОВИЧ** _____
(посада, науковий ступінь, вчене звання, прізвище, ім’я, по батькові) (підпис)

Рецензент: *зав. каф. біомедичної інженерії, д.т.н. доцент*
**ШЛИКОВ ВЛАДИСЛАВ ВАЛЕНТИНОВИЧ** _____
(посада, науковий ступінь, вчене звання, прізвище, ім’я, по батькові) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2023 року

**Національний технічний університет України**

**«Київський політехнічний інститут імені Ігоря Сікорського»**

**Факультет біомедичної інженерії**

**Кафедра біомедичної кібернетики**

Рівень вищої освіти – перший (бакалаврський)

Спеціальність – 122 «Комп'ютерні науки»

Освітньо-професійна програма «Комп'ютерні технології в біології та медицині»

**ЗАТВЕРДЖУЮ**

В.о. завідувач кафедри БМК

_____ Євген НАСТЕНКО

«_30_»__травня___2023 р.

**ЗАВДАННЯ**
**на дипломну роботу студентові**

**КУЛАКІВСЬКИЙ СТАНІСЛАВ МАКСИМОВИЧ**

(прізвище, ім'я, по батькові)

1. Тема роботи **Система аналізу біологічних сигналів**

Керівник роботи
***Настенко Євген Арнольдович, д.б.н., к.т.н., проф. каф. БМК***

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «31» *травня* 2023 р. №***2106-с***

2. Термін подання студентом роботи ***08 червня 2023р.***

3. Вихідні дані до роботи**.** *Набір даних біологічних сигналів, мова програмування python.*

4. Зміст роботи *Анотації (на двох мовах); Вступ; Аналітичний огляд літературних джерел; Теоретична частина; Практична реалізація задачі за темою; Розрахунок економічного ефекту за темою ДР; Загальні висновки; Список використаних джерел.*

5. Перелік ілюстративного матеріалу (із зазначенням плакатів, презентацій тощо) *Презентація – на 17 слайдів, 31 Рисунків у роботі.*

6. Дата видачі завдання ***30 травня 2023 року.***

## Календарний план

| № з/п | Назва етапів виконання дипломної роботи | Термін виконання етапів роб оти | Примітка |
|---|---|---|---|
| 1 | Отримати завдання за темою ДР на практику | До 15.02.20223 р. | *виконано* |
| 2 | Переддипломна практика | За графіком | *виконано* |
| 3 | Виконання розділів ДР (Вступ, аналітичний огляд літературних джерел, теоретична частина) | До кінця практики | *виконано* |
| 4 | Виконання розділів ДР (практична частина, загальні висновки, список джерел) | Не пізніше 1 тижня до засідання каф-ри | *виконано* |
| 5 | Перевірка ДР науковим керівником | Не пізніше 1 тижня до засідання каф-ри | *виконано* |
| 6 | Подання в електронному вигляді ДР та анотації до неї на перевірку нормоконтролера та плагіат (UNICHECK). | ---- « ------ | *виконано* |
| 7 | Надання документів на засідання кафедри | За день до засідання | *виконано* |
| 8 | Предзахист ДР та допуск до захисту дисертації | Згідно плану каф. | *виконано* |
| 9 | Подання ДР рецензенту. Отримання рецензії. | До подання пакету документів до ЕК | *виконано* |
| 10 | Подання пакету документів по ДР та супровідних до неї документів до захисту в ЕК | За 5 днів до дати захисту ДР за графіком | *виконано* |
| 11 | Захист ДР в ЕК | | |

Студент _____ **Станіслав КУЛАКІВСЬКИЙ**
(підпис) (ім'я, ПРІЗВИЩЕ)

Керівник ДР _____ **Євген НАСТЕНКО**
(підпис) (ім'я, ПРІЗВИЩЕ)

Нормоконтролер _____ **Галина КОРНІЄНКО**
(підпис) (ім'я, ПРІЗВИЩЕ)

# АНОТАЦІЯ

Дипломна робота за темою «Система аналізу біологічних сигналів» виконана студентом кафедри біомедичної кібернетики ФБМІ *Кулаківським Станіславом Максимовичем зі спеціальності 122 «Комп'ютерні науки» за освітньо-професійною програмою «Комп'ютерні технології в біології та медицині» та* складається зі: вступу; 3 розділів (*аналітичний огляд літературних джерел, теоретична частина, практична реалізація задачі)*, висновків до кожного з цих розділів; загальних висновків; списку використаних джерел, який налічує 34 джерел. Загальний обсяг роботи 70 сторінок.

***Актуальність теми.*** Актуальною тема є з причини стрімкого розвитку медицини й зростанню потребності аналізу біологічних сигналів.

***Мета і завдання роботи.*** Метою роботи є використання якісних даних для розшифрування біологічних сигналів.

Її досягнення передбачає вирішення наступних *завдань*:

1. Аналіз інформації з наукових джерел.
2. Обрання мови програмування для виконання роботи.
3. Визначення вимог програмного застосунку.
4. Реалізація програмного застосунку.
5. Розшифрування біологічних сигналів.

***Використані матеріали:*** датасет біологічних сигналів, мова програмування python, біблотеки python.

***Отримані результати:*** Розроблено й використано програми для аналізу біологічних даних.

***Ключові слова*:** шум, інформаційне середовище, біологічні сигналіи, python.

***Бібліографічний опис ДР.***

Кулаківський С. М. Система аналізу біологічних сигналів : дипломна роб. бакалавра : 122 Комп'ютері науки / Кулаківський Станіслав Макстимович, 2023. – 69 с.

# ABSTRACT

The thesis on the topic "Biological signal analysis system" was completed by *Stanislav Kulakivskyi, a student of the Department of Biomedical Cybernetics, FBMI, majoring in 122 "Computer Science" under the educational and professional program "Computer technologies in biology and medicine" and* consists of: introduction; 3 sections (*analytical review of literature sources, theoretical part, practical implementation of the problem)*, conclusions to each of these sections; general conclusions; of the list of used sources, which includes 34 sources. The total volume of work is 70 pages.

***Actuality of theme.*** The topic is relevant due to the rapid development of medicine and the growing need to analyze biological signals.

***The purpose and tasks of the work.*** The goal is to use high-quality data to decipher biological signals.

Its achievement involves solving the following *tasks:*

1. Analysis of information from scientific sources.
2. Choosing a programming language to perform the work.
3. Determining the requirements of the software application.
4. Implementation of the software application.
5. Decoding biological signals.

***Materials used.*** Biological signals dataset, python programming language, python libraries.

***Obtained results.*** Programs for analyzing biological data have been developed and used.

***Keywords.*** Noise, information environment, biological signals, python.

**ЗМІСТ**

# ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ЕЕГ – Електроенцефалографія

МЕГ – магнетоенцефалографія

МРТ – магнітно-резонансної томографії

НІСС – ближньо-інфрачервона спектроскопія

НКІ – Нейро-комп'ютерний інтерфейс

ПЗ – програмне забезпечення

Рис. – Рисунок

Табл. – таблиця

ФМРТ – функціональна магнітно-резонансна томографія

# ВСТУП

Актуальною тема є з причини стрімкого розвитку медицини й зростанню потребності якісних даних для подальшого аналізу біологічних сигналів для широкого застосування в межах медицини й поза нею.

**Обґрунтування прийнятих рішень за темою дипломної роботи.**

В процесі проведеного аналізу було визначено та обґрунтовано використання: мови програмування Python, безкоштовних бібліотек BrainCode, NumPy. Додатково для розшифрування сигналів мозку використовувалась large language model.

***Мета і завдання роботи*.** Метою роботи є використання якісних даних для розшифрування біологічних сигналів.

Її досягнення передбачає вирішення наступних *завдань*:

1. Аналіз інформації з наукових джерел.
2. Обрання мови програмування для виконання роботи.
3. Визначення вимог програмного застосунку.
4. Реалізація програмного застосунку.
5. Розшифрування біологічних сигналів.

***Використані матеріали:*** датасет біологічних сигналів, мова програмування python, біблотеки python.

***Отримані результати.*** Виконано всі вказані завдання роботи та реалізовано програмні застосуноки для аналізу біологічних даних.

***Публікації.*** За результатами виконаної роботи публікації не передбачено.

***Структура роботи.*** Дипломна робота за темою «Система аналізу біологічних сигналів» виконана студентом кафедри біомедичної кібернетики ФБМІ *Кулаківським Станіславом Максимовичем зі спеціальності 122 «Комп'ютерні науки» за освітньо-професійною програмою «Комп'ютерні технології в біології та медицині» та* складається зі: вступу; 3 розділів (*аналітичний огляд літературних джерел, теоретична частина, практична реалізація задачі)*, висновків до кожного з цих розділів; загальних висновків;

списку використаних джерел, який налічує 34 джерел. Загальний обсяг роботи 69 сторінок друкованого тексту.

# РОЗДІЛ 1
# АНАЛІТИЧНИЙ ОГЛЯД ЛІТЕРАТУРНИХ ДЖЕРЕЛ

## 1.1 Мозок людини

Головний мозок людини, або церебрум, є центральним органом нашої нервової системи і керує великою кількістю важливих функцій. Він є основним місцем для вищих когнітивних функцій, таких як мислення, мова, обробка інформації та усвідомлення світу навколо нас.

Церебрум складається з двох півкуль, які поділені на чотири основні області, відомі як лобовий, тім'яний, затилковий та скроневий відрізки. Кожен з них має свої особливості та спеціалізовані функції.

Лобовий відрізок відповідає за виконання завдань, пов'язаних з плануванням, організацією, проблемним мисленням та соціальною поведінкою. Тім'яний відрізок відповідає за сприйняття і розпізнавання стимулів зі слуху, запаху, смаку та температури. Затилковий відрізок відповідає за обробку зорової інформації, а скроневий відрізок — за пам'ять та мову.

Мозок складається з маси нервових клітин, відомих як нейрони, які взаємодіють між собою через електрохімічні сигнали. Нейрони утворюють мережу, що створює та передає ці сигнали по всьому мозку і тілу.

Мозок захищений черепом, який служить твердим щитом від зовнішніх пошкоджень. Внутрішнє середовище мозку стабілізується з допомогою спинномозкової рідини, що амортизує удари та вібрації.

З ростом і розвитком людини форма головного мозку змінюється і набуває форми черепа. Це включає в себе розвиток гірок та борозд, які збільшують поверхню мозку і дозволяють вмістити більше нейронів.

Мозок зовнішньо нагадує жовтувату желеподібну масу. Вік та стать мають вплив на вагу мозку. Типова вага мозку дорослого чоловіка становить близько 1375 г, порівняно з 380-400 г для немовлят та 1275 г для жінок. Коли-то

вважалося, що інтелектуальні здібності людини залежать від ваги її мозку, але ця ідея виявилася помилковою.

Хоча вага мозку не корелює безпосередньо з IQ, існує певна межа (900 г), нижче якої мозок вважається дисфункціональним. Відношення ваги спинного мозку до ваги мозку, що у людини становить 1:50, використовується для визначення рівня розвитку мозку.

Мозок складається з трьох основних частин: мозкового стовбура, підкоркового відділу та кори великого мозку.

Мозковий стовбур — це найнижча та найбільш примітивна частина мозку, яка служить перехідним зв'язком між вищими частинами мозку та спинним мозком. Він відповідає за регуляцію вітальних життєвих функцій, таких як дихання, серцебиття та кровообіг.

Мозковий стовбур поділяється на довгастий мозок, місто та середній мозок. Довгастий мозок – це найнижча частина мозку, яка є продовженням спинного мозку і містить центри, відповідальні за керування диханням та кровообігом.

Міст – це частина мозкового стовбура, яка з'єднує довгастий та середній мозок. Він містить важливі провідні шляхи між мозком і тілом та відповідає за координацію рухів.

Середній мозок – найкоротший відділ мозкового стовбура. Він містить дві пари горбків, що відіграють роль в обробці зорової та слухової інформації.

Проміжний мозок розташований між середнім та підкорковим відділом. Він має важливі зв'язки з гіпофізом, ендокринною залозою, що регулює більшість гормональних функцій організму.

Підкорковий відділ включає в себе структури, такі як гіпоталамус, таламус та базальні ганглії, що регулюють емоції, пам'ять, а також координацію та планування рухів.

Кінцева частина, кора великого мозку, є найбільш розвиненою структурою мозку і відповідає за вищі когнітивні функції, такі як мова, мислення, сприйняття та свідомість. «З початку 2000-х років рекомендують для використання в українській медичній термінології термін мозкові оболонки, для означення всіх

інших (райдужної, слизових тощо) і надалі рекомендують використовувати термін оболонки.»[31, 32, 33, 34]

Підсумовуючи, головний мозок людини — витвір складної структурної організації. Він складається зі множини нейронів, які утворюють нейронні мережі та здійснюють передачу інформації. Цей орган також містить спеціалізовані області, що відповідають за різні функції та процеси, такі як сприйняття, мова, рух та когнітивні функції.

Ключова частина – нейрони, які працюють за допомогою системи електрохімічних сигналів.

Нейрони мають унікальну структуру, що дозволяє їм виконувати свої функції. Вони складаються з тіла клітини, яке містить ядро і основні органели, такі як мітохондрії та ендоплазматичну сітку. Аксон – це довгий відросток нейрона, який передає сигнали від тіла клітини до інших нейронів або ефекторних органів.

Сигнали передаються між нейронами через синапси – контактні точки між аксонами одного нейрона та дендритами іншого. У синапсі електричні сигнали перетворюються на хімічні сигнали за допомогою нейромедіаторів – речовин, які передають сигнал від одного нейрона до іншого. Цей процес називається синаптичною передачею.

Система електрохімічних сигналів в нейронах дозволяє реалізовувати складні мозкові функції, включаючи сприйняття, мислення, рух та емоції. Це фантастичне утворення натхненної природою еволюції, яке забезпечує нам можливість розуміти і взаємодіяти з навколишнім світом.

Розуміння роботи нейронів та їхньої взаємодії через синапси є ключовою частиною адже їх роботу можна зчитати й перенести в вигляд даних.

### 1.2 Сигнали мозку. Роль мозкових хвиль у когнітивних процесах.

Мозкові хвилі, або нейронні осциляції, відіграють ключову роль у функціонуванні людського мозку. Ці осциляції створюються синхронізованими

електричними імпульсами від мас нейронів, що спілкуються між собою.

Важливим фактом є еластичність нейронних з’єднань. «Дослідження спростували деякі поширені хибні уявлення про мозок. Це стосується як давніх, так і сучасних міфів. Наприклад, неправда, що нейрони не замінюються після двох років; або що нормальна людина використовує лише десять відсотків мозку.»[11]

Електрична активність мозку може бути виміряна електроенцефалограмою (ЕЕГ), яка класифікує ці осциляції на різні частотні діапазони, кожен з яких пов'язаний з різними когнітивними станами та розумовими активностями.
«У біологічному контексті, зручно розрізняти два види синхронізації:

- локальна синхронізація
- фазова синхронізація)»[13]

Дельта-хвилі (0,5-4 Гц) є найповільнішими мозковими хвилями, які зазвичай асоціюються з глибоким, безсновидним сном та сприяють процесам оздоровлення та зростання організму. Тета-хвилі (4-8 Гц) часто спостерігаються під час фаз легкого сну або глибокої розслабленості та пов'язані з консолідацією пам'яті та внутрішньою увагою. Альфа-хвилі (8-13 Гц) пов'язані зі станом розслабленості та спокою, і часто спостерігаються під час практики міндфулнесу та медитації. Бета-хвилі (13-30 Гц) асоціюються з активним мисленням та концентрацією, тоді як гамма-хвилі (30-100 Гц) пов'язані з високорівневою когнітивною обробкою, такою як сприйняття та свідомість.

Взаємодія між цими різними мозковими хвилями дозволяє мозку функціонувати з високим рівнем складності. Наприклад, взаємодія між тета- і гамма-хвилями вважається важливою для формування та відтворення пам'яті. Тета-хвилі надають певну часову структуру високочастотним гамма-хвилям, дозволяючи різним частинам мозку синхронізувати свою активність та ефективно спілкуватися між собою.

«Під час фази швидкого сну (підвищена тета-активність та локальна синаптична пластичність), ре-активовані спогади у кортексі закріплюються на синаптичному рівні, тобто відкладаються у довготривалу пам'ять.»[7, 6]

## 1.3 Техніки зчитування мозку в дослідженнях когнітивних процесів

### 1.3.1 Неінвазивні техніки зчитування мозку

Неінвазивне зчитування є методом, який дозволяє дослідникам вивчати активність мозку без потреби в хірургічних втручаннях або інших вторгнутих процедурах. Цей підхід базується на використанні різноманітних методів зображення, таких як функціональна магнітно-резонансна томографія, електроенцефалографія (ЕЕГ), магнетоенцефалографія та ближньопольова інфрачервона спектроскопія. Завдяки цим методам дослідники можуть отримувати інформацію про активність мозку та розуміти його функціональні процеси. Це відкриває широкі можливості для досліджень та діагностики нейрологічних захворювань без ризику для пацієнтів.

«ЕЕГ найчастіше використовується для діагностики епілепсії, яка спричиняє аномалії ЕЕГ.»[25]

Електроенцефалографія також використовується для графічного реєстрування біопотенціалів головного мозку та аналізу його фізіологічної зрілості та стану. Цей метод передбачає реєстрацію та аналіз біоелектричної активності головного мозку, що відображається на електроенцефалограмі. ЕЕГ дозволяє виявляти осередки уражень та загальномозкові розлади. Зазвичай ЕЕГ знімається шляхом розміщення електродів на шкірі скальпу, поверхні мозку або навіть у глибоких структурах мозку.

ЕЕГ широко використовується для діагностики епілепсії, виявлення аномалій та в оцінці стану пацієнтів. Цей метод також допомагає діагностувати порушення сну, ступінь анестезії, кому, енцефалопатію та смерть мозку. Хоча раніше ЕЕГ була основним методом для діагностики пухлин, інсультів та інших фокальних порушень головного мозку, з моменту виникнення технік магнітно-резонансної томографії (МРТ) та комп’ютерної томографії (КТ), поширення застосування ЕЕГ було зменшено. Проте ЕЕГ залишається цінним інструментом для досліджень та діагностики завдяки його мобільності та високій часовій роздільній здатності, недосяжній для КТ або МРТ.

Незважаючи на це, неінвазивні методи зчитування мозку, такі як магнетоенцефалографія (МЕГ), функціональна магнітно-резонансна томографія (фМРТ) та ближньо-інфрачервона спектроскопія (НІСС), не потребують хірургічного втручання. Вони надають можливість вимірювати та аналізувати активність мозку, виявляючи зміни, пов'язані з кровотоком та біоелектричною активністю. Ці методи є безпечними та неінвазивними способами дослідження мозку та мають великий потенціал у нейронауці та діагностиці різних нейрологічних станів та захворювань.

«Перевага у вимірюванні магнітних полів, створюваних нервовою активністю, полягає в тому, що вони, ймовірно, будуть менш спотворені навколишніми тканинами (особливо черепом і шкірою голови) в порівнянні з електричними полями, вимірюваними електроенцефалографією.»[30]

«ФМРТ зазвичай класифікується як мінімальний або помірний ризик через його неінвазивність порівняно з іншими методами візуалізації. ФМРТ використовує контраст, залежний від рівня оксигенації крові, щоб створити свою форму зображення.»[5, 26]

Ці техніки є безпечнішими для користувача, але пропонують меншу роздільну здатність та більш схильні до шуму порівняно з інвазивними методами.

Наприклад, «Комп’ютерна томографія може піддати пацієнтів радіації, яка в 100-500 разів перевищує рівень традиційного рентгенівського випромінювання, причому більші дози радіації створюють зображення з кращою роздільною здатністю.»[24]

### 1.3.2 Інвазивні техніки зчитування мозку

Інвазивні методи зчитування мозку передбачають пряму взаємодію з мозковою тканиною, що вимагає хірургічного втручання для встановлення електродів. Ці інвазивні нейро-комп'ютерні інтерфейси є одним з активно вивчених підходів у цій області, забезпечуючи високу роздільну здатність зчитування сигналів та точне керування протезними пристроями, що особливо корисно для пацієнтів зі складними руховими порушеннями. Однак, варто

відзначити, що такі техніки пов'язані з ризиками, такими як інфекції, тканинні ушкодження та проблеми довготривалої стабільності імплантату.

Вибір між інвазивними та неінвазивними техніками часто вимагає знаходження компромісу між роздільною здатністю та безпекою. Для деяких застосувань, наприклад, нейропротезів для пацієнтів зі складною паралізією, переваги інвазивних технік можуть переважити ризики, пов'язані з ними. У той же час, для застосувань, таких як нейрофідбек або тренування мозку, зазвичай віддають перевагу неінвазивним технікам, які є безпечнішими та менш нав'язливими для пацієнтів.

### 1.4 Приклади використання оброблених сигналів

Дослідники з École Polytechnique Fédérale de Lausanne (EPFL) створили алгоритм машинного навчання під назвою "CEBRA", який був навчений відповідності між активністю нейронів і конкретними кадрами у відео. Завдяки цьому алгоритму вдалося передбачити та відновити фрагменти фільмів, які бачить миша [22].

Ця технологія відкриває широкі можливості та потенціал для дослідження діяльності мозку і розуміння того, як ми сприймаємо та обробляємо візуальну інформацію. Розкодоване зображення знизу можна порівняти з оригіналом зверху (на рис 1.1).

Але важливо усвідомлювати можливі ризики та етичні питання, пов'язані зі зловживанням цією технологією людьми.

Така технологія може створювати потенціал для незаконних дій які порушують приватність особин. Наприклад, можливість відтворення того що бачить людина, може бути використана для незаконного перегляду або стеження за іншими людьми без їхньої згоди. Також може виникнути можливість зловживання такою технологією для маніпулювання або контролю над іншими.


Рисунок 1.1 – Зчитування мозку миші для відображення спогадів[19]

**Висновки до розділу 1**

Мозок є системою зі змінними параметрами які можливо зчитати й перевести в біологічні сигнали.

Вибір між інвазивними та неінвазивними техніками часто вимагає компромісу між роздільною здатністю, якістю сингала та безпекою користувача. Попередньо, неінвазивні системи зчитування виглядають кращим варіантом через те що вони є безпечними, не мають ризиків, пов'язаних з хірургічними процедурами, мають простіший спосіб використання

При використанні технологій зчитування сигналів мозку, користувачам необхідно пам’ятати про заходи безпеки для запобігання недозволеним зломам та зловживанням.

Варто враховувати користувачам що пов'язані ризики включають можливість порушення особистих прав, можливу недостатню захищеність від несанкціонованого доступу до персональних даних, а також можливість використання цих технологій для небезпечних або негативних цілей.

# РОЗДІЛ 2
# ТЕОРЕТИЧНА ЧАСТИНА

## 2.1 Аналіз сигналів мозку

Аналіз сигналів зчитування мозку — це важливий компонент у використанні нейро-комп'ютерних інтерфейсів (НКІ). Цей процес включає збір, обробку та інтерпретацію сигналів, які генеруються нейронами мозку. Основними кроками у аналізі сигналів є виявлення характеристик, класифікація сигналів та перетворення цих сигналів у команди для НКІ.

### 2.1.2 Методи аналізу сигналів

Аналіз сигналів представляє собою фундаментальний інструмент у наукових дослідженнях, який залучає ряд методологій для розуміння, інтерпретації та маніпулювання даними. Ці методи мають широкий спектр застосувань, включаючи обробку медичних зображень, аудіо обробку, телекомунікації, радіоастрономію, а також дослідження мозку.

Спектральний аналіз – це один з найбільш відомих методів, що використовуються для аналізу сигналів. Він працює шляхом розкладання сигналу на складові частоти, що дозволяє детально дослідити частотний спектр сигналу. Це надзвичайно корисно для виявлення характеристик сигналу, які можуть бути прихованими в часовому домені. За допомогою спектрального аналізу можна визначити основну частоту сигналу, виявити гармоніки, а також розпізнати шум чи інтерференцію.

Просторовий аналіз – це метод, що використовується для виявлення активності в окремих регіонах простору, наприклад, мозку. Це дозволяє науковцям вивчати, як різні частини мозку співпрацюють та взаємодіють між собою під час виконання різних завдань. Наприклад, за допомогою просторового аналізу можна визначити, які області мозку активуються під час виконання певних видів діяльності, що дозволяє краще зрозуміти функціональну організацію мозку.

Тимчасовий аналіз, спрямований на вивчення сигналу включає в себе вивчення змін сигналу протягом певного періоду часу. Тимчасовий аналіз може допомогти виявити зміни в сигналі, викликані різними факторами, такими як фізіологічні стани або зовнішні стимули. Наприклад, за допомогою тимчасового аналізу можна визначити, як мозок реагує на певний стимул у визначений момент часу.

Усі ці методи разом дозволяють отримати багатошаровий аналіз нейрональної активності, враховуючи часові, просторові та спектральні аспекти сигналу. Вони допомагають виявити складні патерни в даних і відкривають нові можливості для вивчення структури та функції мозку.

### 2.1.2 Обробка сигналів і видалення шуму

Обробка сигналів в контексті зчитування мозку включає в себе ряд методів та технік, спрямованих на забезпечення якісного та точного отримання нейронних сигналів. Важливим етапом є видалення шуму та артефактів, які можуть впливати на точність і достовірність отриманих даних як на прикладі (рис 2.1).

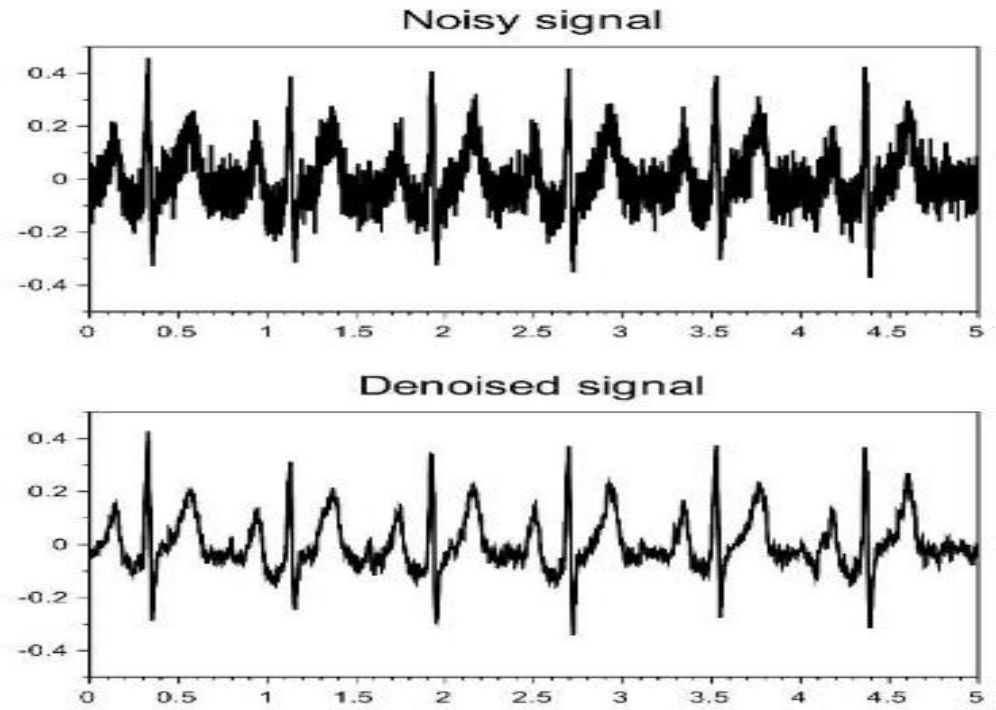



Рисунок 2.1 – Прибрання шуму з сигналу.

Один з підходів до видалення шуму – використання фільтрації сигналів. Це може включати використання фільтрів нижніх частот для видалення високочастотних шумів, або фільтрів верхніх частот для позбавлення низькочастотних спотворень. Крім того, можуть застосовуватися фільтри середньої частоти, які дозволяють забрати шум з певного діапазону частот. Вибір конкретного фільтру залежить від особливостей сигналу та вимог до його обробки. Знешумлений сигнал можна порівняти з оригіналом (рис 2.2)

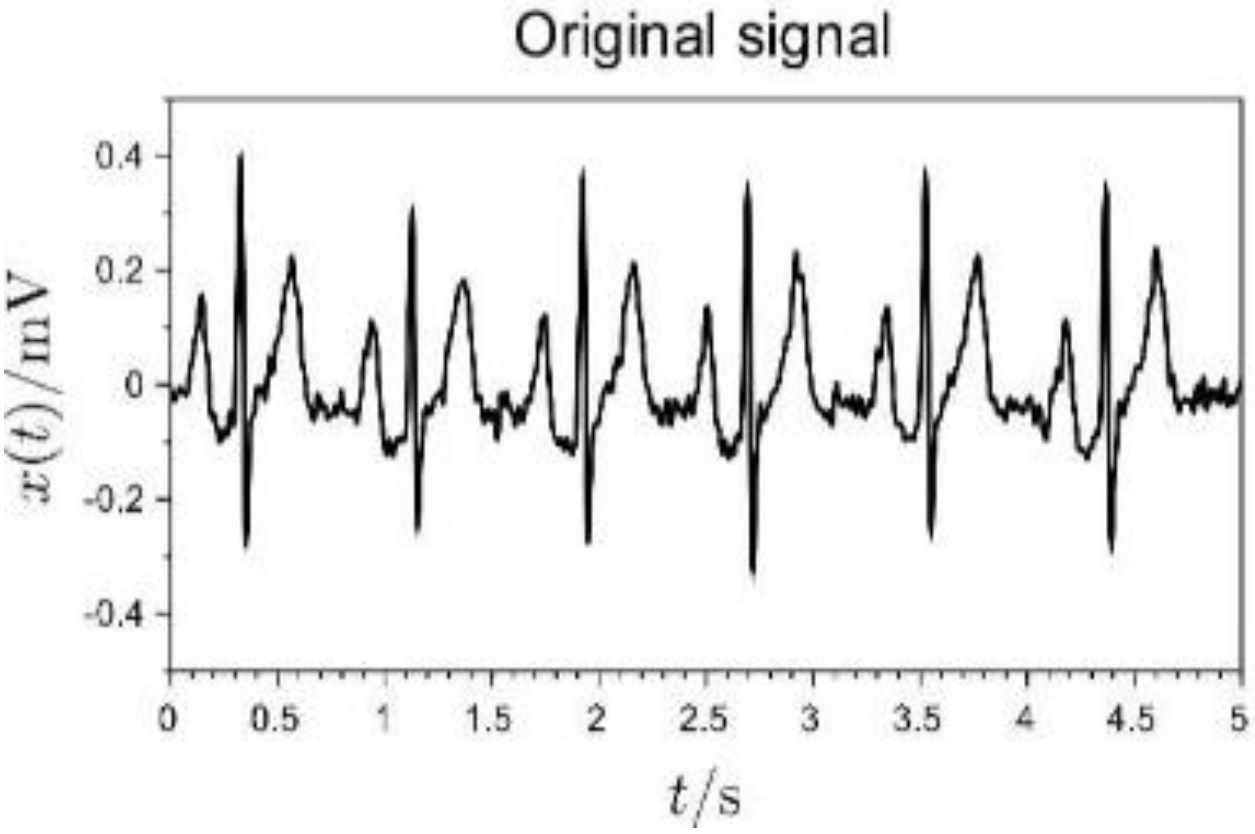



Рисунок 2.2 – Оригінальний сигнал

Ще одним підходом є використання статистичних методів для виявлення та виправлення артефактів, які можуть бути спричинені рухом або м’язовою активністю. Наприклад, метод "ізоляції компонент від фону" дозволяє виділити сигнали, пов'язані з конкретною мозковою активністю, від шуму та фонової активності. Цей підхід базується на розкладанні сигналу на компоненти та

видаленні небажаних компонентів, що допомагає поліпшити якість та достовірність отриманих даних.

Не допускати зашумленості (рис 2.3) шляхом обмеження впливу шумів на сигнал (рис 2.4).

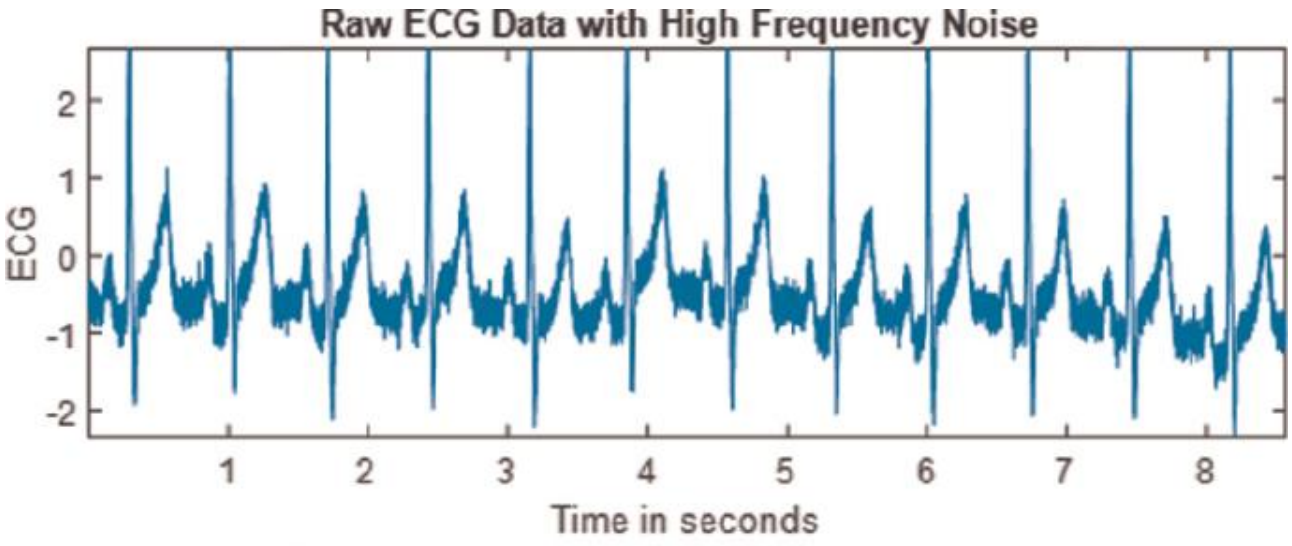



Рисунок 2.3 – Зашумлений сигнал

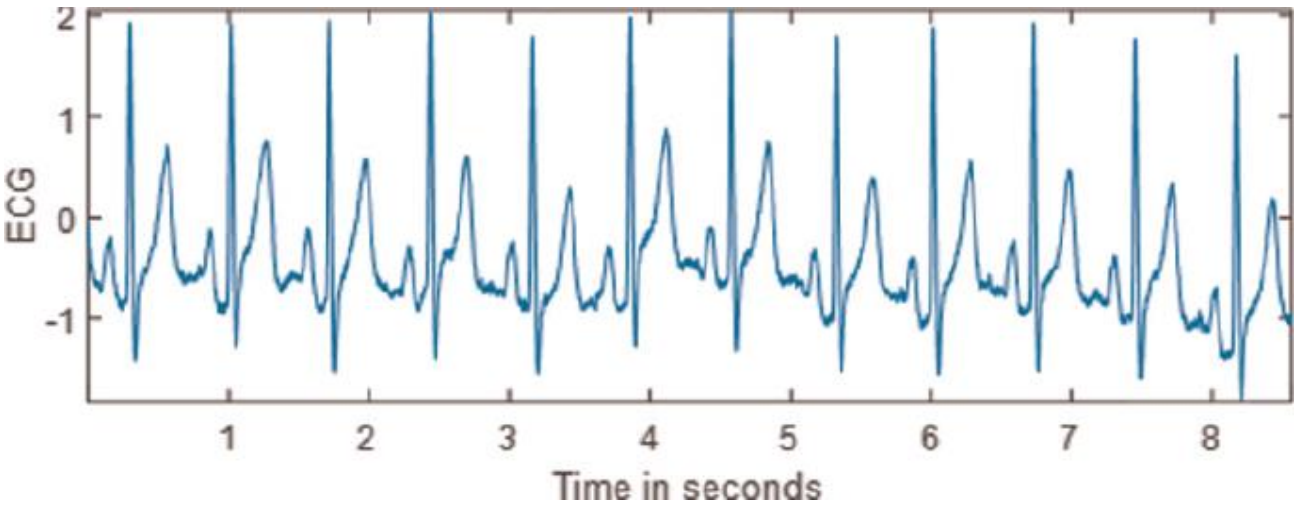



Рисунок 2.4 – Чистий сигнал

Додатковою важливою темою в обробці сигналів є компенсація за рухом. Оскільки зчитування мозку відбувається в реальному часі, рухи голови чи тіла можуть спотворювати сигнал і збільшувати шум. Це може вплинути на точність та достовірність отриманих даних. Для компенсації за рухом застосовуються

різноманітні методи, такі як корекція руху голови за допомогою відеокамер або використанням алгоритмів, які враховують рухи голови та коригують отримані сигнали. Це допомагає знизити вплив руху на сигнали та забезпечити більш точні результати.

### 2.1.3 Виділення характеристик та класифікація

Після отримання та обробки нейронних сигналів, важливим кроком є аналіз цих сигналів з метою виділення характеристик, що можуть бути використані для ідентифікації конкретних станів мозку. Це може включати аналіз часових, частотних або просторових властивостей сигналів, а також використання складніших алгоритмів та методів, таких як перетворення Фур'є, вейвлет-аналіз чи просторова фільтрація.

Отримані характеристики можуть бути використані для тренування класифікаторів, які здатні визначати наміри та дії користувача на основі нейронних сигналів. Класифікатори можуть бути навчені розпізнавати різні мозкові стани, такі як концентрація, візуальне сприйняття або наміри руху. Використовуються різноманітні алгоритми класифікації, включаючи лінійні дискримінантні аналізи, машини опорних векторів, нейронні мережі та інші. Вибір конкретного класифікатора залежить від властивостей сигналів та задачі дослідження.

Важливо зазначити, що тренування класифікаторів вимагає наявності відповідних навчальних даних, які включають нейронні сигнали та відповідні наміри або дії користувача. Ці дані можуть бути зібрані шляхом проведення експериментів з добровольцями, які виконують певні завдання або мають специфічні становища. Добре підготовлені та репрезентативні дані є ключовими для досягнення високої точності та надійності класифікаторів.

### 2.1.4 Перетворення сигналів у команди НКІ

Одним із фінальних етапів дослідження зі зчитування мозку є перетворення класифікованих нейронних сигналів у команди, які можуть бути використані для керування різноманітними пристроями та інтерфейсами. Цей процес відіграє важливу роль у розвитку технологій, які можуть допомагати

людям з руховими вадами або обмеженнями, а також у створенні більш ефективних та інтуїтивно зрозумілих інтерфейсів.

Одним з основних напрямків перетворення класифікованих сигналів є генерація команд руху для пристроїв, таких як протези або екзоскелети. Зчитані нейронні сигнали можуть бути аналізовані та інтерпретовані, щоб зрозуміти наміри руху людини. На основі цих намірів, створюються відповідні команди для керування пристроями, що дозволяють особам з руховими обмеженнями виконувати рухи, які раніше були неможливими. Цей підхід відкриває широкі можливості для відновлення функцій тіла та поліпшення якості життя людей.

Окрім керування НКІ також можна використовувати для відновлення когнітивних функцій, наприклад через нейрофідбек або функціональну електростимуляцію.

«Нейрофідбек можна використовувати для покращення когнітивних функцій шляхом надання зворотного зв’язку щодо активності мозку в реальному часі. НКІ використовувалися, щоб допомогти пацієнтам регулювати свою нейронну активність, що може призвести до покращення когнітивних функцій.»[21, 9, 23]

«Функціональна електростимуляція (ФЕС) включає застосування електричних струмів для стимуляції м’язів або нервів, що дозволяє рухатися особам з руховими порушеннями. НКІ можна використовувати для контролю систем ФЕС, дозволяючи пацієнтам ініціювати рух за допомогою сигналів свого мозку.»[18, 20, 16]

Крім керування пристроями, класифіковані сигнали можуть бути використані для вибору команд в інтерфейсах на основі вибору. У таких системах, нейронні сигнали допомагають користувачу взаємодіяти з комп'ютером або іншими електронними пристроями шляхом вибору зі списку опцій. Зчитані сигнали можуть бути аналізовані для визначення намірів користувача, і на основі цих намірів генеруються команди для вибору варіантів в інтерфейсі. Це особливо корисно для людей з важкими руховими обмеженнями, які не можуть фізично взаємодіяти з екранами або клавіатурами.

Результати класифікації нейронних сигналів повинні бути надійними та точними, а процес перетворення сигналів у команди повинен бути швидким та ефективним.

## 2.2 Нейромережі

Основи нейронних мереж

На базовому рівні нейронна мережа - це алгоритмічна система, призначена для розпізнавання закономірностей. Нейронні мережі інтерпретують сенсорні дані за допомогою свого роду механістичного сприйняття, маркуючи і групуючи вихідні дані. Шаблони, що розпізнаються нейронною мережею, є цифровими, містяться у векторах і мають бути перетворені з будь-яких даних реального світу, як-от зображення, мова, текст і тимчасові ряди.

Нейрони: як будівельні блоки

Нейронні мережі складаються з декількох шарів вузлів або "нейронів", які імітують нейрони біологічного мозку. Кожен нейрон отримує вхідні дані, обробляє їх і передає. На етапі обробки нейрон застосовує набір ваг до вхідних даних і обчислює суму ваг. Ця сума передається у функцію активації, яка визначає, чи активується конкретний нейрон і якою мірою, і передає результат на наступний рівень.

Складні шари

Нейронні мережі мають три окремі шари: вхідний, прихований і вихідний. Вхідний шар отримує вихідні дані, вихідний шар ухвалює рішення і робить прогнози щодо вхідних даних, а прихований шар виконує обчислення і передає інформацію від вхідних вузлів до вихідних. Складність нейронної мережі зазвичай полягає в кількості прихованих шарів і кількості вузлів у кожному шарі (рис 2.5).

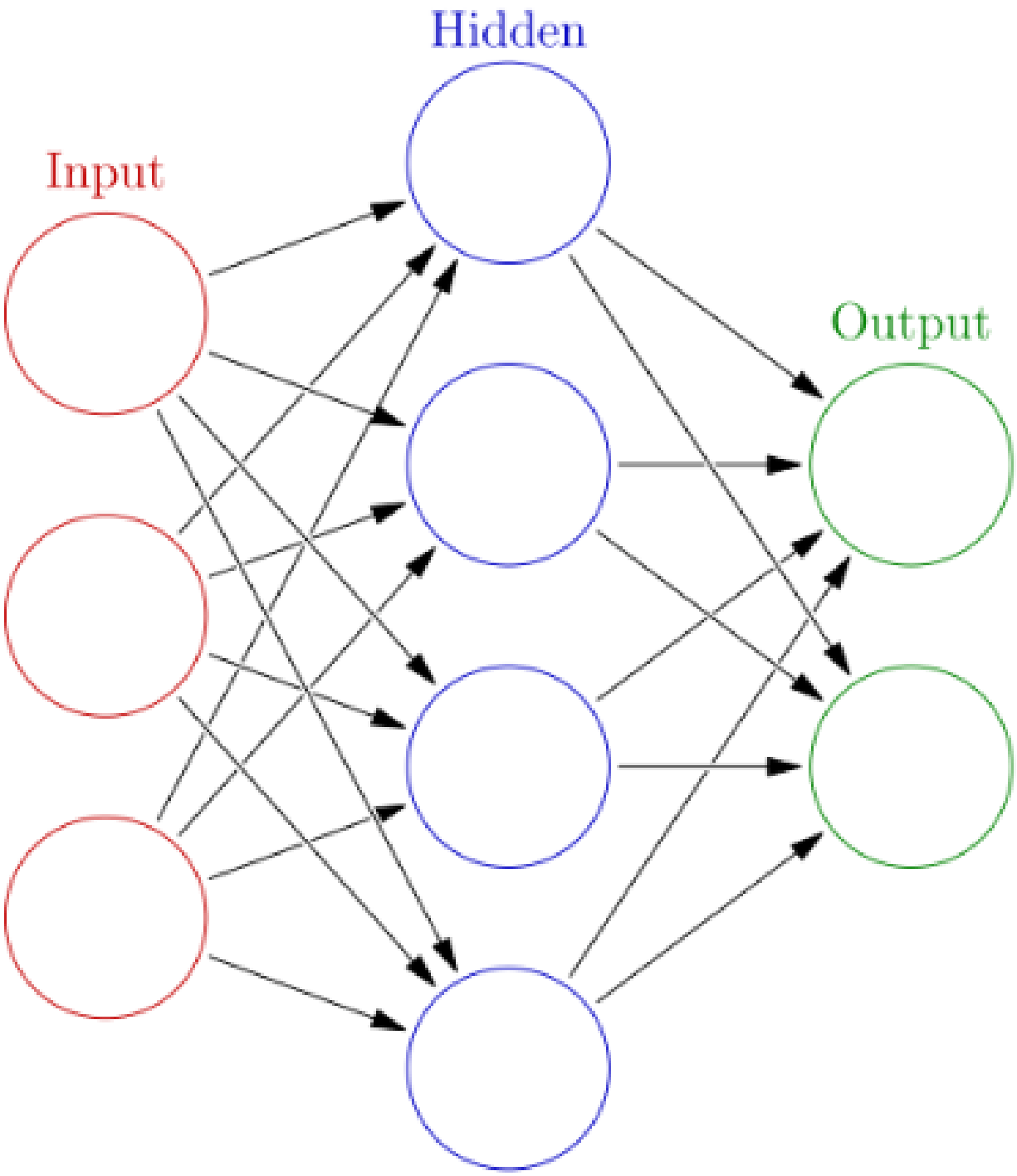



Рисунок 2.5 – Шари нейромережі

Навчання нейронної мережі

Навчання нейронної мережі полягає в коригуванні ваг і зміщень нейронів у відповідь на помилки на виході. Цей процес відомий як "навчання". Якщо мережі дано навчальні дані з відомими виходами, то ваги і зміщення нейронів налаштовують таким чином, щоб мінімізувати різницю між виходом і фактичним

виходом. Для цього зазвичай використовують комбінацію методів, званих зворотним поширенням, і алгоритмів оптимізації, таких як градієнтний спуск.

Глибоке навчання

Коли нейронна мережа має багато прихованих шарів, вона називається глибокою нейронною мережею, а область її вивчення - глибоким навчанням, багатошаровість зображена знизу (рис 2.6).

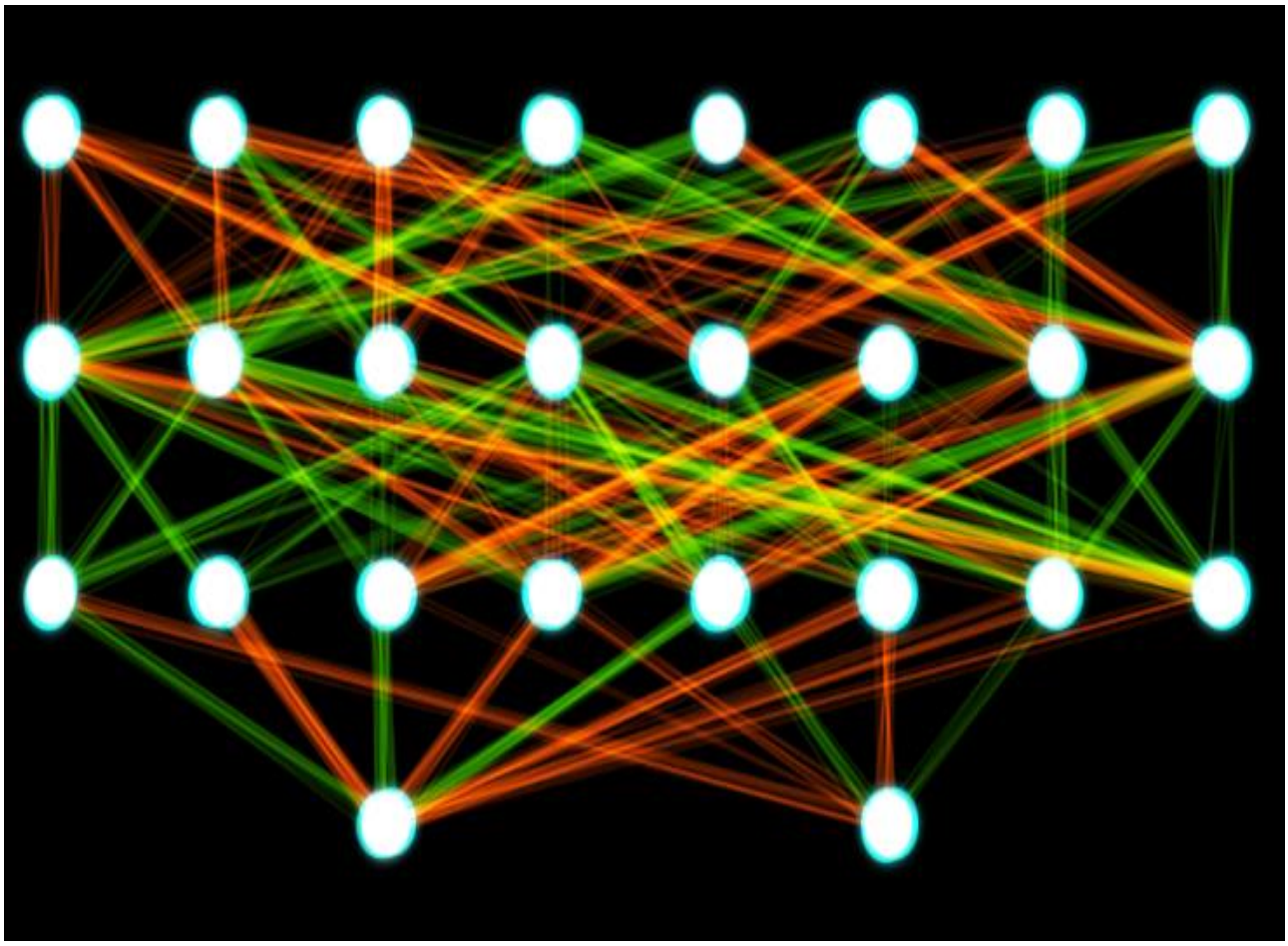


Рисунок 2.6 – Багатошарова нейромережа

Моделі глибокого навчання можуть навчитися представляти світ у вигляді вкладеної ієрархії концепцій і моделей. Наприклад, під час розпізнавання зображень нижні шари можуть розпізнавати краї та кольори, середні шари - форми та об'єкти, а верхні шари - складні сцени.

## Висновки до розділу 2

Очищення сигналу має велику важливість у зчитуванні сигналів мозку людини для управління комп'ютерами. Цей процес дозволяє ефективно відокремити корисну інформацію від шуму та артефактів, що можуть виникати під час зчитування. Чистий сигнал є ключовим елементом для точного та надійного розпізнавання мозкових сигналів та їх подальшого використання для керування комп'ютерними системами.

Очищення сигналу мозку забезпечує високу якість сигнального сигналу, що є необхідним для точного інтерпретування мозкової активності. Видалення шуму та артефактів забезпечує більш точну та надійну інформацію про стан мозку та намір користувача.

Цей процес дозволяє підвищити швидкість та точність інтерфейсу мозок-комп'ютер, забезпечуючи більш точне та швидке розпізнавання команд користувача. Чистий сигнал також зменшує ризик помилкового сприйняття та виконання команд, що є особливо важливим для застосувань, де швидкість і точність є критичними. Нейроним мережам важливо мати гарну якість поступаючим даним, що і забезпечує очищення від шуму.

# РОЗДІЛ 3
# ПРАКТИЧНА РЕАЛІЗАЦІЯ ЗАДАЧІ

## 3.1 Огляд даних біологічних сигналів

Для огляду було обрано набір даних з інтернет ресурсу [3]. «Дані були зібрані у чотирьох осіб (2 чоловіки, 2 жінки) протягом 60 секунд для кожного стану - розслаблений, зосереджений, нейтральний»[3].

Опис роботи з яких було обрано дані «Ця робота спрямована на пошук дискримінативних ознак на основі ЕЕГ та відповідних методів класифікації, які можуть класифікувати патерни мозкових хвиль на основі їх рівня активності або частоти для розпізнавання психічного стану, корисного для людино-машинної взаємодії. Використовуючи пов'язку Muse з чотирма ЕЕГ-датчиками (TP9, AF7, AF8, TP10), ми класифікували три можливі стани, такі як розслаблений, нейтральний і зосереджений, на основі декількох станів розуму, визначених когнітивно-поведінковими дослідженнями. Ми створили набір даних з п'ятьма особами та сеансами тривалістю одна хвилина для кожного класу психічних станів, щоб тренувати та тестувати різні методи. Враховуючи запропонований набір ознак, витягнутих з п'яти сигналів ЕЕГ (альфа, бета, тета, дельта, гамма), ми протестували комбінацію різних алгоритмів вибору ознак і моделей класифікатора, щоб порівняти їхню ефективність з точки зору точності розпізнавання і кількості необхідних ознак. Були проведені різні тести, такі як 10-кратна перехресна перевірка. Результати показують, що при використанні класичних класифікаторів, таких як байєсівські мережі, машини опорних векторів і випадкові ліси, необхідно лише 44 ознаки з набору понад 2100 ознак, що дозволяє досягти загальної точності понад 87%.»[3]

Зображено схему розташування електродів (рис 3.1). Жовтим кольором виділено датчики Muse. Розташування (зелених) використовується як точка відліку для калібрування.

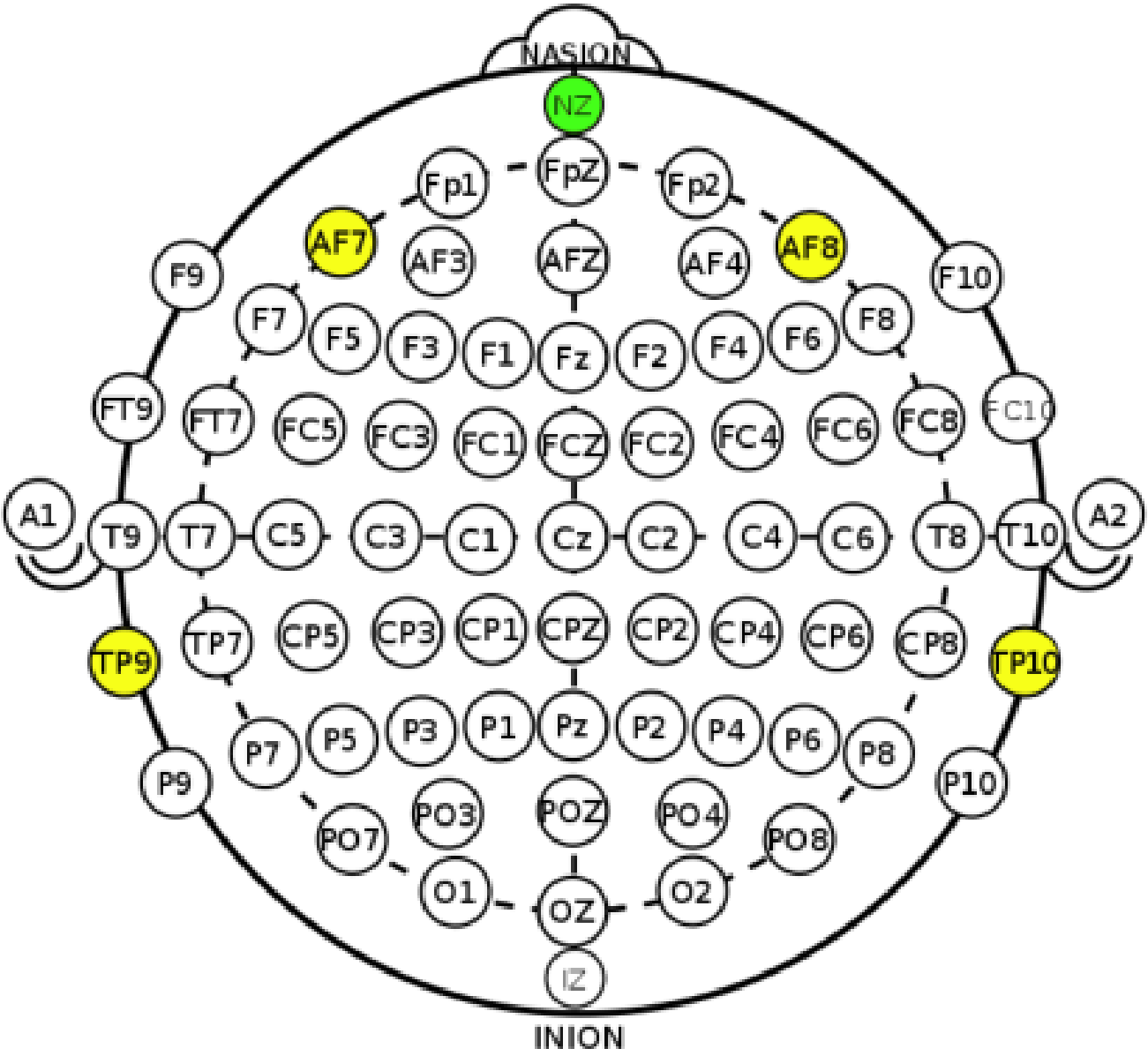



Рисунок 3.1 –Розміщення електродів ЕЕГ.

Розташування електродів ЕЕГ відіграє ключову роль у точному вимірюванні мозкової активності. Правильне розміщення забезпечує деталізовану карту активних зон мозку, що допомагає у діагностиці й наукових дослідженнях. Без цього аналіз може бути неточним.

Далі зображено графік ЕЕГ (рис 3.2).

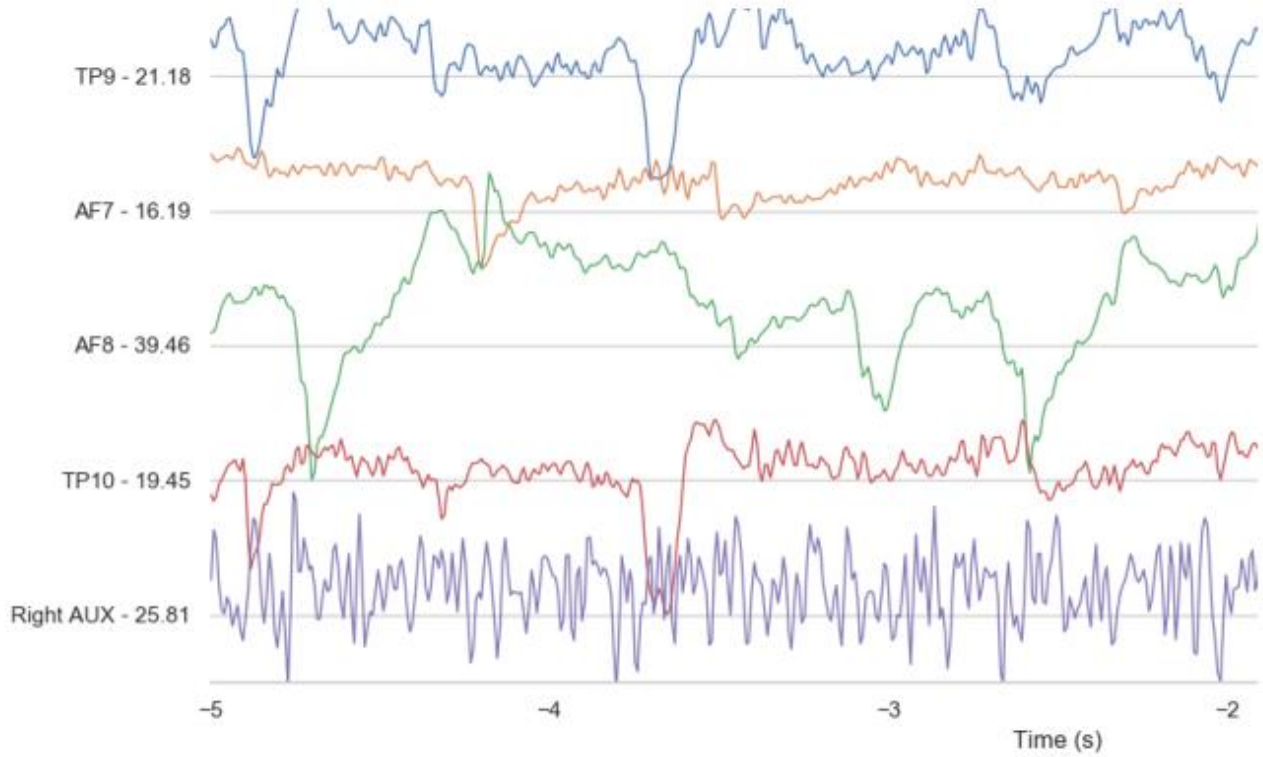



Рисунок 3.2 – Приклад ЕЕГ чотирьох датчиків Muse.

В публікації наведено погляд авторів на прибрання шуму або ж відбір ознак спрямований на виключення даних, які не мають корисного застосування і тільки займають непотрібний обсяг ресурсів. Було створено п'ять наборів даних за допомогою різних алгоритмів. Кожен з них мав ті ж самі точки даних, але з меншою кількістю відібраних атрибутів, визначених алгоритмом. Використані оцінювачі були наступні:

1. «OneR: розраховує помилковість кожного передбачення на основі одного правила й вибирає класифікацію з найнижчим ризиком»[28].

2. «Information Gain: присвоює значення кожному окремому атрибуту, вимірюючи інформаційний приріст щодо класу (різниця ентропії)»[27].

3. «Correlation: вимірює кореляцію між атрибутом і класом за допомогою коефіцієнта Пірсона, що використовується для ранжування вартості атрибутів порівняно з іншими»[17].

4. «Symmetrical Uncertainty: вимірює невизначеність атрибута щодо класу та базує вибір на нижчих значеннях невизначеності»[29].

5. «Evolutionary Algorithm: створює популяцію підмножин атрибутів та ранжує їх ефективність за допомогою функції пристосованості для вимірювання їх передбачальної здатності класу. На кожному поколінні розв'язки розмножуються, а найслабші розв'язки вмирають в турнірі з пристосованістю»[2].

Для проведення дослідження з обробки сигналів мозку, були використані відкриті дані, доступні на публічних інтернет-ресурсах [14]. Large language model (LLM) Alpaca – важливий інструмент передбачення наступного слова користувача для зменшення кількості неправильних відповідей. Детальніше про LLM Alpaca буде розглянуто в розділі 3.3 Налаштування .

Використання відкритих даних сприяє відкритості та реплікації досліджень, оскільки будь-хто може використовувати ті ж дані для проведення подальших досліджень та перевірки результатів.

## 3.2 Технології й бібліотеки використані при розробці

Для розробки програмного застосунку було використано мову програмування Python 3.11.3 в середовищі Visual Studio 2022. Було використано бібліотеки: BrainCode, NumPy, pandas, matplotlib, scikit-learn, PyTorch, CUDA, GIT. Додатко во для розшифрування використовувалась Alpaca large language model (LLM) для розшифрування.

### 3.2.1 Розгляд середовищ програмування

Пропоновані переваги показані на сторінці PyCharm (рис 3.3).

Переваги PyCharm 2023.1.2:

- Підтримка Python: PyCharm є потужним інтегрованим середовищем розробки (IDE), призначеним спеціально для мови програмування Python. Воно надає широкий набір інструментів та функціональності, що сприяють полегшенню розробки проектів на Python.

– Автоматичне завершення коду та відлагодження: PyCharm пропонує інтелектуальне автозавершення коду, що допомагає прискорити процес написання коду та знизити кількість помилок. Він також має потужні інструменти для відлагодження (debugging) коду, які допомагають виявляти та виправляти помилки під час розробки.
– Підтримка віртуальних середовищ: PyCharm дозволяє створювати та керувати віртуальними середовищами Python. Це дозволяє ізолювати проекти та їх залежності, забезпечуючи чистоту та організованість у розробці.
– Засоби роботи з версіями: PyCharm має інтеграцію з такими дійсно популярними системами контролю версій, наприклад такі як Git. Це дозволяє зручно працювати з репозиторіями коду та використовувати функціональність контролю версій безпосередньо з інтерфейсу PyCharm [12].

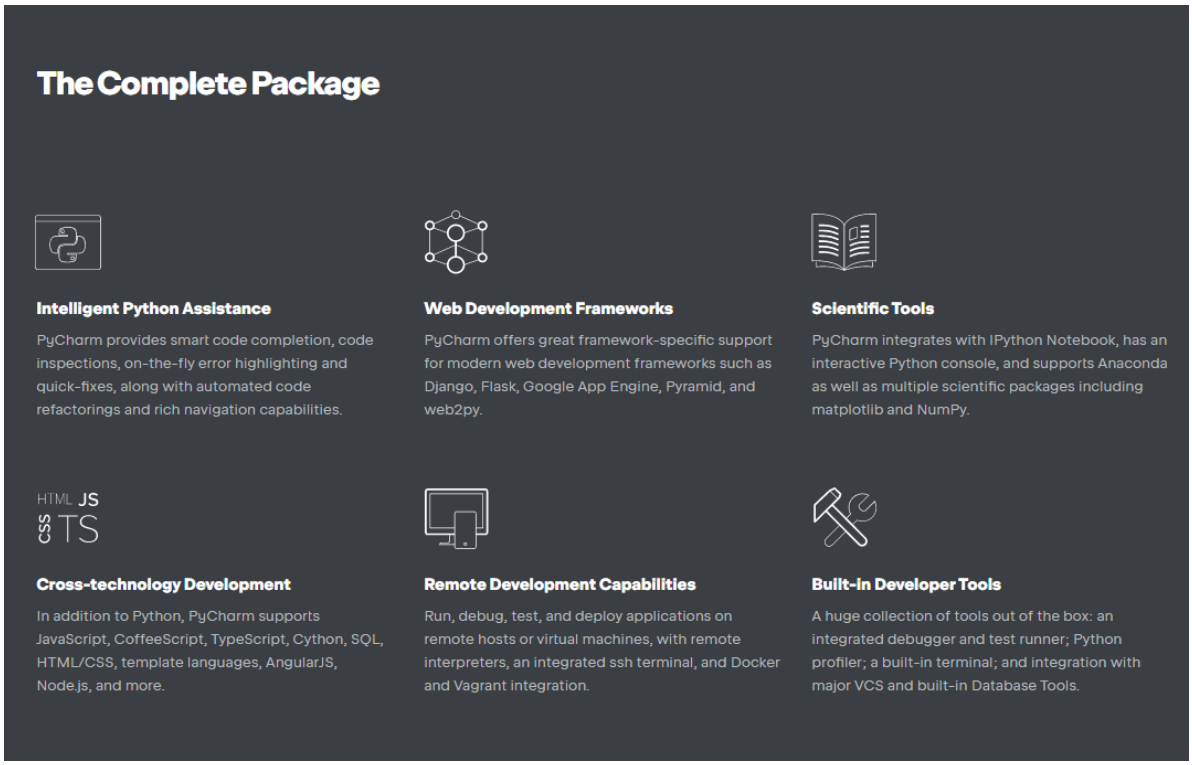



Рисунок 3.3 – Сторінка переваг PyCharm [12].

Пропоновані переваги Visual Studio (рис 3.4).

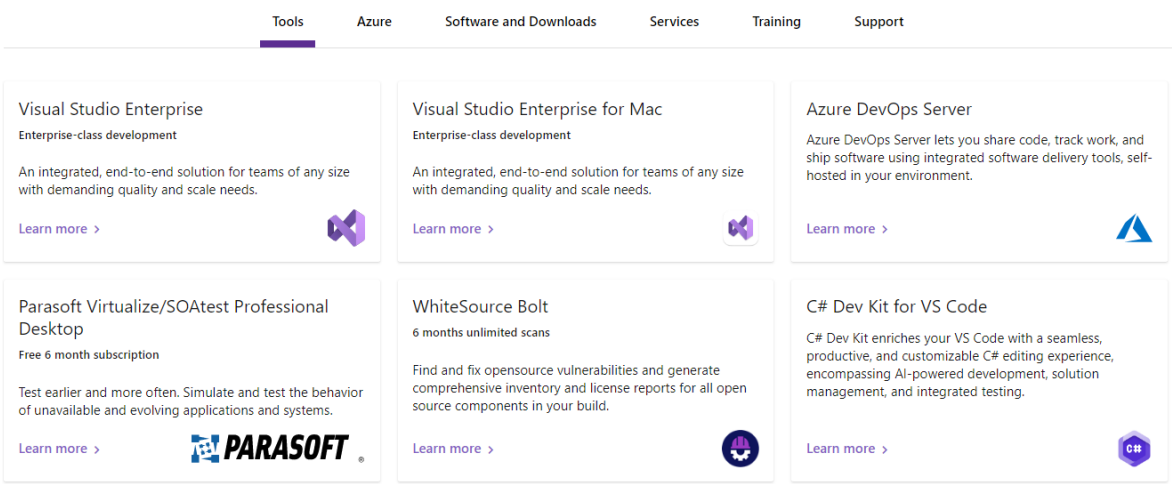



Рисунок 3.4 – Сторінка інструменталльних переваг Visual Studio [15].

Переваги Visual Studio:

- Мультиплатформеність: Visual Studio підтримує розробку на таких платформах що відрізняються, включаючи Windows, macOS та Linux. Це дозволяє розробникам працювати на улюбленій операційній системі та зручно переносити проекти між платформами.
- Розширені можливості: Visual Studio надає широкий набір інструментів та функціональності для розробки різних типів програмних продуктів, включаючи веб-додатки, мобільні додатки, хмарні рішення, ігри та багато іншого. Воно також підтримує різні мови програмування, включаючи C++, C#, Python та інші.
- Інтегроване середовище розробки: Visual Studio надає повноцінне інтегроване середовище розробки (IDE) з розширеними можливостями. Воно має інструменти для автоматичного завершення коду, відлагодження, профілювання, аналізу продуктивності та багато іншого [15].

### 3.2.2 Огляд бібліотек

Braindecode є відкритою бібліотекою Python, в основному використовуваною для декодування сирих електрофізіологічних даних мозку за допомогою моделей глибокого навчання. Бібліотека надає набір інструментів для отримання наборів даних, попередньої обробки даних, візуалізації даних, а також містить реалізації кількох архітектур глибокого навчання. Вона також підтримує аугментацію даних для аналізу електроенцефалографічних (ЕЕГ), електрокортикографічних (ЕКоГ) та магнетоенцефалографічних (МЕГ) даних.

Переваги Braindecode:

- Спеціалізована для електрофізіологічних даних мозку: Бібліотека спеціально розроблена для роботи з такими типами даних, як ЕЕГ, ЕКоГ та МЕГ, що робить її цінним інструментом для нейрофізіологів та дослідників в суміжних галузях.
- Підтримка глибокого навчання: Braindecode містить реалізації кількох моделей глибокого навчання, які можуть бути використані для отримання інсайтів з складних електрофізіологічних даних.
- Інструменти попередньої обробки та візуалізації даних: Braindecode надає інструменти для очищення та візуалізації даних, які є важливими етапами у будь-якому потоці аналізу даних.
- Завантажувачі наборів даних: Бібліотека містить функції для отримання наборів даних, що можуть допомогти дослідникам швидко запуститися зі стандартними джерелами даних.

NumPy, що стоїть за скороченням Numerical Python, є базовою бібліотекою для чисельних обчислень в Python. NumPy надає підтримку для масивів (включаючи багатовимірні масиви) разом з великою кількістю математичних функцій для роботи з цими масивами. Вона має високо оптимізовану продуктивність і надає гнучкий та ефективний об'єкт-масив (ndarray), який є значно більш ефективним і універсальним порівняно зі вбудованими в Python структурами даних, такими як списки. NumPy часто використовується в наукових

обчисленнях, аналізі даних та для будь-якого завдання, що вимагає ефективних чисельних операцій.

Переваги NumPy:

- Ефективні чисельні операції для великих наборів даних
- Підтримка багатовимірних масивів
- Велика кількість доступних математичних функцій
- Основа для багатьох інших бібліотек для аналізу даних та наукових обчислень.

pandas – потужна бібліотека для маніпулювання даними, побудована на базі NumPy. pandas вводить дві нові структури даних в Python – Series і DataFrame, які також побудовані на базі масивів NumPy, що означає їх швидкодію. DataFrame є табличною структурою даних, дуже схожою на те, як дані представлені в Excel або SQL. pandas часто використовується для очищення даних, обробки даних та базового аналізу даних.

Переваги pandas:

- Потужні можливості маніпулювання даними
- Легке опрацювання відсутніх даних
- Автоматичне вирівнювання даних та інтегрована обробка різних форматів даних
- Висока продуктивність злиття та приєднання наборів даних

matplotlib – це бібліотека для побудови графіків в Python. Вона надає об'єктно-орієнтований API для вбудовання графіків у програми, використовуючи універсальні GUI-набори інструментів, такі як Tkinter, wxPython, Qt або GTK. matplotlib також є популярним вибором для створення статичних, анімованих та інтерактивних візуалізацій в Python. Приклад викистання бібліотеки для створення графіку (рис 3.5).

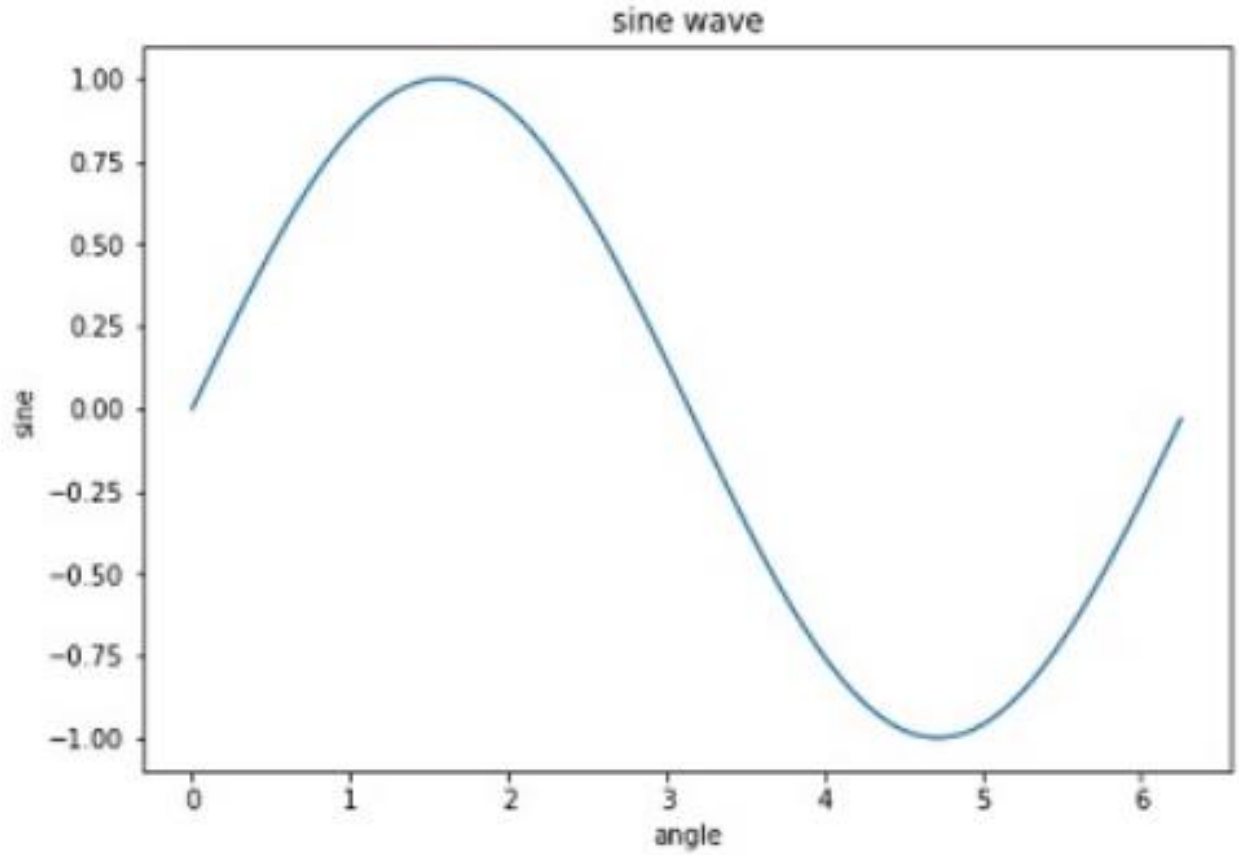



Рисунок 3.5 – Приклад використання бібліотеки

Переваги matplotlib:

- Можливість створення різноманітних статичних, анімованих та інтерактивних графіків
- Налаштування та розширення – ви можете налаштувати кожний аспект графіка
- Хороша сумісність з багатьма операційними системами та графічними движками

scikit-learn – одна з провідних бібліотек для машинного навчання в Python. Вона надає набір алгоритмів навчання з учителем та без учителя через однорідний інтерфейс. scikit-learn також надає інструменти для підгонки моделей, попередньої обробки даних, вибору та оцінки моделей, а також багато інших утиліт.

Переваги scikit-learn:

- Великий вибір алгоритмів машинного навчання

- Інструменти для оцінки та вибору моделей
- Побудована на основі NumPy, SciPy та matplotlib, що полегшує інтеграцію з Python-екосистемою для наукових обчислень
- Велика документація та підтримка спільноти

PyTorch є потужною бібліотекою машинного навчання та обчислень наукового рівня, яка знайшла широке застосування в галузі глибинного навчання та штучного інтелекту. Заснована на мові програмування Python, PyTorch надає ефективні інструменти для розробки та навчання нейромереж зі зручним інтерфейсом та гнучкістю.

Однією з найважливіших переваг PyTorch є його динамічний граф обчислень. Він дозволяє розробникам будувати та модифікувати нейромережі на льоту, що робить його особливо потужним для дослідницьких робіт та прототипування нових моделей. За допомогою PyTorch, студент може легко експериментувати з різними архітектурами, шарами та гіперпараметрами, швидко спробувати нові ідеї та вдосконалювати навчання моделей.

PyTorch також пропонує багато готових до використання функцій та модулів, які спрощують розробку та навчання нейромереж. Вона включає реалізацію популярних архітектур нейромереж, таких як CNN (згорткові нейромережі), RNN (рекурентні нейромережі) та Transformer, а також набір вбудованих функцій активації, функцій втрат та оптимізаторів. Це дозволяє студенту швидко побудувати та навчити нейромережу для рішення різноманітних задач, включаючи зображення, текст та часові послідовності.

Підтримка PyTorch для обчислень на графічних процесорах (GPU) через CUDA також є великою перевагою. Використання GPU для обчислень дозволяє значно прискорити процес навчання нейромереж та забезпечити високу швидкодію. Завдяки PyTorch, студент може використовувати потужні обчислювальні ресурси GPU для тренування нейромереж та отримання результатів швидше.

Іншою важливою особливістю PyTorch є його екосистема, яка включає різноманітні інструменти та бібліотеки для розширення можливостей. Наприклад, бібліотека TorchVision надає засоби для роботи зі зображеннями, включаючи завантаження даних, попередню обробку та аугментацію. Бібліотека TorchText допомагає в роботі з текстовими даними, включаючи токенізацію, побудову словників та векторне представлення слів. Ці бібліотеки роблять PyTorch більш комплексним та потужним інструментом для обробки різних типів даних.

PyTorch також має активну спільноту розробників та документацію, що полегшує навчання та вирішення проблем. На офіційному веб-сайті PyTorch доступні численні ресурси, включаючи документацію, приклади коду та навчальні матеріали. Крім того, існує велика кількість відкритих джерел, блогів та форумів, де студент може знайти відповіді на свої запитання та отримати додаткову підтримку від спільноти.

В цілому, PyTorch є потужним інструментом для розробки та навчання нейромереж з високою гнучкістю та продуктивністю. Його динамічний граф обчислень, підтримка GPU, багатофункціональність та активна спільнота роблять його ідеальним вибором для студентів, які цікавляться глибинним навчанням та застосуванням штучного інтелекту. З використанням PyTorch студент може зосередитись на дослідженні та розробці інноваційних моделей нейромереж з впевненістю в їхній ефективності та розширених можливостях.

### 3.2.3 Огляд додаткових інструментів

CUDA (Compute Unified Device Architecture) – це платформа для паралельних обчислень на графічних процесорах (GPU), розроблена компанією NVIDIA. Ця технологія надає студентам та дослідникам потужний інструментарій для прискорення обчислень та виконання складних завдань в галузі наукових досліджень, штучного інтелекту, комп'ютерного зору та багатьох інших. Додатки зображено нижче (рис. 3.6)

s

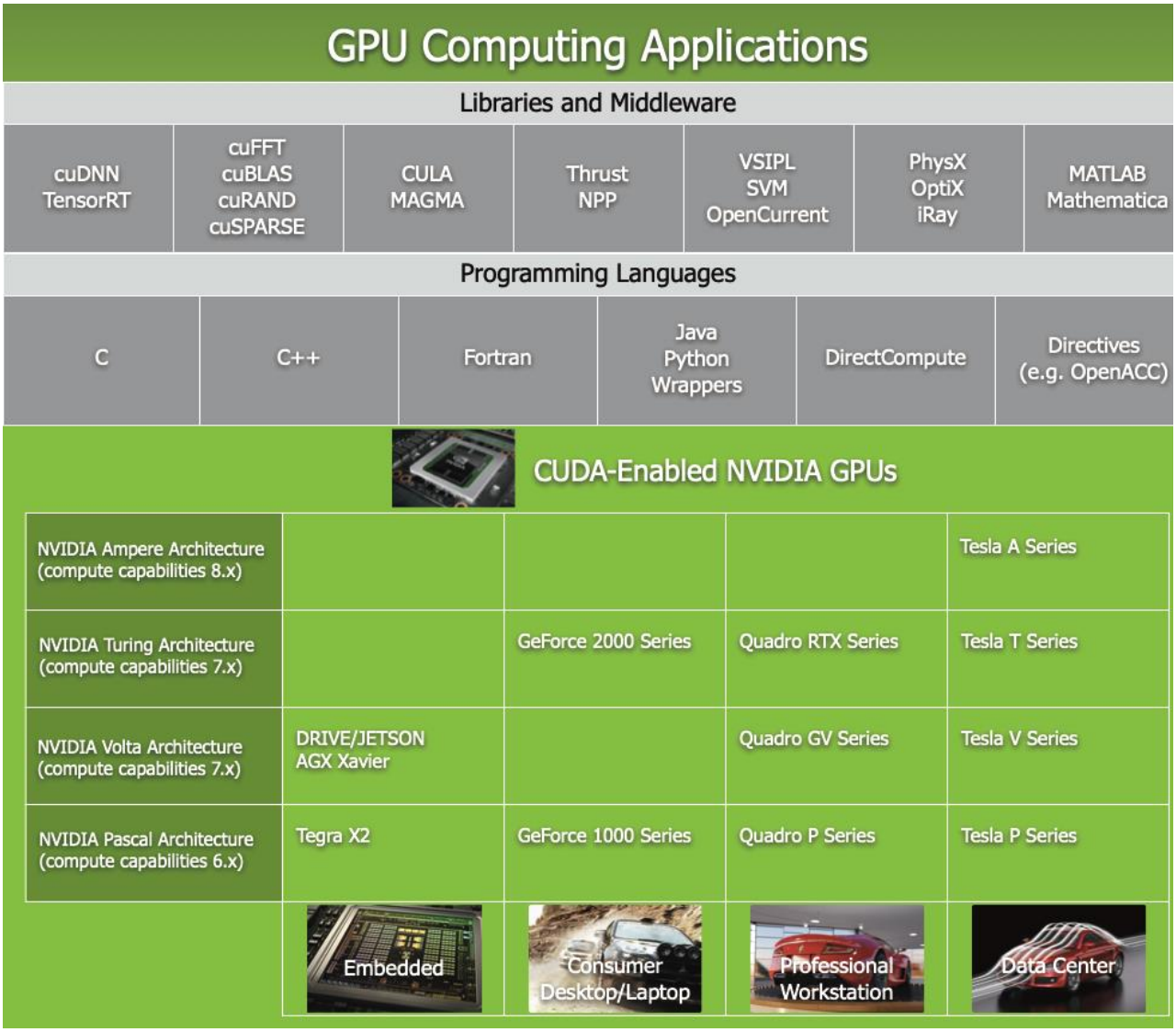



Рисунок 3.6 – Додатки для обчислень на GPU. CUDA розроблена для підтримки різних мов та інтерфейсів прикладного програмування.

Однією з основних переваг CUDA є його здатність використовувати масиви великої кількості ядер графічного процесора для паралельних обчислень. Замість традиційних послідовних обчислень на центральному процесорі (CPU), CUDA дозволяє розпаралелити обчислення та виконувати їх одночасно на багатьох ядрах GPU. Це дозволяє суттєво збільшити продуктивність та швидкість обчислень.

CUDA має широкий набір функцій та інструментів для розробки програм, що використовують GPU. На першому етапі, студенти можуть скористатися

CUDA C/C++, спеціальною мовою програмування для CUDA, для написання паралельних обчислювальних ядер. Це дозволяє студентам ефективно використовувати ресурси GPU та створювати швидкі та ефективні програми.

Крім CUDA C/C++, CUDA надає також підтримку інших мов програмування, включаючи Python та Fortran. Це відкриває багато можливостей для студентів, які володіють різними мовами програмування та бажають використовувати CUDA у своїх проектах.

Окрім того, CUDA надає доступ до багатьох бібліотек та інструментів, що допомагають розробляти швидкі та ефективні програми. CUDA-сумісні бібліотеки, такі як cuDNN (для глибинного навчання), cuBLAS (для лінійної алгебри), cuSPARSE (для розріджених матриць) та багато інших, розширюють можливості CUDA та допомагають студентам ефективно вирішувати різноманітні завдання.

Однією з головних переваг CUDA є його широке застосування в галузі глибинного навчання та нейромереж. Багато фреймворків глибинного навчання, таких як PyTorch, TensorFlow та Keras, мають підтримку CUDA, що дозволяє використовувати потужність GPU для тренування та застосування нейромереж. Студенти можуть використовувати CUDA разом з цими фреймворками, щоб ефективно розробляти та оптимізувати свої моделі нейромереж.

CUDA також надає інструменти для профілювання та оптимізації програм, що дозволяє студентам виявляти й усувати швидкісні та ефективність обчислень. Відладка та оптимізація коду CUDA допомагають студентам отримати максимальну продуктивність використання ресурсів GPU та покращити швидкість своїх програм.

Загалом, CUDA є потужним інструментом для розпаралелення обчислень та використання GPU в наукових дослідженнях, глибинному навчанні та інших областях. Його можливості паралельних обчислень, широкий вибір мов програмування, багатофункціональність та інструменти для профілювання роблять CUDA незамінним інструментом для студентів, які прагнуть використовувати потужність GPU для розв'язання складних завдань обчислень.

GIT – це розподілена система керування версіями, що надає студентам потужний інструментарій для керування змінами у кодовій базі їхнього проекту. GIT є однією з найпопулярніших систем керування версіями та забезпечує ефективний та надійний спосіб керування розвитком програмного продукту.

Однією з основних переваг GIT є його розподілена природа. Кожен студент має повний доступ до локальної копії кодової бази, що дозволяє працювати незалежно від інших учасників команди. Це означає, що студент може робити зміни, створювати гілки та експериментувати з кодом без впливу на роботу інших учасників. Після завершення роботи студент може злити свої зміни з основною гілкою, забезпечуючи цілісність та актуальність кодової бази.

GIT надає також багато інструментів для спільної роботи команди. Студенти можуть створювати гілки для окремих функціональностей або проблем, що дозволяє розробляти різні функціональності паралельно та без конфліктів. Використовуючи можливості злиття та розв'язання конфліктів, команда може ефективно поєднувати свою роботу та долучати нові функції до основного коду.

GIT забезпечує повну історію змін, що дозволяє студентам відстежувати всі зміни, внесені в кодову базу. Це дозволяє відновлювати попередні версії коду, відслідковувати хто та коли вніс зміни, а також вирішувати проблеми, які виникають під час розробки. Крім того, GIT забезпечує можливість створювати теги та випускати версії програмного продукту, що полегшує організацію релізів та розгортання.

GIT також надає інтеграцію з популярними сервісами хостингу, такими як GitHub, GitLab та Bitbucket. Це дозволяє студентам зберігати та спільно працювати над своїми проектами в хмарному середовищі, отримувати зворотний зв'язок від спільноти, проводити рецензії коду та багато іншого.

У підсумку, GIT є незамінним інструментом для керування версіями коду та спільної роботи команди. Його розподілена природа, історія змін, можливості спільної роботи та інтеграція з хостинг-сервісами роблять GIT ідеальним

вибором для студентів, які працюють над програмними проектами та прагнуть до організації та ефективного керування своїм кодом.

### 3.2.4 Додаткові бібліотеки

- accelerate – це бібліотека, яка допомагає запускати ваш скрипт тренування PyTorch на будь-якому типі пристрою, такому як одна або кілька графічних процесорів (GPU) або спеціалізовані процесори для штучного інтелекту (TPU), з або без змішаної точності1.
- appdirs – це модуль, який допомагає визначити відповідні каталоги, специфічні для платформи, для збереження даних користувача, конфігураційних файлів, кеш-файлів тощо2.
- loralib – це модуль, який надає низькорівневий інтерфейс для зв'язку з пристроями LoRa за допомогою Python.
- bitsandbytes – це бібліотека, яка реалізує швидкі та ефективні оптимізатори для PyTorch з використанням стиснення 8 біт.
- black[jupyter] – це додаткова вимога для black, яка дозволяє форматувати блокноти Jupyter.
- datasets – це бібліотека, яка надає простий доступ до широкого спектру наборів даних і метрик для обробки природної мови (NLP).
- fire – це бібліотека, яка автоматично генерує інтерфейси командного рядка (CLIs) з будь-якого об'єкту Python.
- transformers>=4.28.0 – це вимога для встановлення пакету transformers з мінімальною версією 4.28.0. transformers – це бібліотека, яка надає сучасні моделі та інструменти для розуміння та генерації природної мови.
- sentencepiece – це бібліотека, яка реалізує неспрямовану токенізацію та детокенізацію тексту, головним чином для систем генерації тексту на основі нейромереж, де розмір словника передбачається перед навчанням нейромережі.

- gradio – це бібліотека, яка дозволяє швидко створювати настроювані компоненти користувацького інтерфейсу (UI) навколо ваших моделей машинного навчання. Приклад використання бібліотеки (рис 3.7).

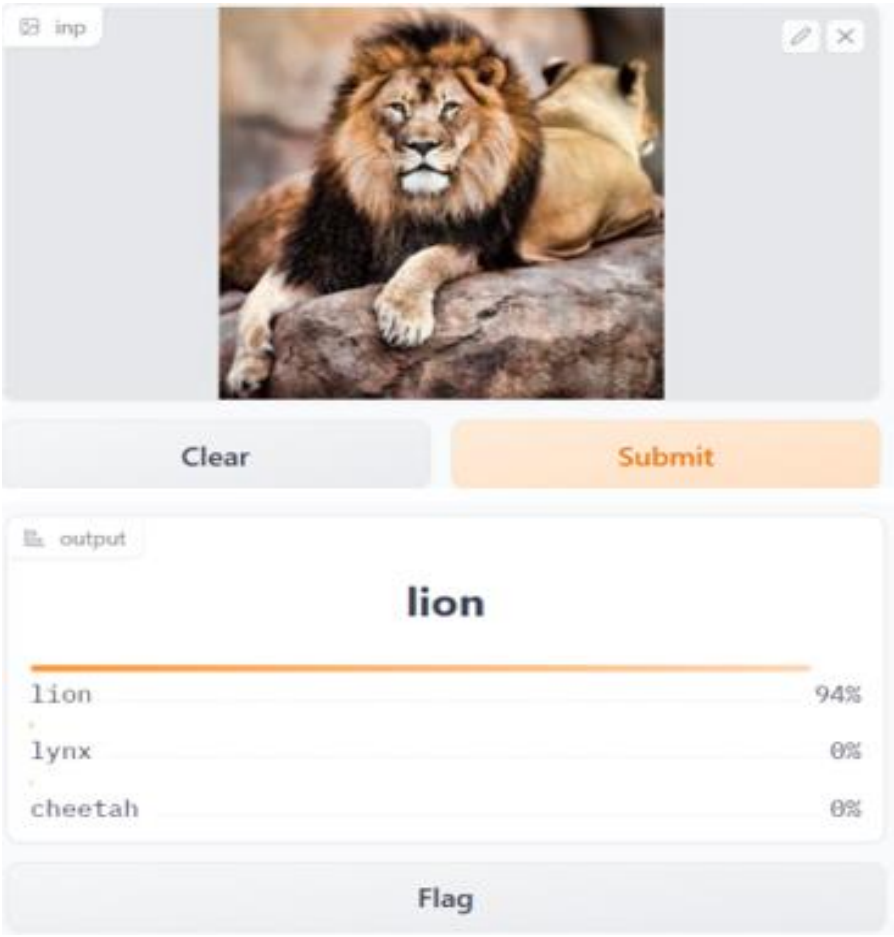



Рисунок 3.7 – Приклад використання бібліотеки gradio.

### 3.3 Налаштування LLM

Для вирішення специфічної задачі мною було вибрано LLM ALPACA, яке є варіантом великої мовної моделі, спеціалізованої на виконанні складних завдань. Однією з переваг LLM ALPACA є те, що модель може функціонувати ефективно, використовуючи процесор замість відеокарти. Це може бути повільніше в термінах обчислювальної швидкості, але це відмінний варіант для тих, хто не має доступу до потужних графічних адаптерів. Також, LLM ALPACA

забезпечує гнучкість і високу продуктивність у багатьох доменах, що робить її відмінним інструментом для обробки природної мови, генерації тексту, машинного перекладу та інших задач. Використання цієї моделі може допомогти оптимізувати ресурси і досягти високої якості розв'язань. (рис 3.8).

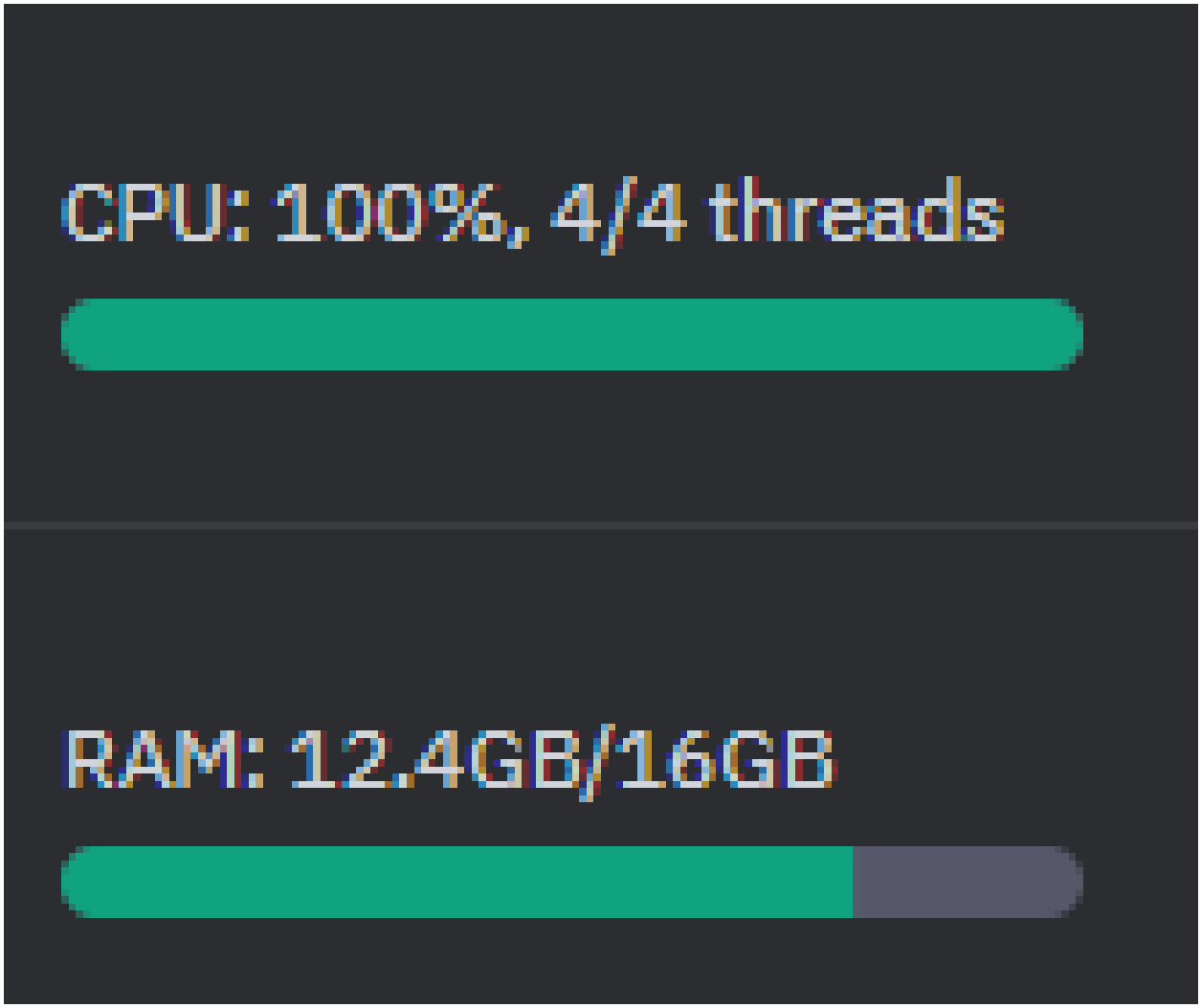



Рисунок 3.8 – Використання моделлю потужностей

Large language models (LLM) є потужним інструментом для обробки природної мови (NLP), який може генерувати текст на різні теми та стилі. Однак багато сучасних LLM мають напрямлену здатність виконувати завдання за інструкціями користувача, наприклад, писати код, створювати зображення або підбирати інформацію. Це ускладнює їх застосування для практичних цілей та вимагає спеціального навчання або адаптації до конкретних доменів.

LLM ALPACA є моделлю мови, яка може виконувати різні складні завдання за інструкціями користувача. Вона була розроблена дослідниками зі Стенфордського університету на основі моделі LLaMA від Meta AI. LLaMA є ефективною та високоякісною моделлю мови, яка перевершує багато комерційних LLM на ряді стандартних бенчмарків NLP. Однак LLaMA не оптимізована для виконання завдань за інструкціями та має обмежений доступ для наукових та комерційних цілей.

Для покращення здатностей LLaMA до виконання завдань за інструкціями дослідники використали метод само-навчання (self-instruct), який полягає у генерації даних для навчання моделі на основі існуючих LLM. Вони використали модель text-davinci-003 від OpenAI для створення 52 тисяч прикладів виконання різних завдань за інструкціями. Потім вони додали цей набір даних до набору даних для передтренування LLaMA та дотренували модель на ньому. Результатом є модель LLM ALPACA, яка може виконувати складні завдання за інструкціями користувача.

LLM ALPACA має багато переваг порівняно з іншими LLM. По-перше, вона є ефективною та якісною, оскільки базується на LLaMA, яка перевершує багато комерційних LLM на ряді стандартних бенчмарків NLP. По-друге, вона є гнучкою та універсальною, оскільки може виконувати розмаїття завдань за інструкц ями користувача, наприклад, писати код, створювати зображення або генерувати текст.

Однак LLM ALPACA також має обмеження та проблеми. По-перше, вона не є безпечною та безшкідною, оскільки не була спеціально оптимізована для запобігання генерації хибної інформації, соціальних стереотипів або токсичної мови. По-друге, вона не є повністю вільною та відкритою, оскільки базується на LLaMA, яка має некомерційну ліцензію, та на text-davinci-003, яка має обмеження на конкуруюче використання. По-третє, вона не є стабільною та надійною, оскільки залежить від якості і актуальності даних для навчання та може мати помилки або непередбачувану поведінку.

У цій дипломній роботі досліджується LLM ALPACA як модель мови для виконання складних завдань.

### 3.3.1 Встановлення інструментів розробки

Встановлення додаткових інструментів:

Встановлення CUDA (рис. 3.9)

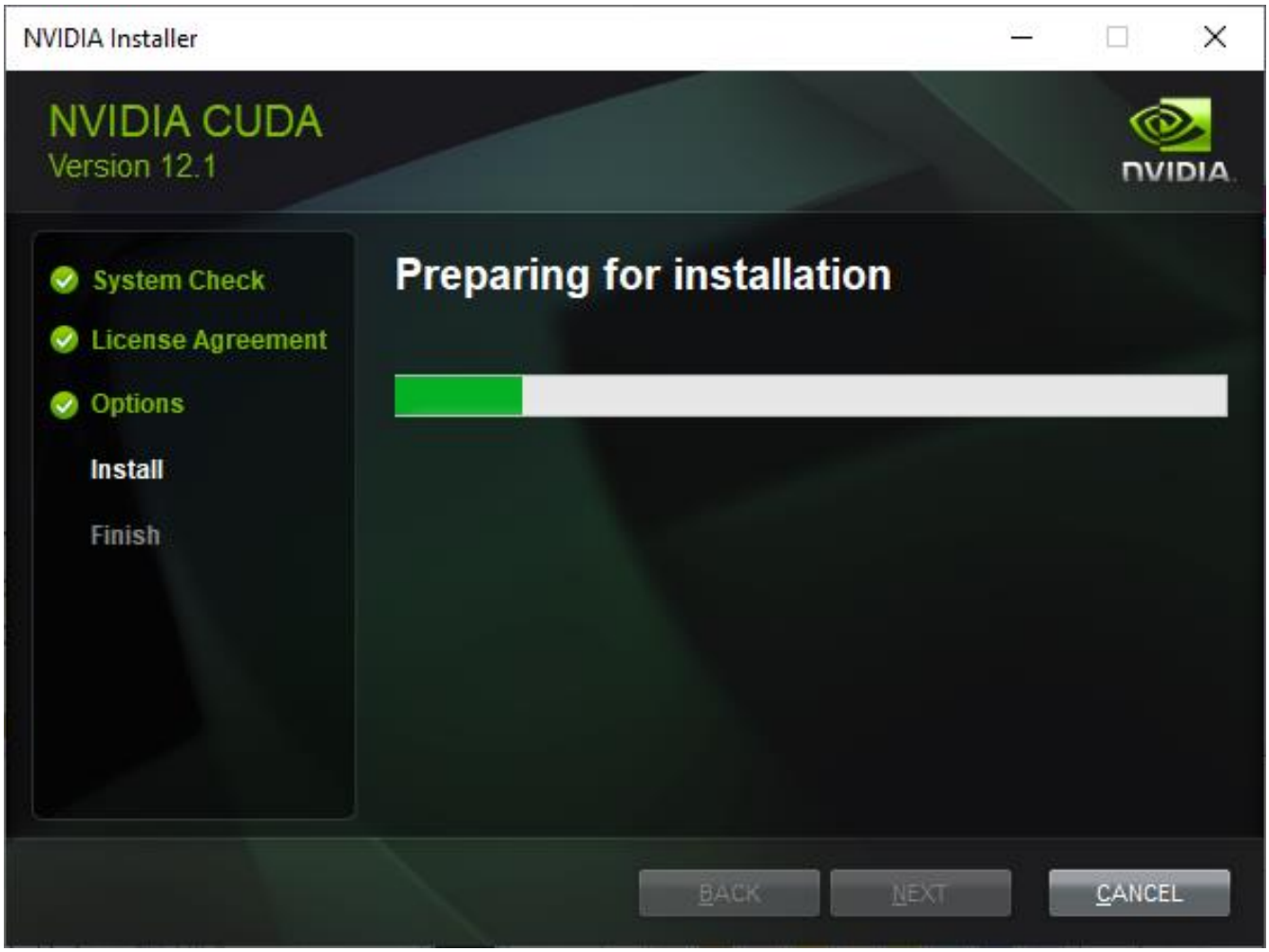



Рисунок 3.9 – Встановлення CUDA

Встановлення Node.js (рис. 3.10). Node.js дозволяє підключати зовнішні бібліотеки та модулі, що спрощує інтеграцію з інструментами. Багато популярних фреймворків та пакетів, таких як Express.js, Socket.IO та MongoDB, були розроблені для Node.js, сприяючи швидкій розробці та розширенню додатків.

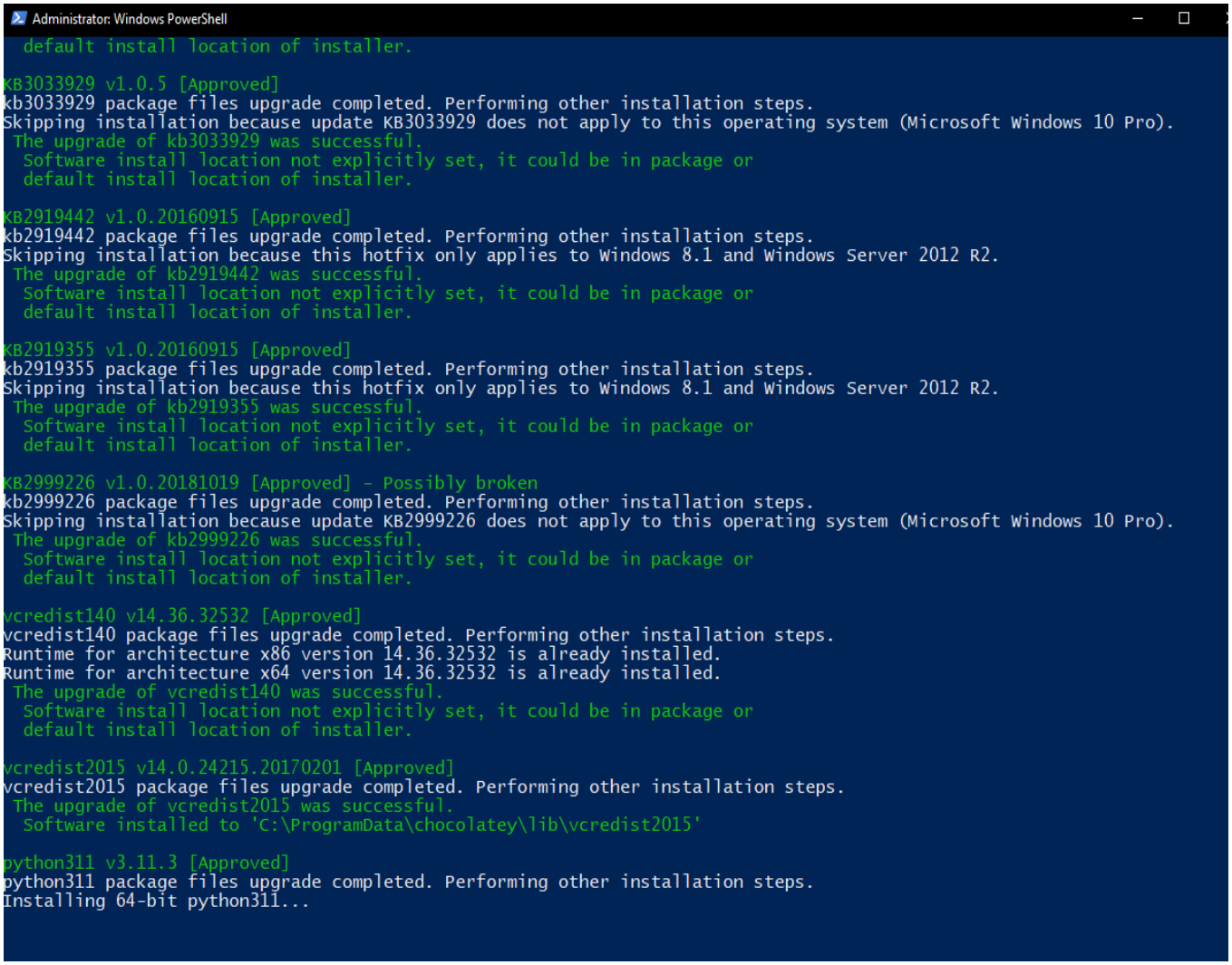



Рисунок 3.10 – Процес встановлення Node.js

Встановлення LLM Alpaca на пристрій:

- Скачати й встановити CMake: [4]
- Скачати й встанговити git. Додатково використовуючи GUI [10]
- Склонувати репозиторій. https://github.com/antimatter15/alpaca.cpp.
- Відкрити Windows Terminal клонованому ропозиторії.
- Запустити команди одна за одною:
    - cmake .
    - cmake --build . --config Release
- Перенести ваги файлу як ggml-alpaca-7b-q4.bin основну директорію.
- У вікні terminal window, запустити команду

  .\Release\chat.exe

Робота програми зображена (рис 3.11 )

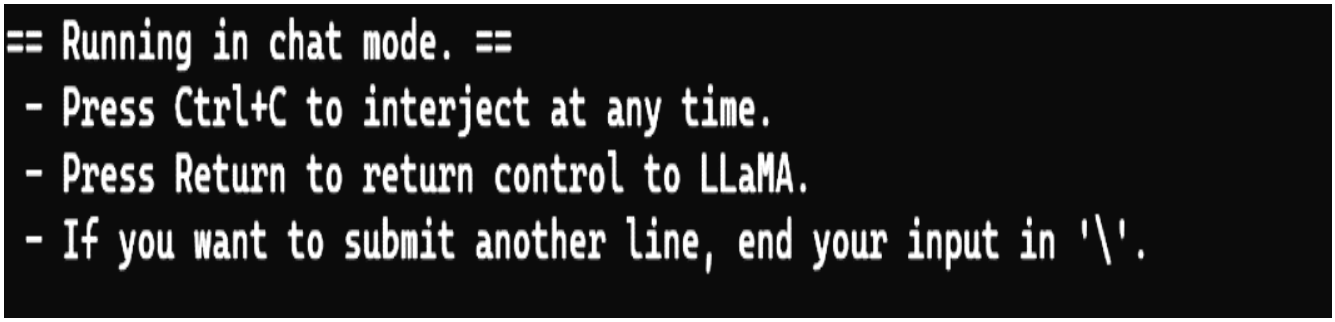



Рисунок 3.11 – Базовий термінал Alpaca

Голвна перевага цієї версії LLM це є використання CPU що дає стабільний, хоч і повільний результат.

### 3.3.2 Fine-tuning дата сету

Для подальшого кроку необхідно провести fine-tuning дата сету Alpaca для переведення сигналів мозку в текст. Для цього необхідно взяти дані з попереднього кроку.

Існубть наступні варіанти:

- Перенесення навчання (Transfer Learning): використовує ваги та архітектуру передтренованої моделі для нової задачі або домену. Передтренована модель зазвичай навчається на великому, загальному наборі даних, і підхід з перенесенням навчання дозволяє ефективно та ефективно адаптувати модель до конкретних задач або доменів.
- Послідовне докладне налаштування (Sequential Fine-tuning): метод, при якому передтренована модель докладно налаштовується на кілька пов'язаних задач або доменів послідовно. Це дозволяє моделі вивчати більш тонкі та складні мовні шаблони для різних задач, що призводить до кращої узагальненості та продуктивності.
- Задачно-специфічне докладне налаштування (Task-specific Fine-tuning): метод, при якому передтренована модель детально налаштовується на конкретну задачу або домен, використовуючи задачно-специфічний

набір даних. Цей метод вимагає більше даних та часу, ніж перенесення навчання, але може призвести до вищої продуктивності на конкретній задачі.

- Навчання адаптерів (Adapter Training): метод, який передбачає навчання легковажних модулів, які включаються до передтренованої моделі, що дозволяє детально налаштувати модель для конкретної задачі, не впливаючи на початкову продуктивність моделі на інших задачах

Основна відмінність між методами детального налаштування та few-shot learning полягає в кількості задачно-специфічних даних, необхідних для адаптації моделі до нової задачі або домену. Методи детального налаштування вимагають помірної кількості задачно-специфічних даних для оптимізації продуктивності моделі, тоді як методи few-shot learning можуть адаптувати моделі до нових задач або доменів лише з кількома позначеними прикладами. Вибір – Task-specific Fine-tuning.

Встановлення xturning pip (рис 3.12 )

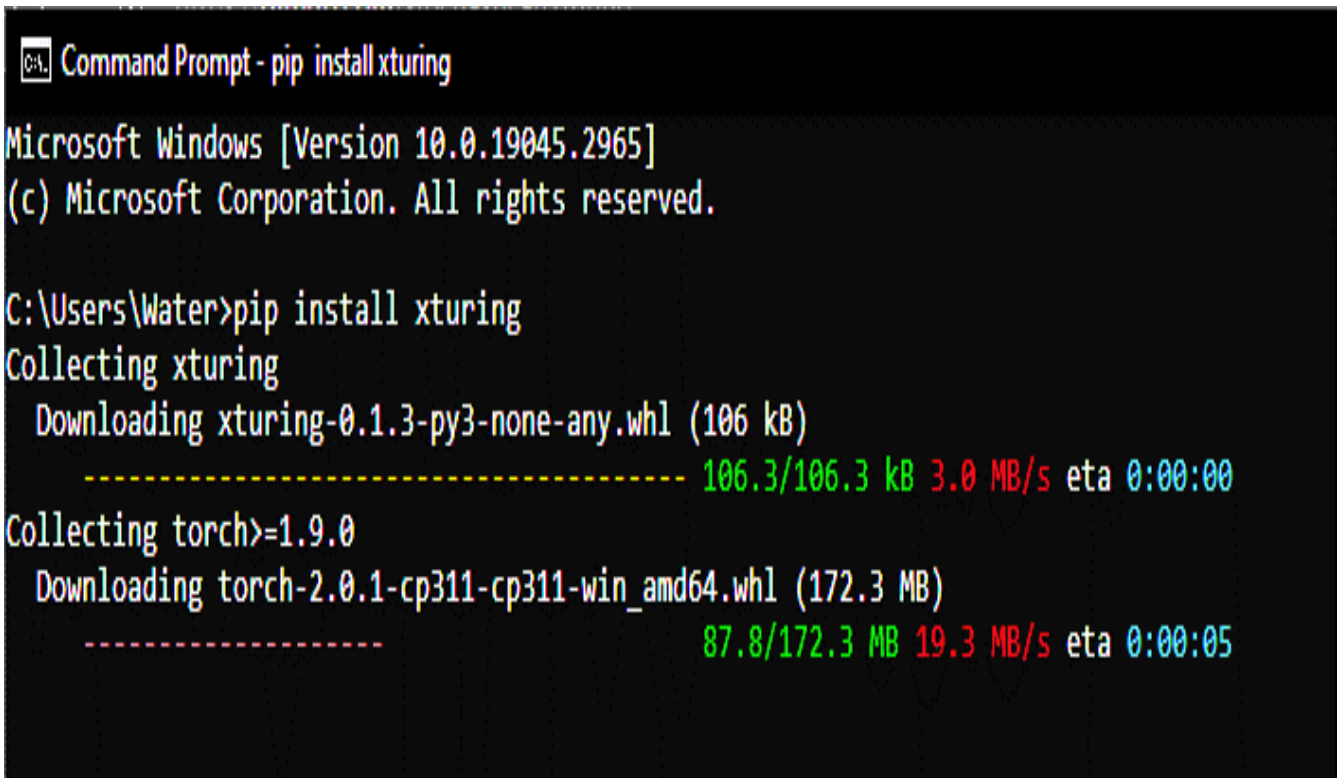



Рисунок 3.12 – Процесс встановлення xturning

Код приготування датасету (рис. 3.13).

```
import json

from datasets import Dataset, DatasetDict

# Convert the alpaca JSON dataset to HF format


# Right now only the HuggingFace datasets are supported, that's why the JSON Alpaca dataset
# needs to be converted to the HuggingFace format. In addition, this HF dataset should have 3 columns for instruction finetuning: instruction, text and target.
def preprocess_alpaca_json_data(alpaca_dataset_path: str):
    """Creates a dataset given the alpaca JSON dataset. You can download it here: https://raw.githubusercontent.com/tatsu-lab/stanford_alpaca/main/alpaca_data.json

    :param alpaca_dataset_path: path of the Alpaca dataset
    """
    alpaca_data = json.load(open(alpaca_dataset_path))
    instructions = []
    inputs = []
    outputs = []

    for data in alpaca_data:
        instructions.append(data["instruction"])
        inputs.append(data["input"])
        outputs.append(data["output"])

    data_dict = {
        "train": {"instruction": instructions, "text": inputs, "target": outputs}
    }

    dataset = DatasetDict()
    # using your `Dict` object
    for k, v in data_dict.items():
        dataset[k] = Dataset.from_dict(v)

    dataset.save_to_disk(str("./alpaca_data"))
```

Рисунок 3.13 – Код приготування датасету [8]

Обрані дані необхідно підвести під необхідний вид для подальшого використання в fine-tuning дата сету (рис 3.14)

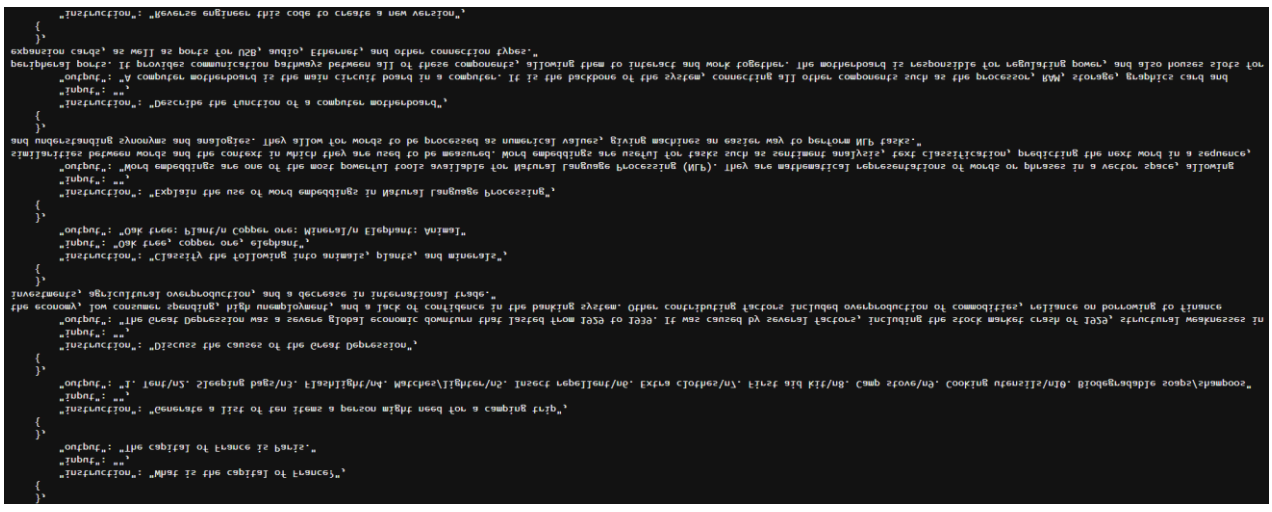


Рисунок 3.14 – Організований вид даних.

З джерела було взято набір даних зчитанних й оброблених з сигналів мозку й закріплено до значення (рис 3.15 ) [1]. Це є набір даних фМРТ під час пасивного прослуховування природної мови. Набір даних містить попередньо оброблені дані, вирівняні транскрипти стимулу, відкоректовані вручну поверхні для кожного суб'єкта та інші файли для аналізу та візуалізації (рис 3.15).

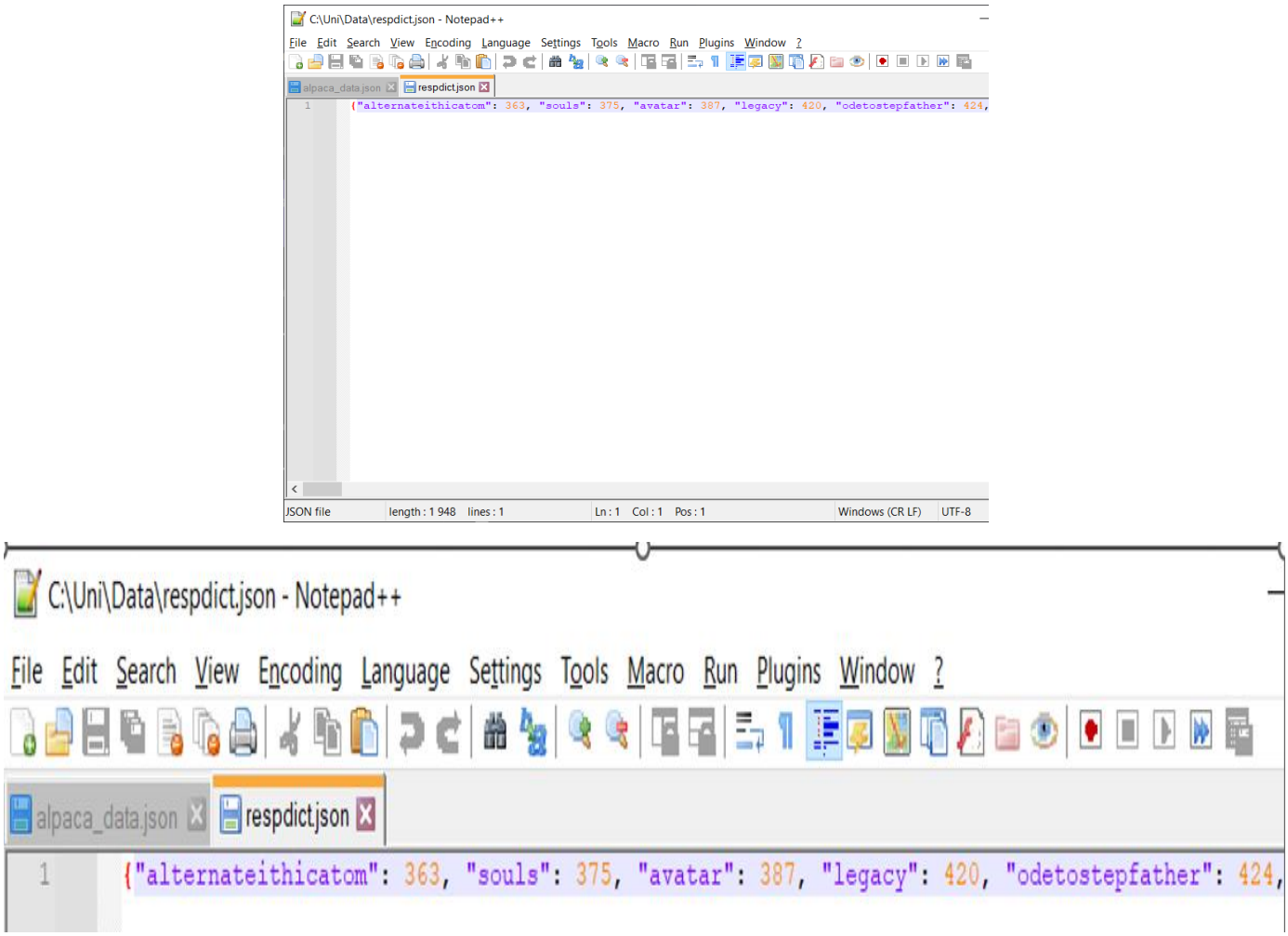



Рисунок 3.15 – Приклад зовнішнього виду файлу з даними.

Ці дані були організовані необхідним видом (рис 3.16)

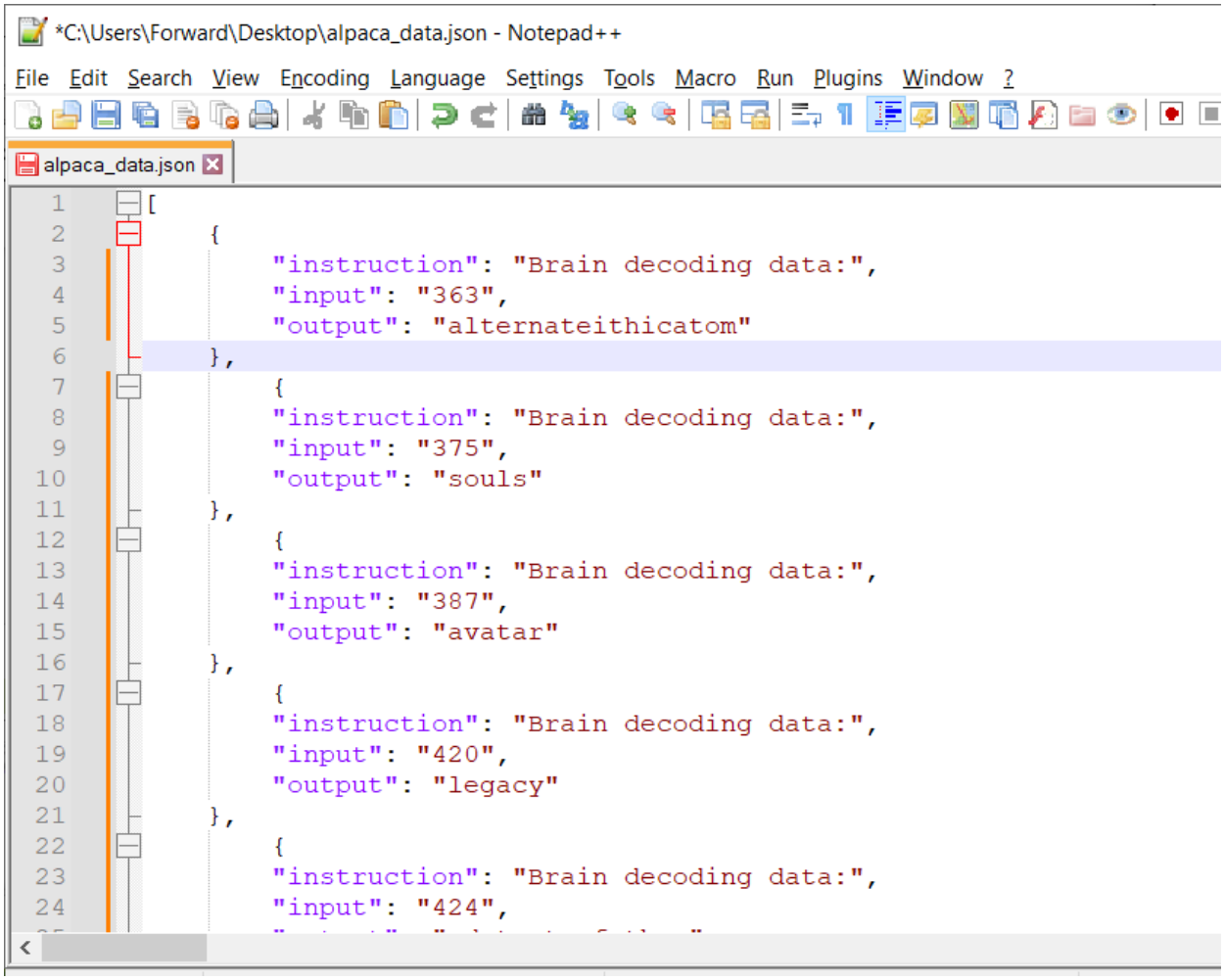



Рисунок 3.16 – Приклад організованих даних для обробки

### 3.3.3 Результат роботи

Після підствляення й заданя параметрім є результати.

Приклад запуску програми. Найкращий результат з декодування:

Оригінал – Avatar is a soulful film with decent legacy

Декодовано – Avatar is a decent film within decent legacy

Схожість 0.75 з оригіналом, 0.125 синтакична схожість, 0.125 хибне розкодування. Також варто враховувати періодичний уявлення внаслідок того що ведика модель мала не вилику вибірку навчання.

Після отримання 10 якісних результатів нижче зображений графік (рис 3.17). Де:

- Зелений колір є однаковим з оригіналом
- Жовтий кольором позначено схожість

- Червоним позначено хибне розкодування

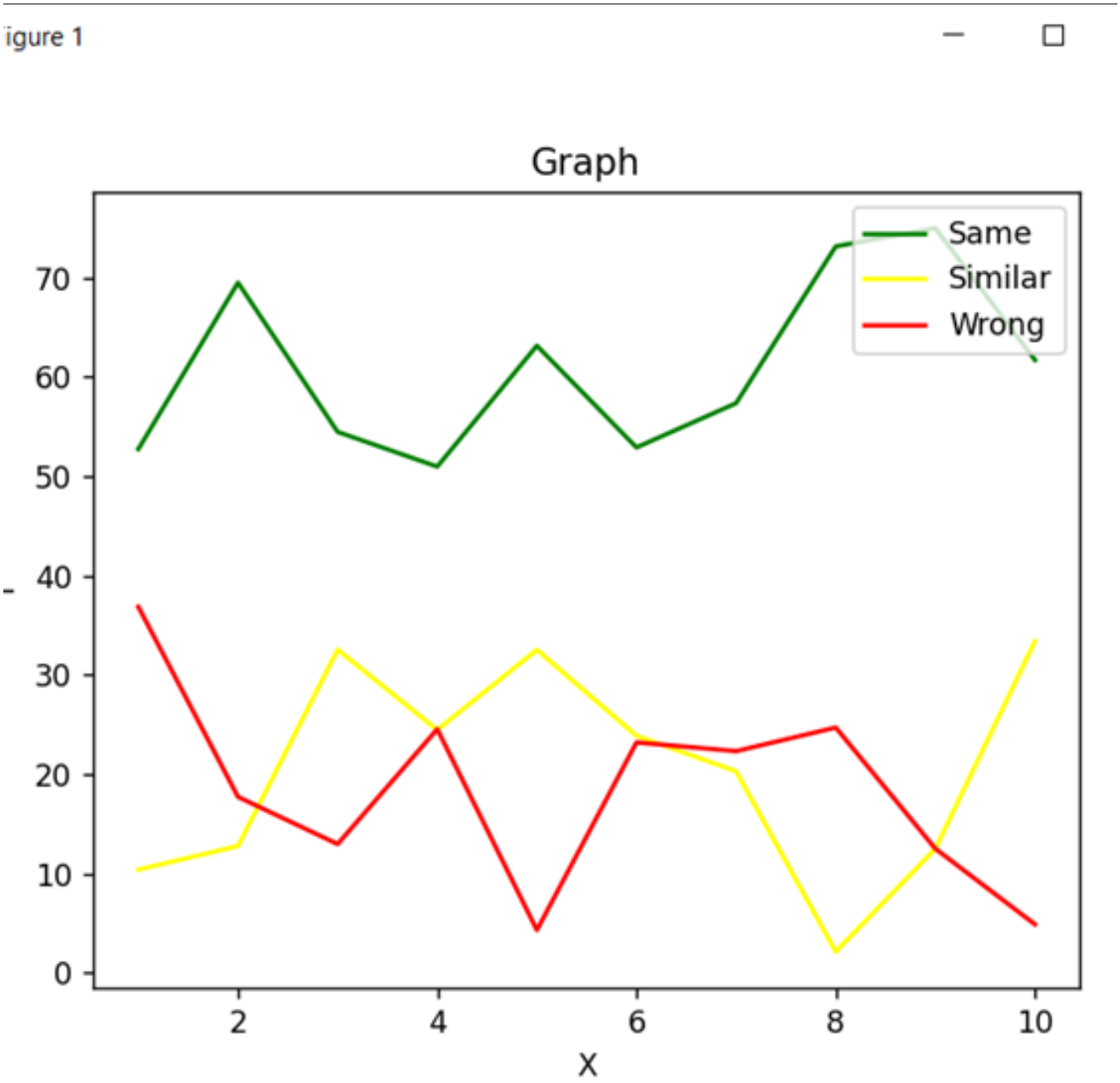



Рисунок 3.17 – Графік схожості результата обробки сигналів.

Середня значення правильних слів після обробок є 61,11. Середне значення схожих є 20,49. Середня значення хибних є 18,40.

Хибні результати без fine tuning – 0.81 що є значно гірше.

Декодовуваня має високий відсоток подібності до оригіналу при високому вмісті вже декодованих даних. При подальшому навчанні моделі збільшується подібність розшифрованого сигналу до оригіналу.

### 3.4 Фільтраці сигналу

#### 3.4.1 Шуми при зчитуванні сигналів мозку

При використанні даниї що не є попередньо обробленими необхідно провести очищення й фільтрацію.

Фільтрація шуму з сигналів є поширеною проблемою в обробці сигналів, і для цього існує багато методів. Найбільш підходящий метод часто залежить від природи шуму і сигналу.

У контексті сигналів мозку (наприклад, ЕЕГ) найпоширенішими типами шуму є

- Шум лінії: Зазвичай це постійна частота 50/60 Гц, спричинена електричним обладнанням.
- М'язовий артефакт: це високочастотний шум, спричинений рухом м'язів користувача.
- Білий шум: Це випадковий шум, який виникає на всіх частотах.
- Дрейф електрода: З часом сигнал від електрода може відхилятися від фактичного значення.

Враховуючи ці поширені типи шуму, загальним підходом є застосування смугового фільтра, який пропускає лише частоти, що становлять інтерес (зазвичай в діапазоні 1-50 Гц для сигналів ЕЕГ).

Приклад програмного забезпечення що може відфільтрувати шуми для збільшення якості даних (рис 3.18).

```python
def butter_bandpass(lowcut, highcut, fs, order=5):
    nyq = 0.5 * fs
    low = lowcut / nyq
    high = highcut / nyq
    b, a = butter(order, [low, high], btype='band')
    return b, a

def butter_bandpass_filter(data, lowcut, highcut, fs, order=5):
    b, a = butter_bandpass(lowcut, highcut, fs, order=order)
    y = lfilter(b, a, data)
    return y



y = butter_bandpass_filter(x, lowcut, highcut, fs, order=6)

import matplotlib.pyplot as plt
plt.figure()
plt.plot(t, x, label='Noisy signal')
plt.plot(t, y, label='Filtered signal')
plt.xlabel('time (seconds)')
plt.hlines([-a, a], 0, T, linestyles='--')
plt.grid(True)
plt.axis('tight')
plt.legend(loc='upper left')
plt.show()
```

Рисунок 3.18 – Приклад коду програми фільтрації шумів

У цьому коді butter_bandpass створює смуговий фільтр Баттерворта, а butter_bandpass_filter застосовує цей фільтр до даних. Параметри lowcut і highcut задають частотний діапазон смугового фільтра, а fs – частоту дискретизації даних.

Приклад використання реалізації очищення (рис. 3.19).

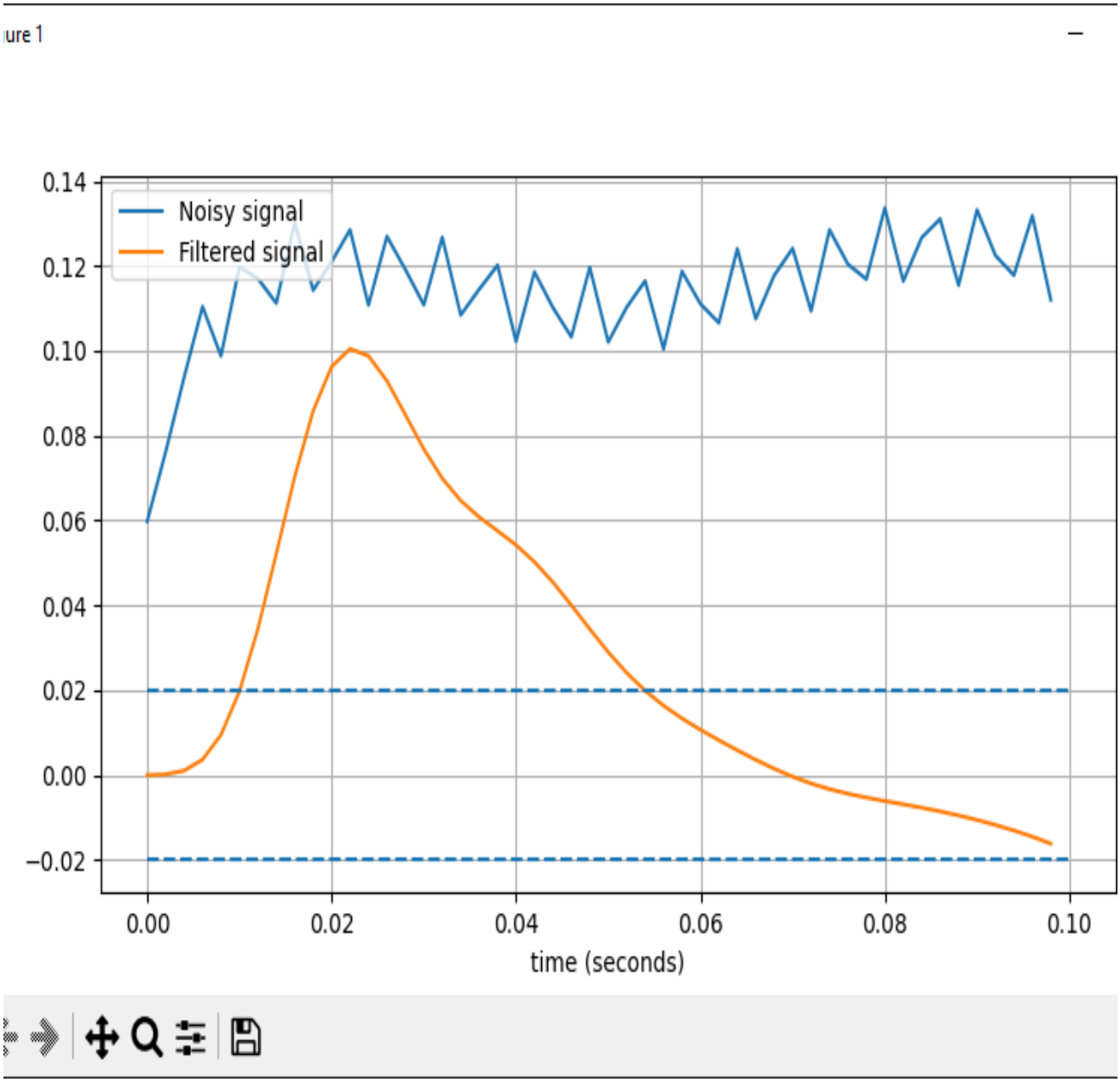
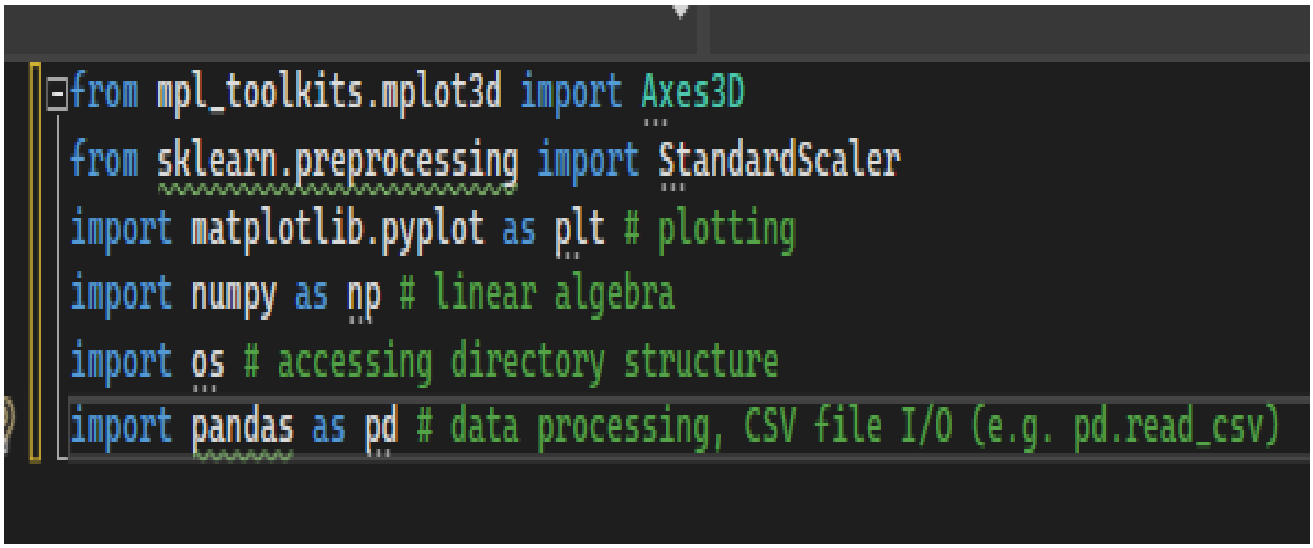



Рисунок 3.19 – Приклад очищення сигналу. Синім позначено неочищений сигнал, помаранчевим позначено очищений.

### 3.5 Модель передбачення стану суб’єкта

Для початку необхідноно встановити необхідні бібліотеки (рис. 3.20)



```
from mpl_toolkits.mplot3d import Axes3D
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt # plotting
import numpy as np # linear algebra
import os # accessing directory structure
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
```

Рисунок 3.20 – Встановлення бібліотек.

Перевірка на зчитування файлу mental-state.csv(152.44 MB) через команду df1.head(5) має видавати наступний результат (рис 3.21)

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 28.326031 | 28.572133 | -67.935922 | 24.112727 | 24.459813 | -5.917549 | 3.501568 | -13.027124 | -1.876510 | -9.98636 |
| 17.448473 | 39.739613 | 33.710473 | 5.910863 | 16.412762 | -8.532382 | -12.929023 | 5.097671 | 1.751522 | 0.48815 |
| 22.459020 | 22.806176 | 14.644605 | 20.750035 | 24.293895 | -11.282420 | 1.136720 | 0.162806 | -14.952338 | -3.10174 |
| 23.092273 | 1.138672 | 18.217086 | 20.891188 | 23.069371 | 2.089529 | 11.776685 | 3.744340 | 1.405053 | -0.32569 |
| 24.370172 | 25.028254 | 28.074277 | 12.775434 | 28.699867 | 5.509730 | -8.577628 | 1.409164 | 8.605645 | -4.94321 |

Рисунок 3.21 – Дані mental-state.csv

Графіки розподілу, є візуальним представленням даних, які показують частоту значень у наборі даних. Вони використовуються для розуміння розподілу точок даних у різних категоріях або діапазонах. (рис 3.22).

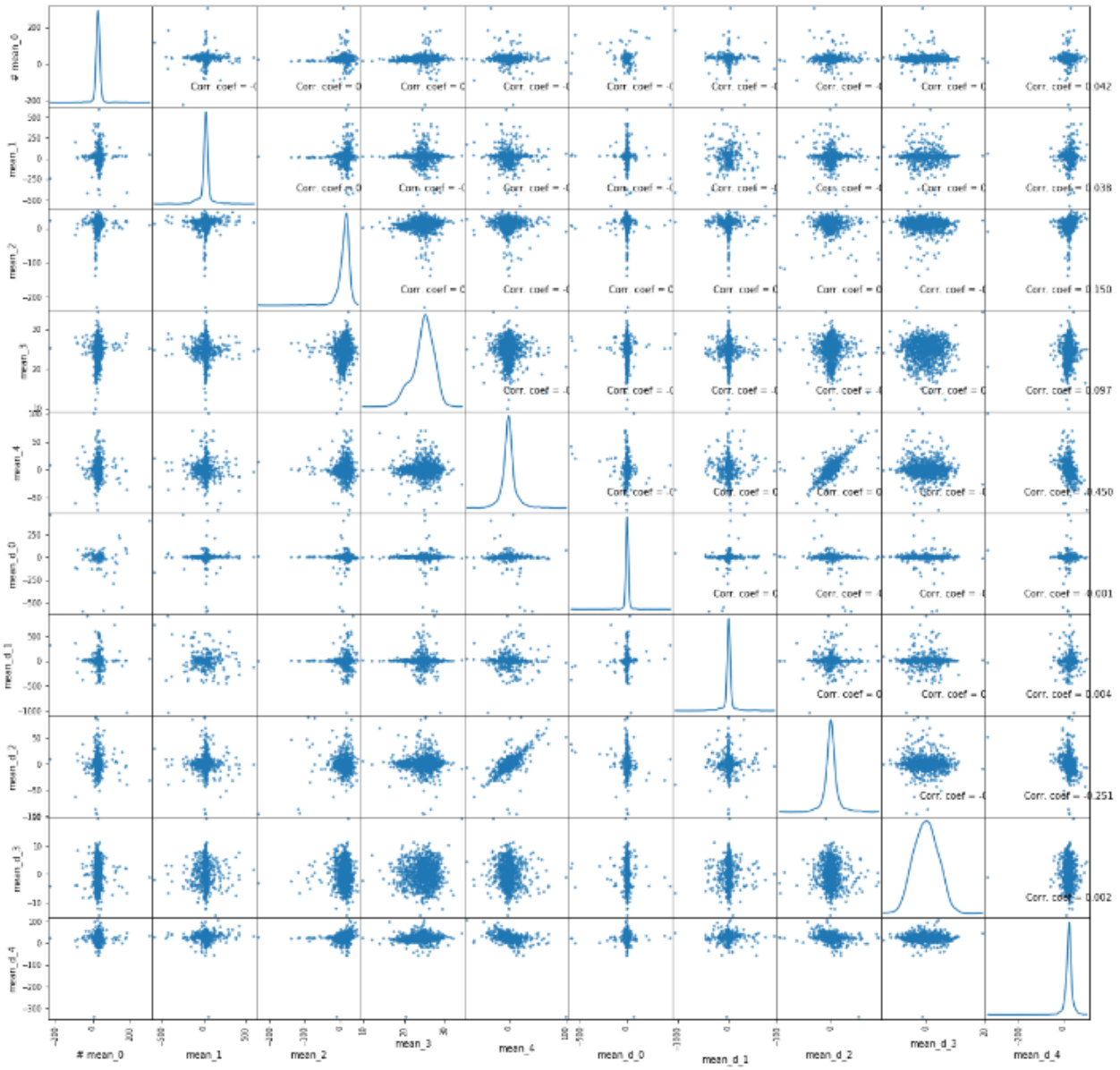



Рисунок 3.22 – Графіки розподілу

В результаті отримана модель передбачення психічного стану суб’єкта з точністю до 70%.

### 3.6 Розрахунок економічного ефекту

В даному розділі проводиться оцінка основних характеристик програмного продукту й можливих альтернатив рішення проблеми.

З економічної перспективи, не інвазивні системи мають перевагу у вартості та привабливості на ринку. Вони є безпечними, не включають ризики, пов'язані з хірургічними процедурами, і мають простіший спосіб використання, що робить їх більш зручними для споживачів. Проте, їхній сигнал може бути менш якісним, що обмежує їхню ефективність. З іншого боку, інвазивні системи забезпечують кращу якість сигналу та мають потенціал для більш вдосконалених застосувань, але вони дорожчі через хірургічну імплантацію, пов'язані ризики для здоров'я та регуляторні перешкоди, що можуть обмежити їхню розповсюдженість на ринку. Додатково, використання інвазивних процедур у немедичних цілях може бути небезпечним.

Вибір мови програмування для розробки може суттєво вплинути на продуктивність та впровадження технології. Python має перевагу в меншому часу витрат на розробку. З іншого боку, C++ є мовою, спрямованою на продуктивність, і може бути більш підходящою для додатків, які потребують обробки в реальному часі та низькорівневої взаємодії з апаратурою. Однак розробка на цій мові може бути більш трудомісткою та витратною.

Головна функція F0 – розробка програмного продукту, який аналізує сигнали та застосовує команди контролю комп’ютерними приладами. Від конкретної мети, можна зробити і виділити наступні основні функції цього програмного забезпечення ПЗ:

F1 – метод створення ПЗ обробка сигналів.

F2 – метод отримання сигналу;

F3 – вибір мови програмування;

Кожна з основних наведених функцій може бути реалізована через декілька варіантів реалізації.

Функція F1:

а) універсальна модель;

б) конкретна модель.

Функція F2:

а) неінвазивна система;

б) інвазивна система;

Функція F3:

а) мова програмування Python;

б) мова програмування C++;

Варіанти реалізації основних важливих функцій наведені нижче у морфологічній карті (рис. 3.23 ).

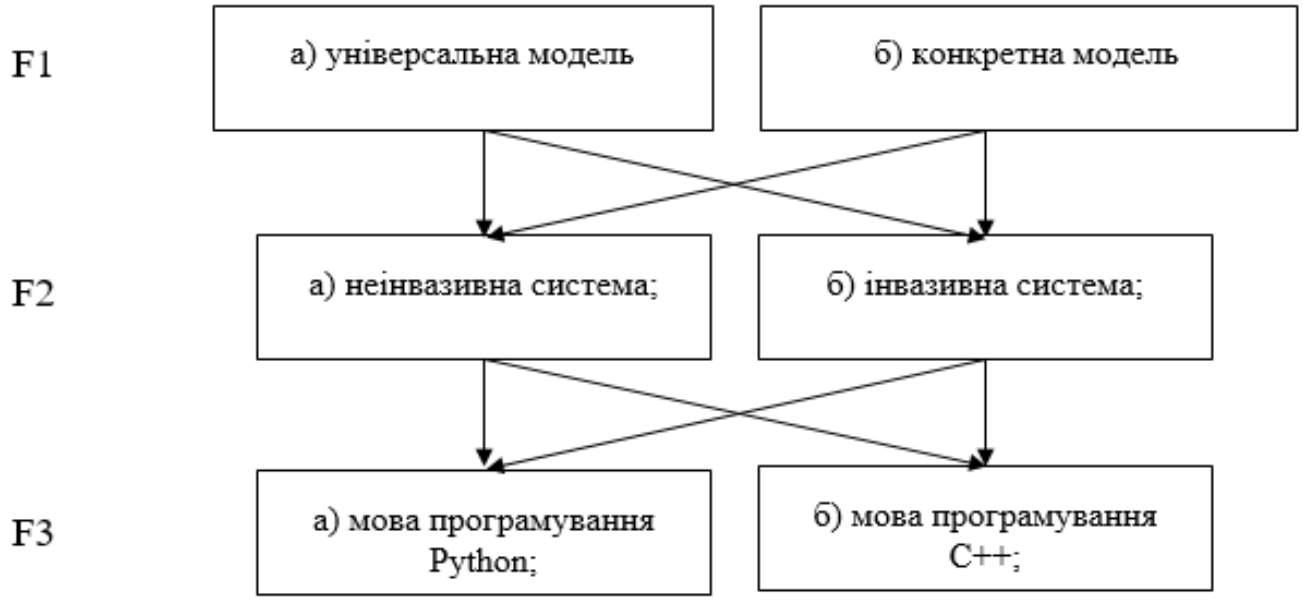



Рисунок 3.23 – Морфологічна карта

За результатами проведеного функціонально-вартісного аналізу ПЗ що найоптимальнішим варіант КТЕР1 = $8{,}22 \cdot 10^{-6}$.

Вартість витрат становить 852006,06 грн.

Цей варіант реалізації програмного продукту має наступні параметри:

- а) універсальна модель;
- а) неінвазивна система;
- а) мова програмування Python;

Неінвазивні системи зазвичай більш комфортні для користувача і менш ризиковані і є рекомендованим способом отримання біологічних даних. У

поєднанні з універсальністю мови програмування Python вони можуть полегшити розробку ефективного, зручного та доступного програмного забезпечення відповідаючи вимогам предстанної задачі.

**Висновки до розділу 3**

Найкраща схожість моделі LLM – 0.75 з оригіналом. 0.125 хибне розкодування. 0.125 схоже за суттю. Середня значення правильного результату після обробок є 61,11. Середне значення схожих є 20,49. Середня значення хибних є 18,40.

Без fine tuning в найкращих випадках дає низьку схожість з оригіналом передбачення – 0.19 що є значно гіршим результаттом. Порівнюючи значення хибних 0.184 (fine tunned LLM) з 0.81 видно високий вплив біологічних сигналів на відсутність хибного результату.

Декодовуваня має високий відсоток подібності до оригіналу при високому вмісті вже декодованих даних. При подальшому навчанні моделі збільшується подібність розшифрованого сигналу до оригіналу. Також варто враховувати можливі галюцінації моделі внаслідок особливості структури описаної моделі. Розмір тренувальних даних є важливим для високого рівня схожості.

Проте варто враховувати що за межами навчальних даних якість розкодування має потенціал впасти до рівня без fine tuning.

Було опрацьовано й реалізовано передбачення психічного стану суб’єкта за даними біологічних сигналів. В результаті отримана модель дає передбачення психічного стану суб’єкта з точністю до 70%.

Було встановлено й застосовано Alpaca large language model для допомоги розшифрування сигналів мозку шляхом передбачення наступного слова користувачем за сигналами мозку користувача. Присуноктній недолік повільності виконання через нестачу VRAM комп’ютера розробника. Для досягнення максимальної ефективності у реалізації програмного застосунку, за допомогою економічного аналізу були ретельно вибрані компоненти та

технології, які стали основою подальшої розробки. Цей вибір забезпечує надійну та стійку платформу для реалізації поставлених завдань з найвищою продуктивністю.

# ЗАГАЛЬНІ ВИСНОВКИ

У дипломній роботі було приділено увагу біологічним даним внаслідок стрімкого розвитку медицини й зростанню потребності аналізу біологічних сигналів.

Метою роботи було використання якісних даних для розшифрування біологічних сигналів.

***Завдання, які були реалізовані у ході роботи:***

1. Аналіз інформації з наукових джерел показав перспективність направлення.
2. Обрання мови програмування для виконання роботи є мова програмування python, завдяки зручним бібліотекам.
3. Визначення вимог програмного застосунку було виконано при розробці й закріплено результатом функціонально-вартісного аналізу.
4. Реалізація програмного застосунку було виконано для подальшого аналізу біологічних сигналів.
5. Розшифрування біологічних сигналів було проведено за застосунком, проаналізовано якість виходу, визначено критичну важливість обробки біологічних даних таких як прибирання шумів.

***У роботі було зроблено розділ аналітичного огляду літературних джерел.***

Мозок - система зі змінними параметрами, що можуть перетворюватися в біологічні сигнали. Вибір між інвазивними та неінвазивними методами зчитування вимагає балансу між роздільною здатністю, якістю сигналу та безпекою. Неінвазивні методи виглядають привабливішими через безпеку, відсутність хірургічних ризиків та простоту використання. Однак, при використанні технологій зчитування мозкових сигналів, важливо дотримуватися заходів безпеки для запобігання злому та зловживанню. Ризики включають

порушення приватності, незахищеність від несанкціонованого доступу до даних та можливість використання технологій для небезпечних цілей.

***У роботі було зроблено теоритичну частину.***

Очищення мозкових сигналів від шуму та артефактів є ключовим для ефективного збільшення якості біологічних сигналів. Також це забезпечує точність і надійність розпізнавання сигналів, підвищує швидкість обробки системами і зменшує ризик помилок, що критично для медичних застосувань та корисно для інших застосувань.

***У роботі було зроблено практичу частину.***

Модель LLM показала високу схожість з оригіналом (0.75), та низький рівнем хибного розкодування (0.125). Середні значення правильних, схожих і хибних результатів становлять 61.11, 20.49 і 18.40 відповідно. Без fine tuning схожість з оригіналом значно знижується (0.19). Високий вплив біологічних сигналів на зменшення хибного результату. Декодування показує високу подібність до оригіналу, що збільшується з навчанням моделі, але може впасти за межами навчальних даних. Модель передбачає психічний стан суб'єкта з точністю до 70%. Використано Alpaca large language model для розшифрування сигналів мозку. Обмеження включають повільність виконання через нестачу VRAM. Ефективність забезпечується ретельним вибором компонентів і технологій.

***У роботі було зроблено розрахунок економічного ефекту.***

У частині функціонально-вартісного аналізу було проведено аналіз ПЗ та варіантів його реалізації за оцінками експертів. Визначено й обґрунтовано оптимальний варіант та передбачено витрати.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ


1. An fMRI dataset during a passive natural language listening task [Електронний ресурс] https://openneuro.org/datasets/ds003020/versions/2.0.1 (дата звернення: 20.05.2023).
2. Back, T., 1996. Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms. Oxford university press.
3. Bird, J.J.; Manso, L.J.; Ribeiro, E.P.; Ekart, A.; Faria, D.R. A Study on Mental State Classification Using EEG-Based Brain-Machine Interface. In Proceedings of the 2018 International Conference on Intelligent Systems (IS), Funchal, Portugal, 25–27 Вересня 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 795–800. DOI:10.1109/IS.2018.8710576
4. CMake downloader [Електронний ресурс] / re-search. https://cmake.org/download/ (дата звернення: 20.05.2023).
5. Crosson B, Ford A, McGregor KM, Meinzer M, Cheshkov S, Li X, Walker-Batson D, Briggs RW (2010). Functional imaging and related techniques: an introduction for rehabilitation researchers. *Journal of Rehabilitation Research and Development*(2): vii–xxxiv. PMC 3225087. PMID 20593321. doi:10.1682/jrrd.2010.02.0017.
6. Diekelmann, Susanne; Born, Jan (2010). Slow-wave sleep takes the leading role in memory reorganization. *Nature Reviews Neuroscience*(3): 218–218. ISSN 1471-003X. doi:10.1038/nrn2762-c2.
7. Diekelmann, Susanne; Born, Jan (2010). The memory function of sleep. *Nature Reviews Neuroscience*. ISSN 1471-003X. doi:10.1038/nrn2762.
8. Easily build, customize and control your own LLMs [Електронний ресурс] / re-search. https://github.com/stochasticai/xturing (дата звернення: 20.05.2023).

9. Gruzelier, John H. (1 липня 2014). EEG-neurofeedback for optimising performance. I: A review of cognitive and affective outcome in healthy participants. Neuroscience & Biobehavioral Reviews (англ.) 44. с. 124–141. ISSN 0149-7634. doi:10.1016/j.neubiorev.2013.09.015.
10. GUI github [Електронний ресурс] / re-search. https://desktop.github.com/ (дата звернення: 20.05.2023).
11. Jarrett, C. (17 листопада, 2014). *Great Myths of the Brain*. John Wiley & Sons. ISBN 978-1-118-31271-1.
12. JetBrains. PyCharm: the Python IDE for Professional Developers. [Електронний ресурс] https://www.jetbrains.com/pycharm/ (дата звернення: 20.05.2023).
13. Juergen Fell & Nikolai Axmacher (February 2011). The role of phase synchronization in memory processes. *Nature reviews. Neuroscience*(2): 105–118. PMID 21248789
14. LLM Alpaca [Електронний ресурс] / re-search. GitHub. URL: https://github.com/cocktailpeanut/dalai (дата звернення: 20.05.2023).
15. Microsoft. Visual Studio. [Електронний ресурс] https://visualstudio.microsoft.com/ (дата звернення: 20.05.2023).
16. Müller-Putz, Gernot R.; Scherer, Reinhold; Pfurtscheller, Gert; Rupp, Rüdiger (1 липня 2005). EEG-based neuroprosthesis control: A step towards clinical practice. Neuroscience Letters (англ.) 382 (1). с. 169–174. ISSN 0304-3940. doi:10.1016/j.neulet.2005.03.021.
17. Pearson, K., 1895. Note on regression and inheritance in the case of two parents. Proceedings of the Royal Society of London, 58, pp.240-242.
18. Pfurtscheller, Gert; Müller, Gernot R; Pfurtscheller, Jörg; Gerner, Hans Jürgen; Rupp, Rüdiger (6 листопада 2003). 'Thought' – control of functional electrical stimulation to restore hand grasp in a patient with tetraplegia. *Neuroscience Letters* (англ.). с. 33–36. ISSN 0304-3940. doi:10.1016/S0304-3940(03)00947-9
19. Predict what a mouse sees by decoding brain signals [Електронний ресурс].

– 2023. – 2 хв. – Режим доступу: https://www.youtube.com/watch?v=bWZiPT8IhRY

20. Remsik, Alexander B.; van Kan, Peter L. E.; Gloe, Shawna; Gjini, Klevest; Williams, Leroy; Nair, Veena; Caldera, Kristin; Williams, Justin C. та ін. (2022). BCI-FES With Multimodal Feedback for Motor Recovery Poststroke. Frontiers in Human Neuroscience 16. ISSN 1662-5161. PMC PMC9296822. PMID 35874158. doi:10.3389/fnhum.2022.725715.
21. Rogala, Jacek; Jurewicz, Katarzyna; Paluch, Katarzyna; Kublik, Ewa; Cetnarski, Ryszard; Wróbel, Andrzej (2016). The Do's and Don'ts of Neurofeedback Training: A Review of the Controlled Studies Using Healthy Adults. *Frontiers in Human Neuroscience* **10**. ISSN 1662-5161. PMC PMC4911408. PMID 27378892. doi:10.3389/fnhum.2016.00301.
22. Schneider, S., Lee, J.H. & Mathis, M.W. Learnable latent embeddings for joint behavioural and neural analysis. Nature 617, 360–368 (2023). https://doi.org/10.1038/s41586-023-06031-6
23. Sitaram, Ranganatha; Ros, Tomas; Stoeckel, Luke; Haller, Sven; Scharnowski, Frank; Lewis-Peacock, Jarrod; Weiskopf, Nikolaus; Blefari, Maria Laura та ін. (2017-02). Closed-loop brain training: the science of neurofeedback. Nature Reviews Neuroscience (англ.) 18 (2). с. 86–100. ISSN 1471-0048. doi:10.1038/nrn.2016.164.
24. Smith-Bindman R (July 2010). Is computed tomography safe?. *The New England Journal of Medicine*(1): 1–4. PMID 20573919. doi:10.1056/NEJMp1002530.
25. Tatum, William O. (2014). *Handbook of EEG interpretation*. Demos Medical Publishing. с. 155–190. ISBN 9781617051807. OCLC 874563370
26. Tsai LL, Grant AK, Mortele KJ, Kung JW, Smith MP (October 2015). A Practical Guide to MR Imaging Safety: What Radiologists Need to Know. *Radiographics* **35** (6):1722–37. PMID 26466181. doi:10.1148/rg.2015150108.

27. University of Waikato. 2018. InfoGainAttributeEval. [Електронний ресурс] Weka.sourceforge.net. Available at: http://weka.sourceforge.net/doc.dev/weka/attributeSelection/InfoGainAttributeEval.html.
28. University of Waikato. 2018. OneR. [Електронний ресурс] Weka.sourceforge.net. Available at: http://weka.sourceforge.net/doc.dev/weka/classifiers/rules/OneR.html
29. Witten, I.H., Frank, E., Hall, M.A. and Pal, C.J., 2016. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann.
30. Wolters CH, Anwander A, Tricoche X, Weinstein D, Koch MA, MacLeod RS (April 2006). Influence of tissue conductivity anisotropy on EEG/MEG field and return current computation in a realistic head model: a simulation and visualization study using high-resolution finite element modeling. *NeuroImage*(3): 813–26. PMID 16364662. doi:10.1016/j.neuroimage.2005.10.014.
31. Анатомія мозку. [Електронний ресурс] Архів оригіналу за 4 березня 2016.
32. Ахтемійчук Ю.Т., Вовк Ю.М., Дорошенко С.В., Кобзар О.Б., Ковальський М.П., Півторак В.І., Прокопець К.О., Радомська Н.Ю., Радомський О.А., Пархоменко М.В., Хворостяна Т.Т. (2010). Оперативна хірургія та топографічна анатомія. Київ: Медицина. с. 474. ISBN 978-617-505-058-3.
33. Головацький А. С., Черкасов В. Г., Сапін М. Р., Федонюк Я. І. Анатомія людини. У трьох томах. Том 2. — 2007. — Нова книга. — 456 с. — підручник, що рекомендований МОН та МОЗ України для Вищих медичних навчальних закладів України найвищих рівнів акредитації.
34. Про мозкові оболони. Архів оригіналу за 4 березня 2016. Процитовано 18 грудня 2018.